**RĪGAS VALSTS TEHNIKUMS**

DATORIKAS NODAĻA

Izglītības programma: Programmēšana

**KVALIFIKĀCIJAS DARBS**

**“Finanšu sistēmas tīmekļa vietne”**

Audzēknis: Valentīns Kaspers

Prakses vadītājs: Igors Litvjakovs

Nodaļas vadītājs: Normunds Barbāns

**Rīga**

**2024**

**ANOTĀCIJA**

Kvalifikācijas darbs ir finanšu sistēmas tīmekļa vietne. Sistēma ļauj automātiski izveidot rēķinus, ilgstoši un droši elektroniski uzglabāt rēķinus eurocipari.lv serverī, informēt klientus par rēķinu apmaksu un nodrošināt iespēju apmaksāt rēķinus uzņēmuma klientiem. Sistēma tika izstrādāta, izmantojot PHP programmēšanas valodu, LARAVEL frameworku un MySql datubāzes pārvaldības sistēmu.

Kvalifikācijas darbā ir ievads, uzdevūma nostādne, prasību specifikācija, līdzekļu izēle ar pamatojumu, modelēšana un projektēšana, datu struktūra ar aprakstu, lietotāja jeb klienta ceļvedi. Tīmekļa vietnes izstrāde sākās ar analīzi, lai saprastu, kas vēl ir nepieciešams lietotājiem, tad modelēšanu un projektēšanu kopā ar ER modeļiem un datubāzes izveidi.

**ANNOTATION**

The qualification work is a financial system website. The system enables the automatic creation of invoices, the long-term and secure electronic storage of invoices on the eurocipari.lv server, the notification of invoices to customers and the possibility to pay invoices to the company's customers. The system was developed using the PHP programming language, the LARAVEL framework and the MySql database management system.

The qualification work consists of an introduction, a problem statement, a requirements specification, a means statement with justification, modelling and design, a data structure with description, a user or client guide. The development of the website started with an analysis to understand what else users need, then modelling and designing together with ER models and database creation.

**SATURS**

[**IEVADS** 5](#_Toc168869729)

[**1.** **UZDEVUMA NOSTĀDNE** 6](#_Toc168869730)

[**2.** **PRASĪBU SPECIFIKĀCIJA** 7](#_Toc168869731)

[**2.1.** **Ieejas un izejas informācijas apraksts** 7](#_Toc168869732)

[2.1.1. Ieejas informācijas apraksts 7](#_Toc168869733)

[2.1.2. Izejas informācijas apraksts 8](#_Toc168869734)

[**2.2.** **Funkcionālās prasības** 9](#_Toc168869738)

[**2.3.** **Nefunkcionālās prasības** 9](#_Toc168869739)

[**3.** **UZDEVUMA RISINĀŠANAS LĪDZEKĻU IZVĒLES PAMATOJUMS** 13](#_Toc168869740)

[**4.** **PROGRAMMATŪRAS PRODUKTA MODELĒŠANA UN PROJEKTĒŠANA** 14](#_Toc168869741)

[**4.1.** **Sistēmas struktūras modelis** 14](#_Toc168869742)

[4.1.1. Sistēmas arhitektūra 14](#_Toc168869743)

[4.1.2.Sistēmas ER modelis 15](#_Toc168869749)

[**4.2.** **Funkcionālais sistēmas modelis** 17](#_Toc168869753)

[4.2.1.Datu plūsmu modelis 17](#_Toc168869754)

[**5.** **DATU STRUKTŪRAS APRAKSTS** 19](#_Toc168869755)

[**6.** **LIETOTĀJA CEĻVEDIS** 24](#_Toc168869756)

[**6.1.** **Sistēmas prasības aparatūrai un programmatūrai** 24](#_Toc168869757)

[**6.2. Sistēmas instalācija un palaišana** 24](#_Toc168869758)

[**6.3. Programmas apraksts** 26](#_Toc168869759)

[**6.4. Testa piemērs** 36](#_Toc168869760)

[**NOBEIGUMS** 38](#_Toc168869761)

[**INFORMĀCIJAS AVOTI** 39](#_Toc168869762)

[PIELIKUMI 40](#_Toc168869763)

[1.pielikums. b2b sistēmas shēma 41](#_Toc168869764)

[2.pielikums. Pieteikšanās shēma, autorizācija 42](#_Toc168869765)

[3.pielikums. Jautājumu shēma 43](#_Toc168869766)

[4.pielikums. Uzņēmumu shēma 44](#_Toc168869767)

[5.pielikums. Lietotāju shēma 45](#_Toc168869768)

[6.pielikums. Ziņu shēma 46](#_Toc168869769)

[7.pielikums. Maksājumu shēma 47](#_Toc168869770)

[8.pielikums. Fiziskās struktūra 48](#_Toc168869771)

[9.pielikums. API-Postman 49](#_Toc168869772)

[10.pielikums. Programmas pirmteksts 50](#_Toc168869773)

# **IEVADS**

**Finanšu sistēmas tīmekļa vietne:**

Mūsdienās finanšu sistēmas ir kļuvušas par svarīgu rīku produktivitātes un efektivitātes uzlabošanai. Šādas sistēmas ietaupa laiku un resursus, pārbaudot un analizējot uzņēmuma darbību.

