LARAVEL PROJECT DOCUMENTATION

Usage and Deployment

Cristian Datu

Table of Contents

Laravel Project

Project description

Laravel Manager is a product designed to control users and Terms of Service with ease, providing a simple and intuitive interface for user signup, management and notifications. The integrated Terms of Service manager enables complete control over editing and publishing terms, with automatic user notification.

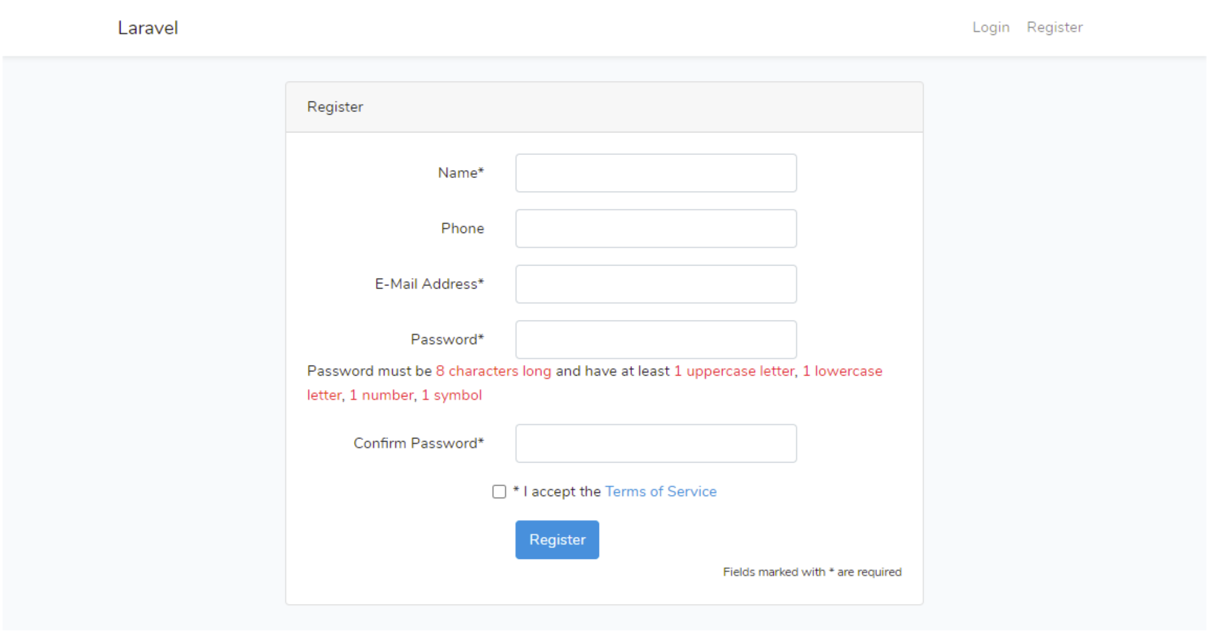
Used Technologies:

- Php v7.4.9 as runtime environment
- Laravel Framework v7.25.0
- Laravel UI v2.1 plugin for user interface scaffolding
- Laravel Sanctum V2.5 plugin for API authentication using session
- Bootstrap v4 for layout and design
- Vue.js V2.5.7 for SPA (Single Page Application) functionalities
- MySQL v8 relational database server for persisting application data

Front-End Functionality

User registration

- Allows creation of new user accounts
- Users must provide their full name, email address, password and accept the site's Terms of Service
- Phone number can also be provided, although optional
- To ensure a minimum password strength, it must contain at least 8 characters and have at least 1 uppercase letter, 1 lowercase letter, 1 digit and 1 symbol (~ ! @ # $ % ^ & * ( ) _ + = - ` { } [ ] : " ; ' < > , . ? | / \ )
- A verification email with validation link will be issued
- The registration system does not allow duplicate email addresses in the database (no two users may signup with the same address)


Laravel

Login Register

Register

Name*

Phone

E-Mail Address*

Password*

Password must be 8 characters long and have at least 1 uppercase letter, 1 lowercase letter, 1 number, 1 symbol