Šī kvalifikācijas darba tēma ir Finanšu sistēmas tīmekļa vietne. Šādas sistēmas aktualitāti nosaka pieaugošais pieprasījums pēc ērtām un mūsdienīgām funkcijām, jo viena vietnē var gan apmaksāt uzņēmuma rēķinus, gan apskatīt visu maksājumu vēsturi.

**Mērķauditorija:**

Finanšu sistēmas tīmekļa vietne ir paredzēta plašam lietotāju lokam, tostarp uzņēmumiem, privātpersonām un grāmatvežiem, lai ērti veikt, sniegt un apskatīt maksājumus.

**Sistēmas priekšrocības:**

Ir iespēja apmaksāt rēķinus, jo ne visiem konkurentiem vietnē ir norēķinu sistēma, strādājot ar sistēmai pieslēgtu uzņēmumu klientiem. Salīdzinājumā ar konkurentu “mansjums” ir mans konkurents.

**Automatizācija:**

Sistēmas lietotājiem nav nepieciešamas specializētas zināšanas vai prasmes, lai darbotos ar sistēmu. Sistēma ir viegli lietojama, intuitīva, skaidra un efektīva, ietaupot lietotāju laiku un resursus. Šīs sistēmas izmantošana ļauj lietotājiem gūt virkni priekšrocību, tostarp palielināt produktivitāti un efektivitāti, samazināt kļūdu risku, palielināt pārskatu precizitāti. Pilna automatizācija šobrīd nav pieejama, sakarā ar to, ka nav iespējams pieslēgt sistēmu reālam uzņēmumam, to nevar izdarīt un neviens to neļaus, administrators ģenerē rēķinus manuāli, kas jau atvieglo grāmatveža darbu (skat. 1.pielikums b2b sistēmas shēma).

**Secinājumi:**

Tīmekļa vietne ir moderna un efektīva sistēma, kas var sniegt ievērojamus ieguvumus uzņēmumiem un privātpersonām. Šāda sistēma var atvieglot dāvināšanas procesus, ātri un viegli veikt maksājumus.

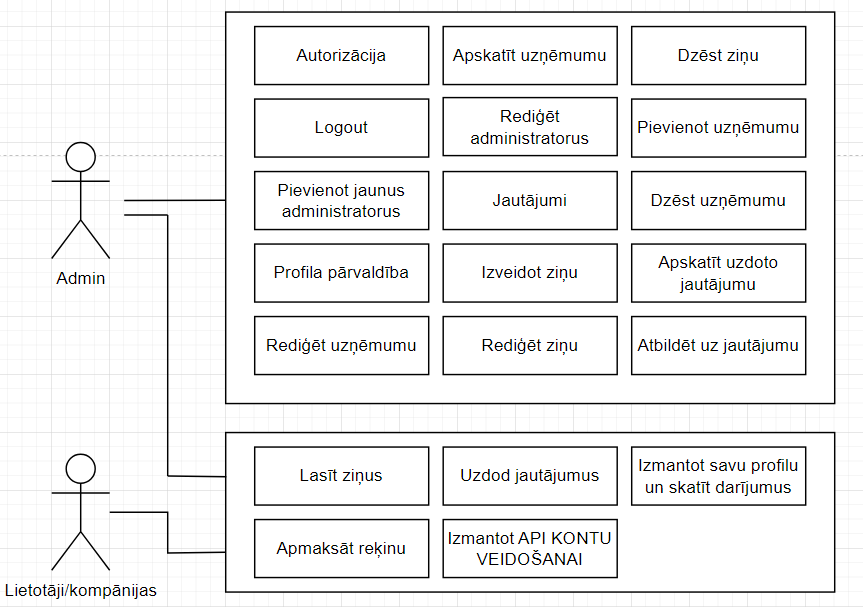
Sistēma ir B2B (business-to-business) risinājums (skat. 1.pielikums 1.att b2b sistēmas shēma).

# **UZDEVUMA NOSTĀDNE**

Tēmas plašais skaidrojums - tīmekļa vietne ir aktuāla un nepieciešama sistēma, jo tā piedāvā vairākas priekšrocības salīdzinājumā ar esošajiem analogiem un var sniegt lietotājiem ievērojamus ieguvumus, vienkāršību. Finanšu sistēmas tīmekļa vietne ir aktuāla un nepieciešama sistēma, jo tā piedāvā vairākas priekšrocības salīdzinājumā ar esošajiem risinājumiem un var sniegt lietotājiem ievērojamus ieguvumus, vienkāršību un efektivitāti. Salīdzinot, mana sistēma ir “mansjumis” konkurents.

Ir plānotas vairākas funkcijas:

* Maksājuma parvāldība.
* Administratora pārvaldībā.
* Kompanijas pārvaldībā.
* Ziņas pārvaldībā.
* Jautājumi prieķš sistēmas īpašniekiem un atgriezeniskā saite.
* Iespēja saglabāt rēķina pdf.

****

1.att. Lietojumgadījuma diagramma.

# **PRASĪBU SPECIFIKĀCIJA**

## **Ieejas un izejas informācijas apraksts**

### ***Ieejas informācijas apraksts***

Sistēmā tiks nodrošināta šādas ieejas informācijas apstrāde:

1. Informācija par **lietotājiem/administratoriem** sastāvēs no šādiem datiem.

* Vārds – lietotāja/administrator vārds – burtu teksts ar izmēru līdz 255 rakstzīmēm (piem., “Valentīns”).
* E-pasts – lietotāja/administratora e-pasts – teksts ar izmēru līdz 255 rakstzīmēm (piem., “testemail@gmail.com”).
* Parole – lietotāja/administratora konta parole – teksts ar izmēru līdz 30 rakstzīmēm (piem., “sr567mj54aszre”).

1. Informācija par **kompanijam** sastāvēs no šādiem datiem.

* Nosaukums – kompanijas nosaukums – burtu teksts ar izmēru līdz 255 rakstzīmēm (piem., “Sportland”).
* Reg. numurs - Kompanijas reģistrācijas numurs – (piem., “3946568200654”).
* VAT numurs - PVN numurs – (piem., “21”).
* Apraksts – apraksts par kompaniju – (piem., “uzņēmuma lokācija Rīga”).
* Adrese – kompanijas atrašanās vietā – burtu teksts ar izmēru līdz 255 rakstzīmēm (piem., “K.Barona 115”).
* E-pasts – kompanijas e-pasts – teksts ar izmēru līdz 255 rakstzīmēm (piem., “testemail@gmail.com”).
* Tālrunis - kompanijas tālrunis – teksts ar izmēru līdz 20 rakstzīmēm (piem., “+371 25682251”).
* Api kods – kods priekš sistēmai jeb unikālais kods, ar kuru var uzzināt sikāk detāļus - teksts ar izmēru līdz 255 rakstzīmēm (piem., “3242gf234234k67”).

1. Informācija par **jautājumiem** sastāvēs no šādiem datiem.

* Teksts – jautājuma teksts jeb jautājuma saturs – burtu teksts līdz 255 rakstzīmēm (piem., “Cik maksāt par lietošanu ?”).
* E-pasts – lauks e-pasts, lai redzēt, kur var sutīt atbildes – teksts ar izmēru līdz 255 rakstzīmēm (piem., “testemail@gmail.com”).

1. Informācija par **ziņam** sastāvēs no šādiem datiem.

* Nosaukums – ziņojuma nosakums – teksts ar izmēru līdz 255 rakstzīmēm (piem., “testemail@gmail.com”).
* Ziņojuma apakšnosaukums – ziņojumu otrais nosaukums - teksts ar izmēru līdz 255 rakstzīmēm (piem., “Kvīts”).
* Teksts – ziņojuma saturs – teksts ar izmēru līdz 255 rakstzīmēm (piem., “Šodien ir atlaides”).

1. Informācija par **rēķiniem** sastāvēs no šādiem datiem.

* Procentu likme – decimal(20,2) formats.
* Sniedzējs\_uzņēmuma\_nosaukums – teksts ar izmēru līdz 255.
* Sniedzējs\_uzņēmuma\_epasts – teksts ar izmēru līdz 255.
* Sniedzējs\_uzņēmuma\_tālrunis – teksts ar izmēru līdz 255.
* Sniedzējs\_PVN\_numurs – teksts ar izmēru līdz 255.
* Sniedzējs\_uzņēmuma\_reg. numurs – teksts ar izmēru līdz 255.
* Sniedzējs\_iban\_numurs – konta numurs - teksts ar izmēru līdz 255.
* Sniedzējs\_swift – bankas kods - teksts ar izmēru līdz 255.
* Klients\_vārds – klienta vārds - teksts ar izmēru līdz 255.
* Klients\_epasts – klienta e-pasts - teksts ar izmēru līdz 255.
* Klients\_tālrunis – klienta tālrunis - teksts ar izmēru līdz 255.
* Klients\_uzņēmuma\_nosaukums – klienta uzņēmums - teksts ar izmēru līdz 255.
* Klients\_iban – konta numurs – teksts ar izmēru līdz 255.
* Klients\_swift – bankas kods – teksts ar izmēru līdz 255.

### ***Izejas informācijas apraksts***

**PDF Kvīts.** PDF nodrošina iespēju pieladēt kvītu par maksājumu.

**Kļūdas ziņojumi.** Viņi dod zīņus jeb bridinājumus, ka kaut kas darām nepareizi (piem., nerapareizi ievadam paroli).

**Jautājumi no administatora dashboard’a.** Administratoriem ir iespēja redzēt uzdotus jautājumus, sistēma izvadi viņus.

**Precīzas uzņēmumu un maksājumu uzskaites attēlošana, un izvadīšana.** Vietnes galvenajā ekrānā tiek rādīta automātiska reģistrēto uzņēmumu un veikto maksājumu uzskaite.