Confirm Password*

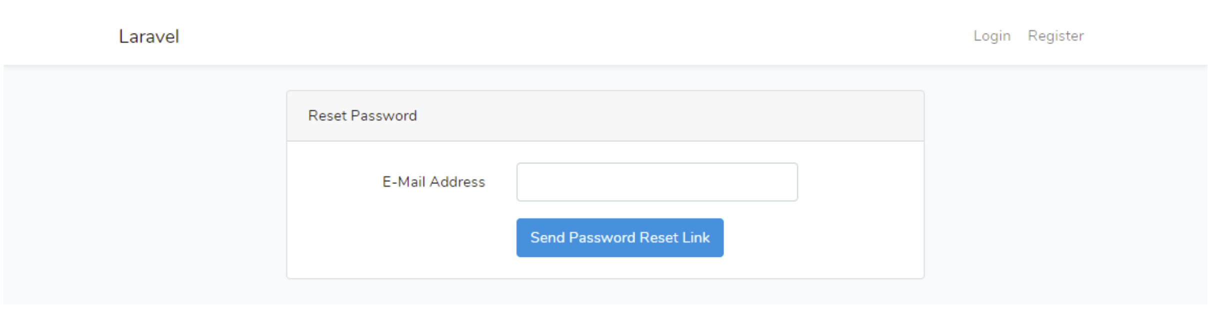
☐ * I accept the Terms of Service

Register

Fields marked with * are required

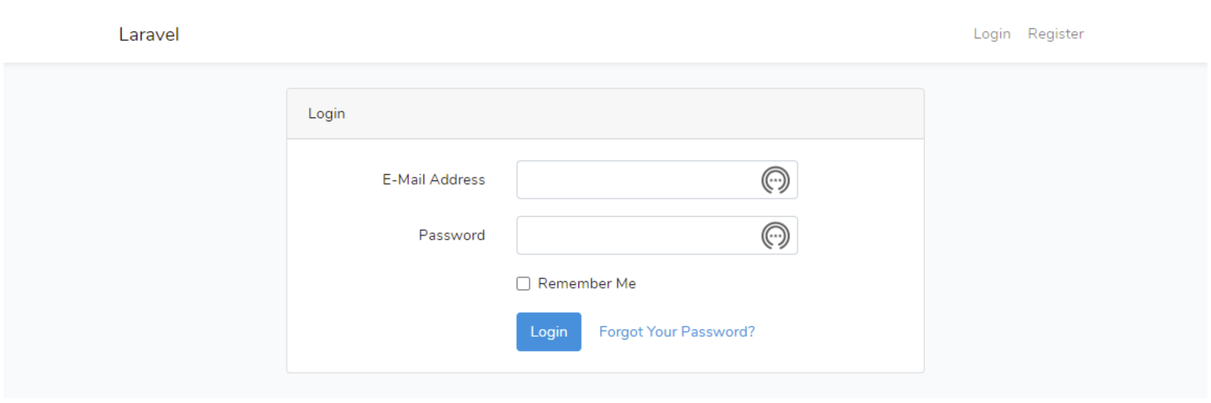
User password reset

- Allows users to reset their passwords based on provided email address
- Email notification containing the reset link will be issued
- Reset link validity is set to 60 minutes
- User may generate reset links multiple times



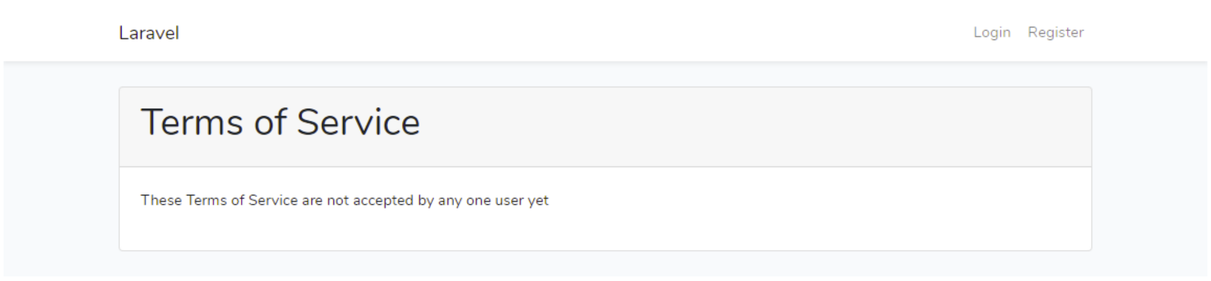
User authentication

- Users may log in to the system using their email address and password
- Security from brute-force hacking is provided by limiting successive login attempts to 10, with a 30 second time delay to the next 10 attempts



Terms of Service review

- Page displaying latest version of the site's Terms of Service
- Access is public
- Content for this page is manageable from the back-end



Terms of Service

These Terms of Service are not accepted by any one user yet

Back-End Functionality

In order to access the back-end, user must be authenticated trough the login page.

All users have access to add/edit/remove people and add/edit/remove/publish/un-publish Terms of Service without any constraints.