## **Funkcionālās prasības**

1. Lietotāja autentifikācija:

Lietotājam ir jāspēj autentificēties sistēmā, izmantojot e-pasta adresi un parole.

1. Darījumu veikšana:

Lietotājam ir jāspēj veikt maksājumus sistēmā.

1. Administrācijas funkcijas:

Administratoriem ir jāspēj administrēt sistēmu, viņš visu kontrole.

1. Uzņēmums un tā pārstāvji:

Uzņēmumi var izmantot tikai API un skatīt savus kontus.

1. Jautājumi:

Lietotāji uzdod jautājumus, bet administratori tos pārvalda un atbild uz tiem.

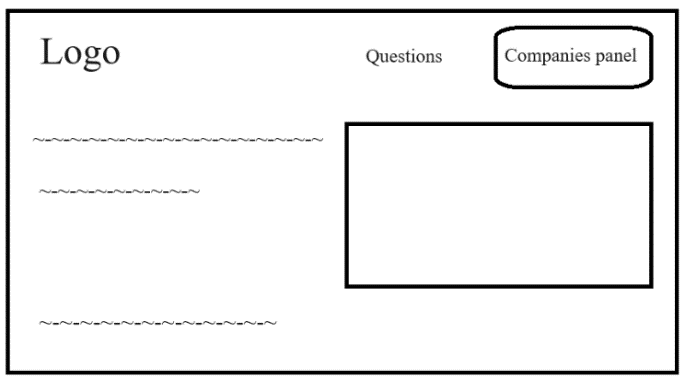
1. Ziņu sadaļas:

Pēc tehniskiem darbiem ar vietni un sistēmu, kā arī ja ir kāda svarīga informācija, administrators raksta ziņu, un tā tiek parādīta galvenajā lapā. Ierakstus pārvalda administrators.

## **Nefunkcionālās prasības**

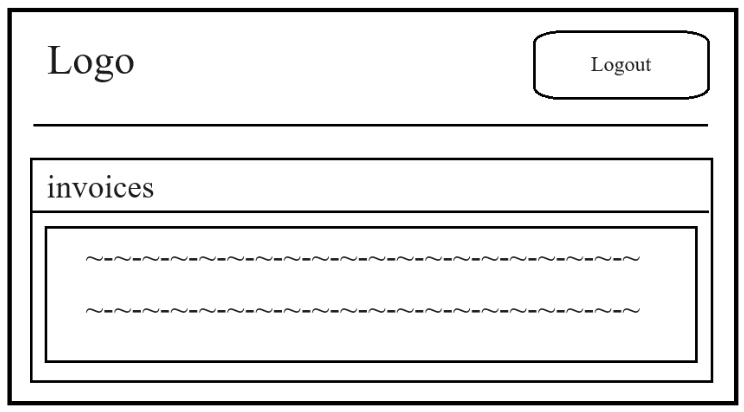
1. Sistēmai jāspēj izpildīt lietotāja pieprasījumus saprātīgā laika posmā.
2. Sistēmai jāspēj izturēt noteiktu lietotāju skaitu.
3. Sistēmai jābūt viegli lietojamai un saprotamai.
4. Sistēnai jābūt pievilcīgai vizuāli.
5. Sistēnai jābūt aizsargātai pret nesankcionētu piekļuvi, izmantošanu vai atklāšanu.
6. Sistēnai jābūt aizsargātai pret datu zādzību vai iznīcināšanu.
7. Sistēnai jābūt pieejamai un funkcionējošai.
8. Sistēnai jābūt izturīgai pret kļūdām.

Skīces:

Uz galvenās lapas sīktēla var redzēt teksta informācijas daļu, pogu, lai ieietu panelī jautājumu nosūtīšanai un ieiešanai uzņēmuma panelī (skat. 2.att.galvenā lapa).

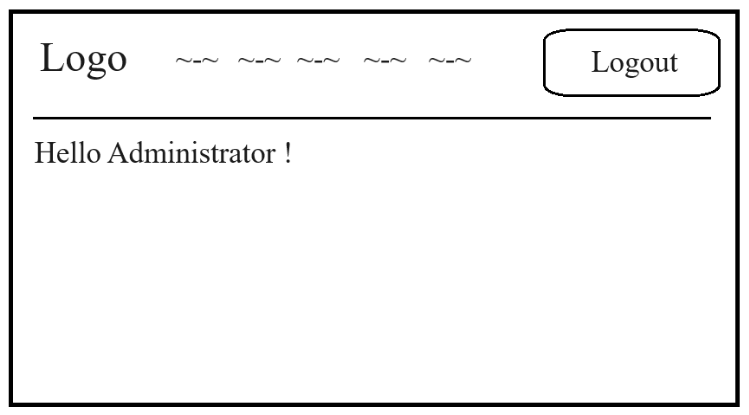
2.att. galvenā lapa.

Uzņēmuma panelī varat redzēt uzņēmuma maksājumu rēķinus un pogu, lai izietu no sava konta (skat. 3.att.uzņēmuma panelis).



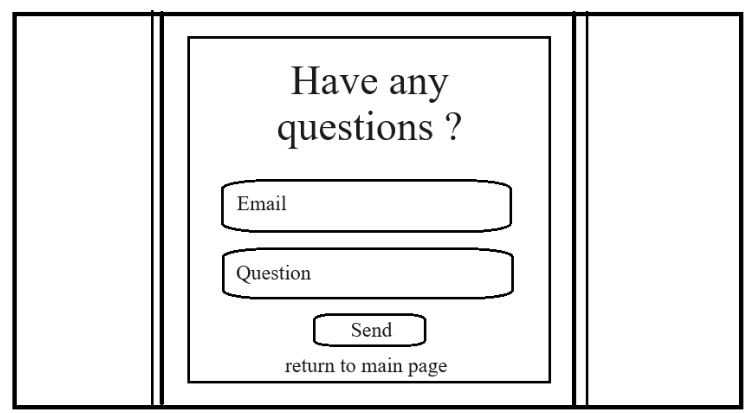
3.att. uzņēmuma panelis.

Admin paneļa administratori var redzēt savas sistēmas administrēšanas funkcijas un pogu, lai izietu no sava konta (skat. 4.att.admin panelis).



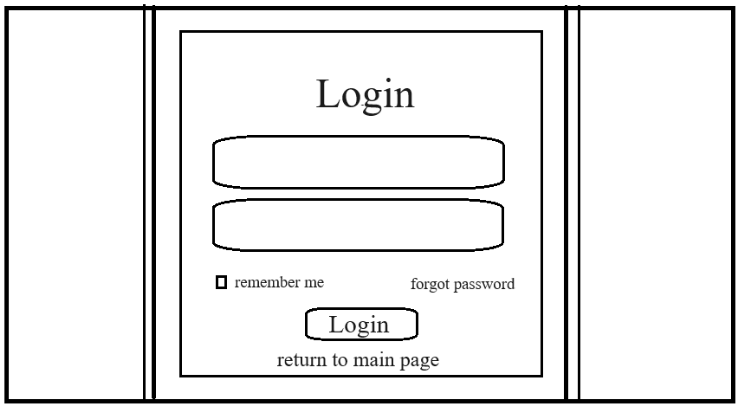
4.att. admin panelis.

Jautājumu lapā būs e-pasts, no kura tiks nosūtīts jautājums, lauks pašam jautājumam, poga jautājuma nosūtīšanai un poga, lai atgrieztos galvenajā lapā (skat. 5.att.jautājumi).

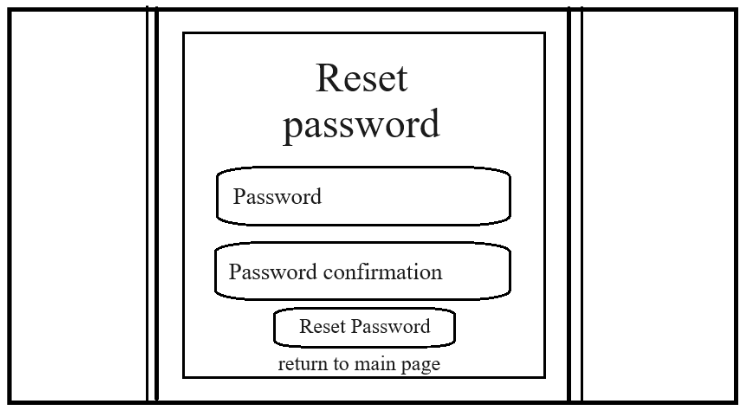


5.att. jautājumi.

Uzņēmumu un administratoru pieteikšanās izvēlnē būs divi lauki, ar kuriem jūs varat pieteikties savā kontā, noklikšķinot uz pieteikšanās pogu, pirmais lauks ir epasts, otrais lauks ir parole (skat. 6.att.admin-user login).

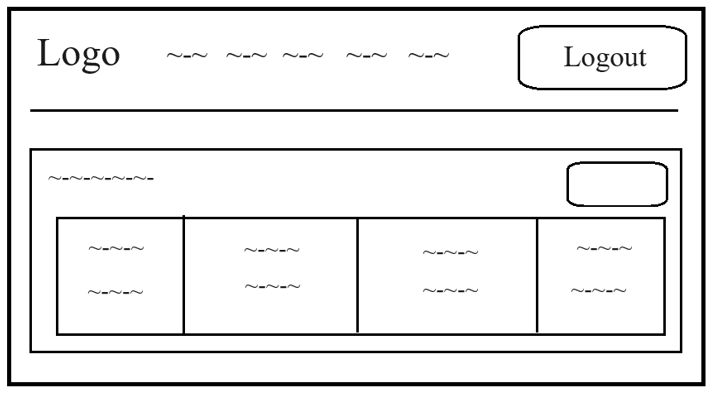


6.att. admin-user login.