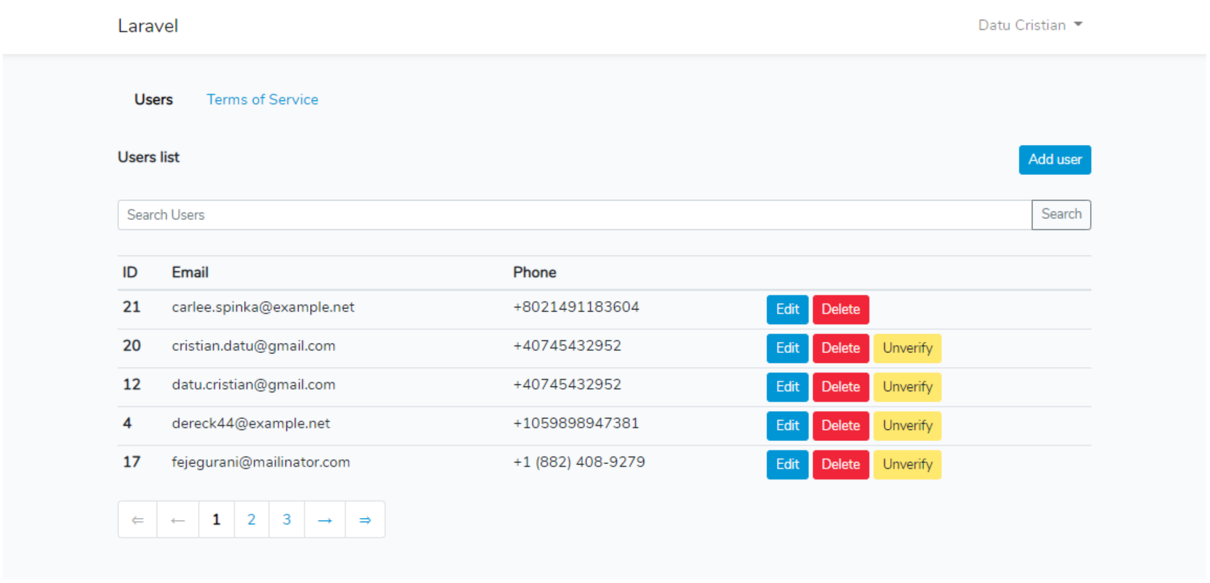
Back-end Header

- Logo (left) is a link to the dashboard page
- In the right part of the header, users name is displayed. When clicked, the “Logout” button appears and enables the user to terminate the session

Laravel
Datu Cristian

Dashboard

- Displays a list of users (with ID, email and phone) registered in the system
- The list is paginated, having 5 items per page
- Users can be searched by name, email or phone
- Users can be edited using provided “Edit” button
- Users can be deleted using provided “Delete” button
- User email can be unverified by pressing the “Unverify” button. Un-verified users will have to re-validate their email address upon login



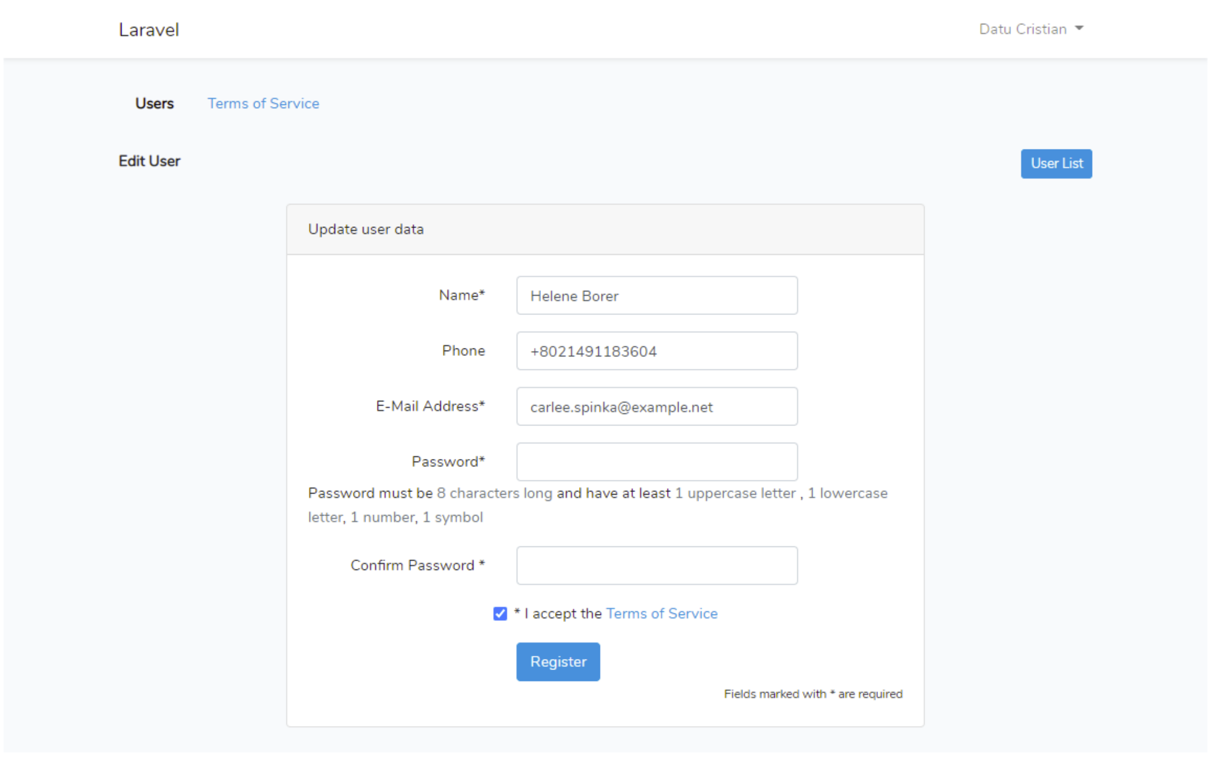
Users list

Search Users

ID	Email	Phone	
21	carlee.spinka@example.net	+8021491183604	Edit Delete
20	cristian.datu@gmail.com	+40745432952	Edit Delete Unverify
12	datu.cristian@gmail.com	+40745432952	Edit Delete Unverify
4	dereck44@example.net	+1059898947381	Edit Delete Unverify
17	fejegurani@mailinator.com	+1 (882) 408-9279	Edit Delete Unverify

User Edit Page

- Can be accessed by clicking the “Edit” button associated to each user
- All fields, except “Phone” are mandatory and data validation rules applied are the same as on the Registration front-end page
- Information will be saved by pressing the “Save” button
- After successfully saving user information, the page will be redirected to the users list page
- Navigation back to the user list page can also be achieved by clicking on the “User List” button