Paroles atiestatīšanas panelis sastāvēs no diviem laukiem, kas būs pieejami pēc noklikšķināšanas uz paroles atiestatīšanas saites e-pastā. Lai atiestatītu paroli, būs jāievada jauna parole un vēlreiz jāievada jaunā parole lai apstiprināt. (skat. 7.att.paroles atjaunošana).

7.att. paroles atjaunošana.

Šeit var redzēt funkciju veidni, kā izskatās visas administratora funkcijas, kad viņš kādu no tām ieslēdz - uzņēmumi, administratori, ziņojumi, jautājumi, maksājumi (skat. 8.att. uzņēmumi, administratori, ziņojumi, jautājumi, maksājumi).



8.att. admin - uzņēmumi, administratori, ziņojumi, jautājumi, maksājumi.

# **UZDEVUMA RISINĀŠANAS LĪDZEKĻU IZVĒLES PAMATOJUMS**

Lai risinātu uzdevumu, jums būs nepieciešams izvēlēties atbilstošus līdzekļus, kas palīdzēs izveidot plānoto sistēmu, ietverot Lietojumgadījuma modeli un citus prasību specifikācijas elementus. Izvēloties līdzekļus, ir svarīgi ņemt vērā vairākus faktorus.

**Analīzes un projektēšanas rīki:**

Draw io un dbdiagram.io: Rīki piedāvā diagrammu izstrādei un sadarbībai tiešsaistē.

Datu bāzes pārvaldības sistēma (DBMS): MySQL 8.0: Datu bāzes pārvaldībai un uzglabāšanai.

**Programmēšanas valoda:**

PHP 8.1, HTML 5, CSS, JS: Plaši izmantotas valodas, kas piedāvā daudzas bibliotēkas un rīkus web lapas izstrādei un rakstu es visu caur framework LARAVEL 10, jo LARAVEL ir bezmaksas PHP ietvars, kas paredzēts tīmekļa lietojumprogrammu izveidei. Tas nodrošina izstrādātājus ar rīku un bibliotēku kopu, kas atvieglo un paātrina sarežģītu tīmekļa lietojumprogrammu izstrādi.

**Vide izstrādei un testēšanai:**

PHPstorm 2022.3.3: Kodēšanai, jo viņš tas ietver atkļūdošanu un testēšanu, lietojamību, sistēmu atbalstu, integrāciju ar sistēmām.

**Versiju kontroles sistēma:**

Git: Nepieciešams, lai efektīvi pārvaldītu izmaiņas kodā un sadarbību vairāku izstrādātāju vidū.

**Dokumentēšanas un prezentācijas rīki:**

Microsoft Word 2016: Dokumentēšanai, prasību specifikācijas veidošanai.

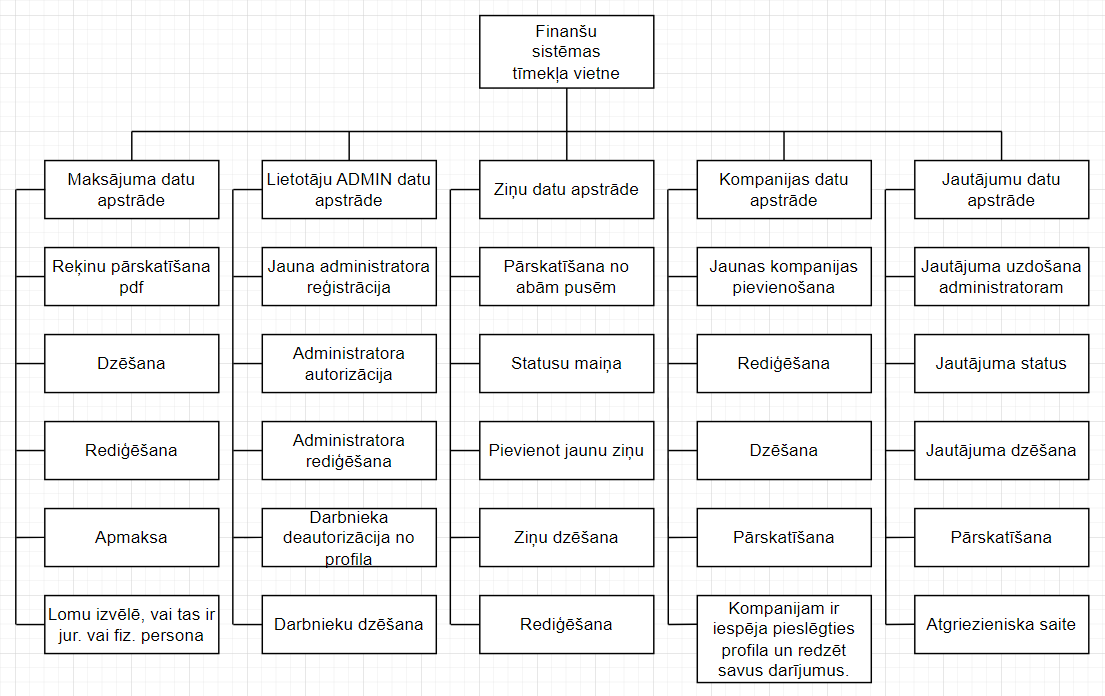
Microsoft powerpoint 2016: Prezentācijas veidošanai.