Update user data

Name* Helene Borer

Phone +8021491183604

E-Mail Address* carlee.spinka@example.net

Password*

Password must be 8 characters long and have at least 1 uppercase letter , 1 lowercase letter, 1 number, 1 symbol

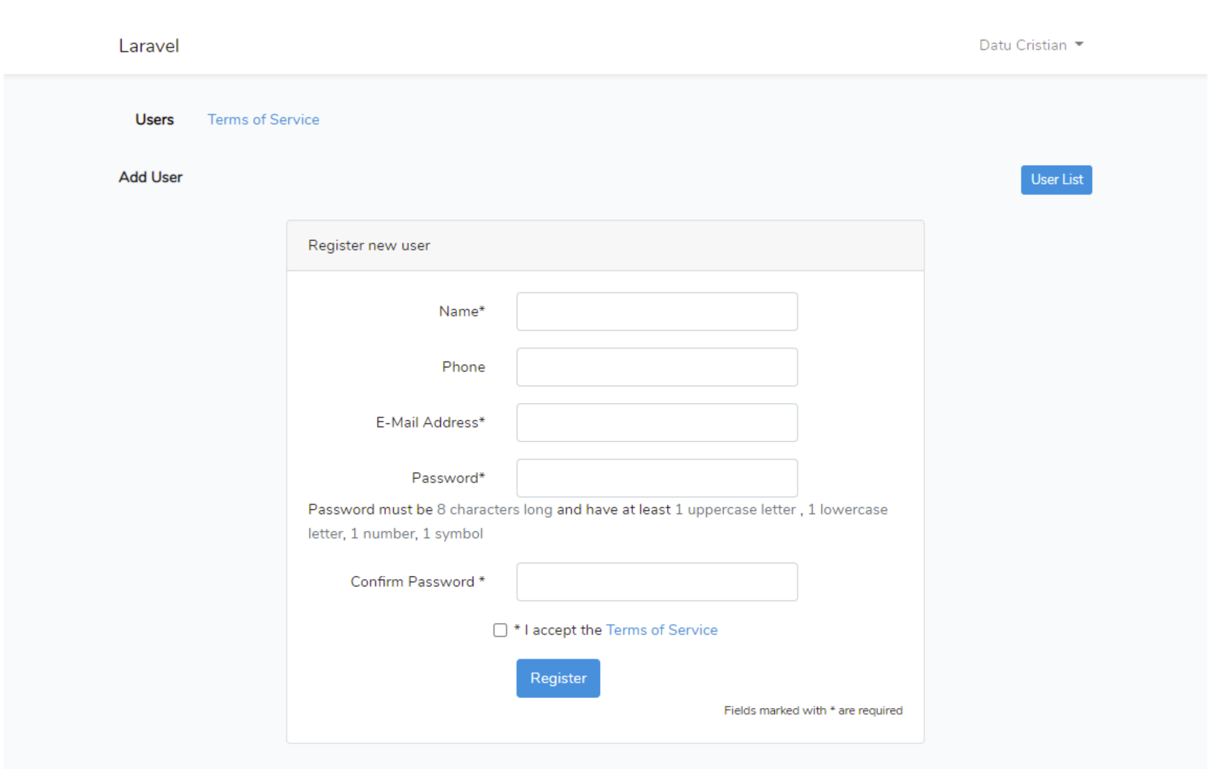
Confirm Password *

✓ * I accept the Terms of Service

Fields marked with * are required

User Create Page

- Has the same functionality as the user edit page, except fields will not be pre-populated with existing values in the database
- All validation rules apply as on the Register and User Edit page



Register new user

Name*	
Phone	
E-Mail Address*	
Password*	

Password must be 8 characters long and have at least 1 uppercase letter , 1 lowercase letter, 1 number, 1 symbol

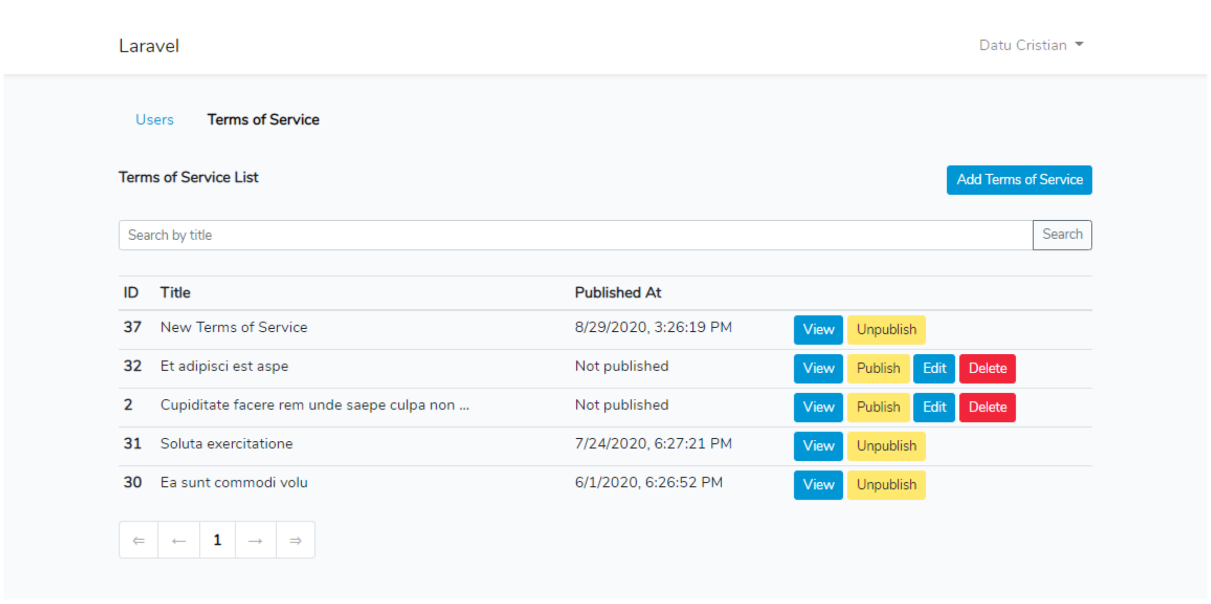
Confirm Password *	

☐ * I accept the Terms of Service

Fields marked with * are required

Terms of Service

- Page listing all available Terms of Service (ID, administrative title and publishing date if applicable)
- Terms of Service (published and unpublished) can be viewed by pressing “View” button
- Terms can be searched by title
- To edit an item, press “Edit”.
- To publish an item, press “Publish”
- To delete an item, press “Delete”
- Some rules apply
    - Terms already published cannot be edited or deleted, but can be unpublished
    - Only unpublished terms can be edited, delete or published
    - When Terms of Service are published, the system issues emails to all users with a link to the newly published terms and an “Accept” link

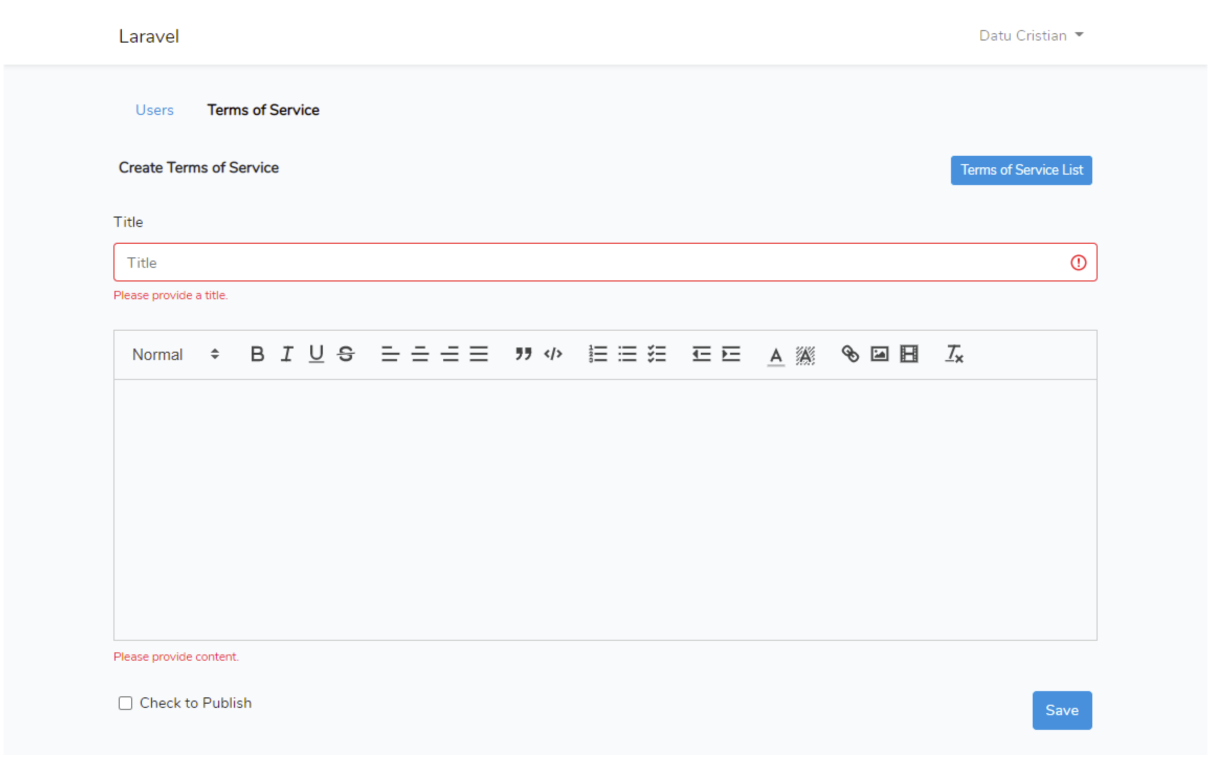


Terms of Service List

ID	Title	Published At	
37	New Terms of Service	8/29/2020, 3:26:19 PM	View Unpublish
32	Et adipisci est aspe	Not published	View Publish Edit Delete
2	Cupiditate facere rem unde saepe culpa non ...	Not published	View Publish Edit Delete
31	Soluta exercitatione	7/24/2020, 6:27:21 PM	View Unpublish
30	Ea sunt commodi volu	6/1/2020, 6:26:52 PM	View Unpublish