Pamatojoties uz sistēmas prasībām es izvēlējos līdzekļus, kas vislabāk atbilst projekta prasmēm. Katrs no šiem rīkiem un tehnoloģijām ir jāpielāgo konkrētajām prasībām un uzdevuma apjomam. Lai pareizi izvēlēties rīkus, es detalizēti izvērtēju katru iespējamo līdzekļu un paņēmu vērā to priekšrocības un ierobežojumus.

# **PROGRAMMATŪRAS PRODUKTA MODELĒŠANA UN PROJEKTĒŠANA**

## **Sistēmas struktūras modelis**

### ***Sistēmas arhitektūra***

Sistēma būs 5 apakšsistēmās (skat. 9.att. funkcionālās dekompozīcijas diagramma) un katra no šīm apakšsistēmām atbild par konkrētu funkcionalitāti.

9.att. Funkcionālās dekompozīcijas diagramma

**Maksājuma datu apstrādes apakšsistēma** - ir atbildīga par visiem maksājuma datiem un to pārvaldību.

**Lietotāju/ADMIN datu apstrādes apakšsistēma** - ir atbildīga par visu stradnieku informāciju apstrādi un uzturēšanu.

**Ziņas datu apstrādes apakšsistēma** - ir atbildīga par visu zinu informācijas apstrādi un pārvaldīšanu. Tas ietver visu, ko var izdarīts ar zinu.

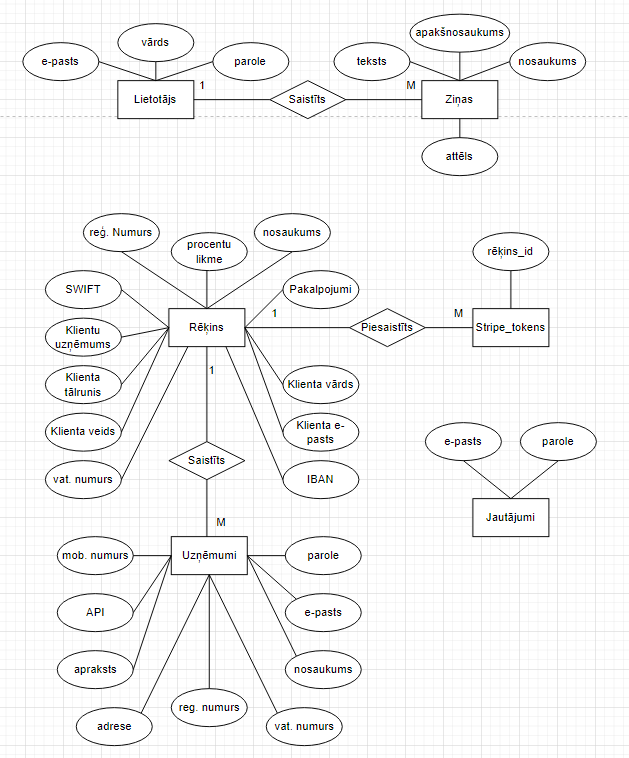
**Kompanijas datu apstrādes apakšsistēma** - ir atbildīga par visam kompanijam un informācijas apstrādi un pārvaldīšanu.

**Jautājumu datu apstrādes apakšsistēma** - ir atbildīga jautājumu apstrādi un pārvaldīšanu. Tas ietver visu, ko saistīts ar klienta jautājumiem un kas atsutīts administratoriem.



### ***Sistēmas ER modelis***

Sistēmas ER-modelis sastāv no 6 entitijām (skat. 10.att. ER diagram), kas nodrošina pamatinformācijas uzglabāšanu un apstrādi. Tas ir Uzņēmums, Ziņas, Lietotājs, Rēķins, Jautājumi, Stripe\_tokens.



10.att. ER diagram

**Uzņēmumi:**

Kompānijas glabā informāciju par uzņēmumiem, kas izmanto sistēmu. Katram uzņēmumam ir unikāls API, mob. numurs, parole, apraksts, adrese, reg. numurs, vat. numurs, e-pasts un nosaukums.

**Ziņas:**

Ziņas glabā informāciju par ziņām, kas tiek publicētas sistēmā. Katrai ziņai ir ziņas saturs, nosaukums, attēls un apakšnosaukums.

**Lietotājs:**

Lietotājs glabā informāciju par sistēmas lietotājiem. Katram ir unikāls vārds, e-pasts, parole.

**Jautājumi:**

Jautājumi glabā informāciju par atsutītam jautājumiem. Katram jautājumam ir e-pasts, no kura bija atsutīts jautājums un jautājuma saturs.