Add Terms of Services

- Page may be accessed by clicking on the “Add Terms of Services” button in the listing page
- Title and Content fields are required in order to save
- The title cannot be longer than 255 characters
- Content can be edited using the WYSISWYG (What You See Is What You Get) HTML editor included, allowing basic text formatting
- By checking the “Check to Publish” checkbox, the newly created Terms of Service will automatically be published and all users informed via email
- After successfully saving the new Terms of Service, page will automatically redirect to the “Terms of Service List” page mentioned above



Create Terms of Service

Title

Title

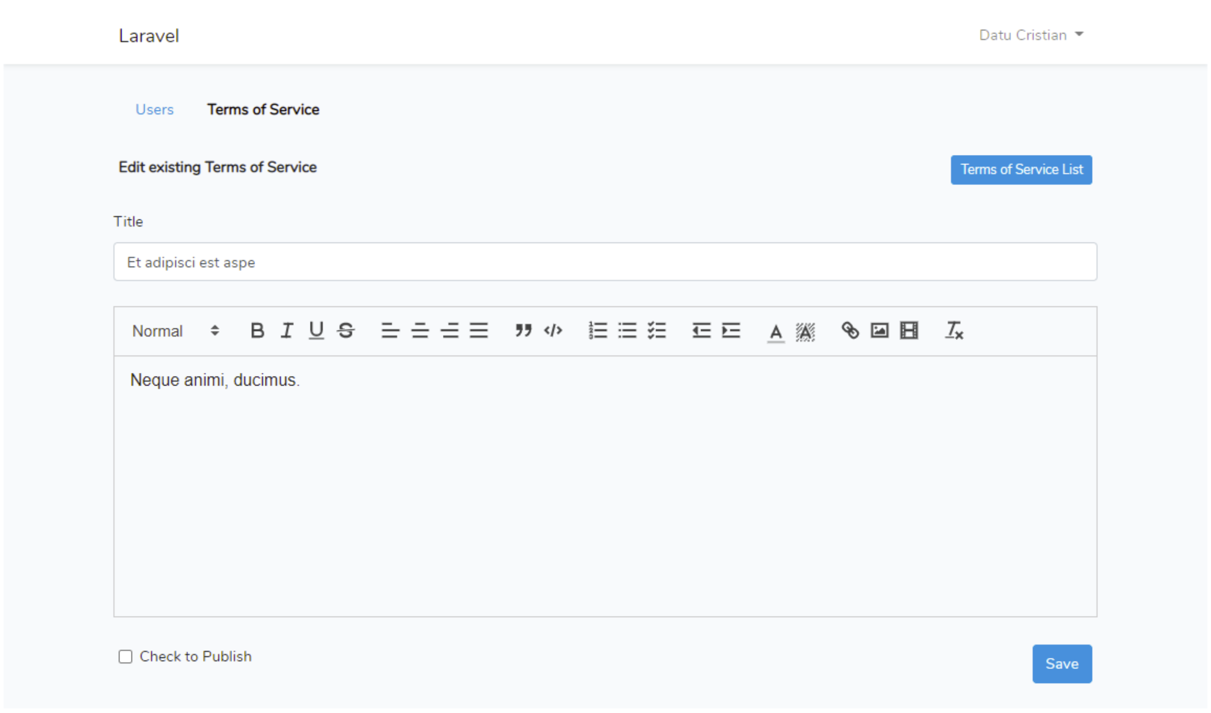
Please provide a title.

Please provide content.

Check to Publish

Edit Terms of Service

- All of the rules from the “Add Terms of Service” page apply
- Fields will be pre-populated with existing data from the database



Edit existing Terms of Service

Title

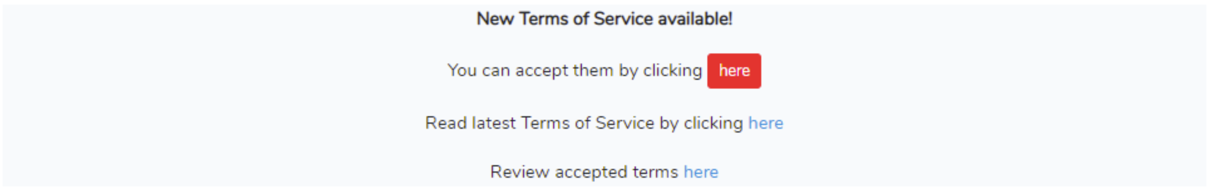
Et adipisci est aspe

Neque animi, ducimus.

Check to Publish

New Terms of Service Available Header

- Informs the current user that new Terms of Service were published, but only if the user did not accept them yet
- User has the option to accept the new terms by clicking the red “here” button on line 2
- User may read the new terms by accessing the “here” link in line 3
- User may also review the previously accepted Terms of service by accessing the “here” link in line 4


New Terms of Service available!

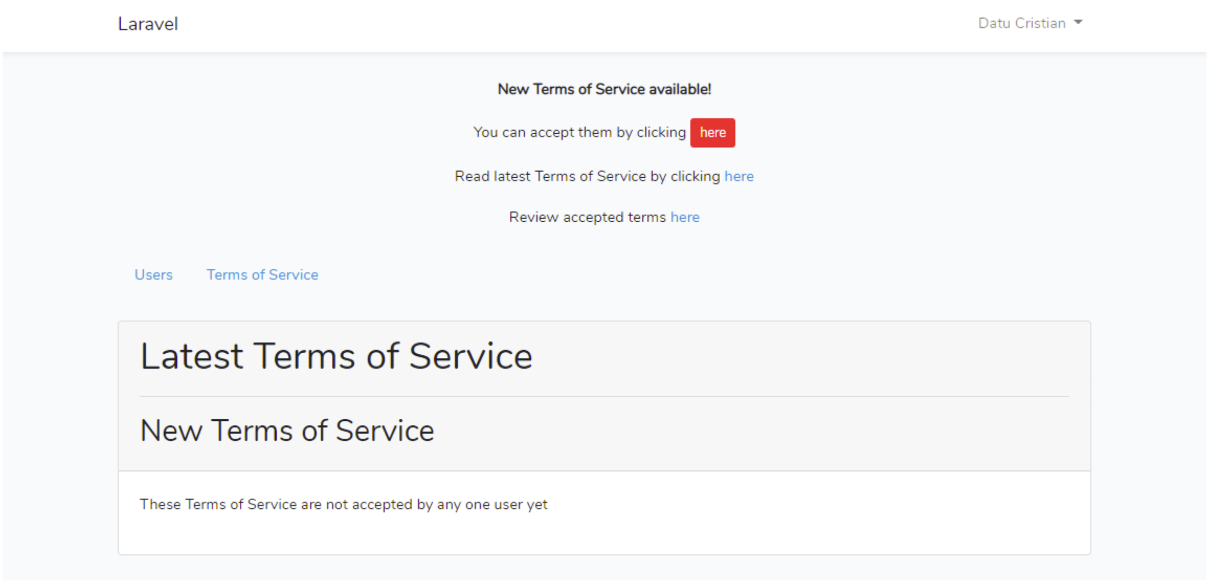
You can accept them by clicking here

Read latest Terms of Service by clicking here

Review accepted terms here

Latest Terms of Service

- Page displaying the latest Terms of Service



New Terms of Service available!

You can accept them by clicking here

Read latest Terms of Service by clicking here

Review accepted terms here

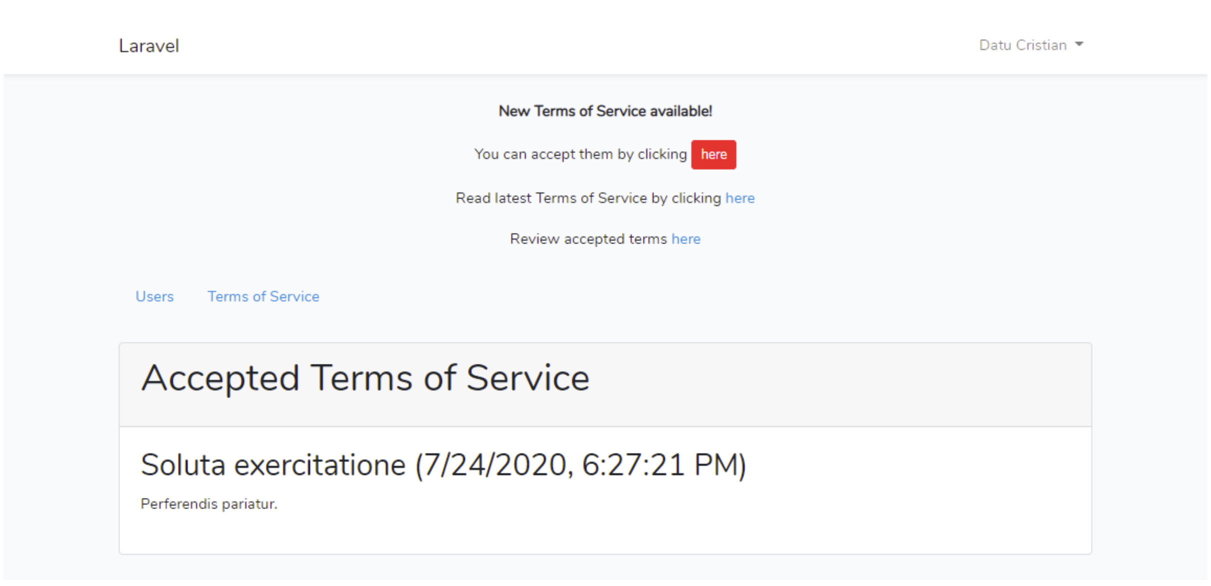
Latest Terms of Service

New Terms of Service

These Terms of Service are not accepted by any one user yet

Previously accepted Terms of Service

- Logged in users may review their currently accepted Terms of Service



Laravel

Datu Cristian

New Terms of Service available!

You can accept them by clicking here

Read latest Terms of Service by clicking here

Review accepted terms here

Users Terms of Service

Accepted Terms of Service

Soluta exercitatione (7/24/2020, 6:27:21 PM)

Perferendis pariatur.

Application Deployment

Minimum Environment Requirements

- Git and Composer
- Linux / Windows operating system
- Apache v2.4 / Nginx v1
- PHP v7.2.5 or higher with the following extensions:
  - BCMath PHP Extension
  - Ctype PHP Extension
  - Fileinfo PHP extension
  - JSON PHP Extension
  - Mbstring PHP Extension
  - OpenSSL PHP Extension
  - PDO PHP Extension
  - Tokenizer PHP Extension
  - XML PHP Extension
- MySQL Server v5.6 or newer
- Process monitor (ex. Supervisor - supervisord.org) to manage queue:work process used for processing queued jobs

Setting up the Laravel project with Apache2.4 on Linux

All files must be kept outside the public directory of the Apache server.

Setting up Project Directory

Navigate to the desired destination folder (we will name it /home/user for practical purposes)

Clone the git repository

```
$ cd /home/user
$ git clone https://github.com/cristian-datu/laravel1.git .
```

Navigate to the newly created **laravel1** directory

```
$ cd laravel1
```

Copy the .env.example file to .env

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

Generate a new encryption key

```
$ php artisan key:generate
```

Setting up the Laravel Application

Edit the new .env file and adjust settings:

- Disable debug
  - APP_DEBUG=false
- Setup site URL (in this case https://www.example.com)
  - APP_URL=https://www.example.com
- Setup database connection parameters
  - DB_CONNECTION=mysql
  - DB_HOST=127.0.0.1
  - DB_PORT=3306
  - DB_DATABASE=your_database_name
  - DB_USERNAME=your_database_user
  - DB_PASSWORD=your_database_pass
- Enabled queued jobs to be stored in the database
  - QUEUE_CONNECTION=database
- Setup the domain verification for Sanctum
  - SANCTUM_STATEFUL_DOMAINS=www.example.com
- Setup mailer
  - MAIL_MAILER=smtp
  - MAIL_HOST=smtp_host_name
  - MAIL_PORT=smtp_host_port
  - MAIL_USERNAME=smtp_username
  - MAIL_PASSWORD=smtp_password
  - MAIL_ENCRYPTION=ssl
  - MAIL_FROM_ADDRESS=from_email_address
  - MAIL_FROM_NAME=from_name
- After all settings are adjusted, save and close the .env

After you create the new database identified by your_database_name that can be accessed with the credential specified in the .env file, launch the migration process to create required tables

```
$ php artisan migrate
```

Setting up Apache Virtual Host

Open Apache virtual host file, with administrative privileges, for editing (default path would be /etc/apache2/sites-available/000-default.conf).

Add the following configuration:

```
<VirtualHost *:80>
        ServerName www.example.com
        DocumentRoot /home/user/laravel1/public
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
        RewriteEngine On
        ServerAdmin webmaster@example.com
        <Directory "/home/user/ laravel1/public">
                AllowOverride All
                Options FollowSymLinks +Indexes
                Order allow,deny
                Allow from all
        </Directory>
</VirtualHost>
```

Restart apache to apply new virtual host.

```
$ service apache2 restart
```

Setting Up Nginx

If you are deploying your application to a server that is running Nginx, you may use the following configuration file as a starting point for configuring your web server.

```
server {
        listen 80;
        server_name example.com;
        root /home/usr/laravel1/public;

        add_header X-Frame-Options "SAMEORIGIN";
        add_header X-XSS-Protection "1; mode=block";
        add_header X-Content-Type-Options "nosniff";
```

```
                index index.php;

                charset utf-8;

                location / {

                        try_files $uri $uri/ /index.php?$query_string;

                }


                location = /favicon.ico { access_log off; log_not_found off; }

                location = /robots.txt  { access_log off; log_not_found off; }

                error_page 404 /index.php;

                location ~ \.php$ {

                        fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;

                        fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;

                        include fastcgi_params;

                }


                location ~ /\.(?!well-known).* {

                        deny all;

                }

        }
```

Optimizations

When deploying to production, make sure that you are optimizing Composer's class autoloader map so Composer can quickly find the proper file to load for a given class:

```
        $ composer install --optimize-autoloader --no-dev
```

To give your application a speed boost, you should cache all of your configuration files into a single file using

```
        $ php artisan config:cache
```

You should create a symbolic link at public/storage which points to storage/app/public. You may create the link using:

```
$ php artisan storage:link
```

Optimize route loading:

```
$ php artisan route:cache
```

Optimize view loading

```
$ php artisan view:cache
```

Setting up queue:work

To ensure that our queue:work command keep running all the time, we need to install a process monitor on our server like Supervisor.

If you are on Ubuntu, you can install it via this command:

```
$ sudo apt-get install supervisor
```

Configuring Supervisor

We need to tell Supervisor which queues we want to consume and how many workers we want to create. Create one configuration file for each queue, and store them in the /etc/supervisor/conf.d directory

Configuration files look like this (example taken from the Laravel documentation):

```
[program:laravel-worker]
process_name=%(program_name)s_%(process_num)02d
command=php /home/forge/app.com/artisan queue:work sqs --sleep=3 --tries=3
autostart=true
autorestart=true
user=forge
numprocs=8
redirect_stderr=true
stdout_logfile=/home/user/laravel1/worker.log
stopwaitsecs=3600
```

After you add all the configuration files you need, you’d need to execute the following commands in order to take the new changes into consideration:

```
$ sudo supervisorctl reread

$ sudo supervisorctl update

$ sudo supervisorctl start all
```