**Rēķins:**

Glabā informāciju par darījumiem/rēķiniem. Katram iban, e-pasts, klienta vārds, pakalpojumi, nosaukums, procentu likme, reg. numurs, swift, klientu uzņēmums, klienta tālrunis, klienta veids, vat. numurs.

**Stripe\_tokens:**

Glabā informāciju par rēķinu\_id, jo tas ir saišū tabula priekš maksājumiem.



## **Funkcionālais sistēmas modelis**

### ***Datu plūsmu modelis***

Autorizācijas datu plūsmas diagramma (skat. 2.pielikums 3.att. Autorizācijas datu plūsmu diagramma) parāda, kā lietotājs tiek autorizēts piekļūt sistēmai. Diagramma sākas ar to, ka lietotājs ievada savu e-pasta adresi un paroli. Šie dati tiek nosūtīti uz sistēmu, kur tie tiek pārbaudīti. Ja e-pasts jau pastāv sistēmā un parole ir pareiza, lietotājam tiek piešķirta autorizācija. Ja e-pasts neeksistē vai parole ir nepareiza, lietotājam tiek nosūtīts paziņojums par kļūdu.

Jautājumu datu plūsmu diagramma (skat. 3.pielikums 3.att. Jautājumu datu plūsmu diagramma) parāda, ja epasts ir ievadīts pareizi, var ievadīt pašu jautājumu, ja pasts ir ievadīts nepareizi, tiks parādīta kļūda, ka pasta formāts ir nepareizs. Ja pasts ir ievadīts pareizi, var ievadīt pašu jautājumu un ja jautājumā nav norādīts nekas, tiks parādīta kļūda, ka lauks ir tukšs, tas būs jāaizpilda. Ja viss ir pareizi, datubāze saglabās jautājumu un viņu nosūtīs administratoram.

Uzņēmumu datu plūsmu diagramma (skat. 4.pielikums 4.att. Uzņēmumu datu plūsmu diagramma) parāda jauna uzņēmuma pievienošanu. Ievadiet nosaukumu, reģistrācijas numuru, vat numuru, e-pastu, paroli, tālruni, piešķiriet API kodu, ar kuru uzņēmums varēs apskatīt savus maksājumus, pēc vēlēšanās tiek pievienots uzņēmuma apraksts. Ja viss ir ievadīts pareizi, uzņēmums tiks izveidots, pievienots datu bāzei un pievienošana būs veiksmīga citos gadījumos, tiks parādīti bridinājumi, kas norāda, ka dati kaut kadi ievadīti nepareizi.

Lietotāju datu plūsmu diagramma (skat. 5.pielikums 5.att. Lietotāju datu plūsmu diagramma) parāda jauna lietotāja, jeb administratora pievienošanas procesu. Ievadīt ir nepieciešams vārdu, paroli un e-pasta adresi. Sistēma pārbauda ievadīto informāciju un, ja viss ir pareizi, izveido jaunu administratoru. Ja dati ir nepareizi, tiek nosūtīts paziņojums par kļūdu.

Ziņu datu plūsmu diagramma (skat. 6.pielikums 6.att. Ziņu datu plūsmu diagramma) parāda, kā izveidot jaunu ziņu. veidojot ierakstu, tiek ievadīts virsraksts, tiek pārbaudīts vai lauks ir tukšs, pēc tam tiek ievadīts otrs nosaukums, tiek veikta tā pati pārbaude, lai pārbaudītu, vai lauks ir tukšs vai nav, tiek ievietota bilde uz tā arī ir validācija, lai pārbaudītu tukšo lauku, tad saturs tiek uzrakstīts, saturs nevar būt tukšs, pārbaude turpinās, ziņas statuss, statuss var būt aktīvs vai neaktīvs, ja dati ir visi ievadīti un visi lauki ir aizpildīti pēc tam tiek veikta saglabāšana datu bāzē un ieraksts ir veiksmīgi izveidots ja kāds lauks paliek tukšs tad tiek parādīts logs ar paziņojumu, ka lauks ir jāaizpilda.

Maksājumu datu plūsmu diagramma (skat. 7.pielikums 7.att. Maksājumu datu plūsmu diagramma) parāda jauna konta pievienošanu. Uzņēmuma pusē tas tiek darīts, izmantojot API kodu, ja konts tiek pievienots no administratora puses, administrators izvēlas uzņēmumu, kuram jāpievieno rēķinu, pēc tam ievada procentu likmi, klienta veidu - juridiska vai fiziska persona, klienta vārdu, e-pastu, tālruņa numuru, produkta vai pakalpojuma nosaukumu un maksājamo cenu. Pēc tam tiek pārbaudīts, vai tajos ir dati, lauki nav tukši un laukiem ir noteikts rakstzīmju skaits. Ja kaut kas ir ievadīts nepareizi, tiek parādīts kļūdas paziņojums ar lūgumu aizpildīt attiecīgo lauku vai ka minimālajam rakstzīmju skaitam jābūt vienādam ar noteiktu skaitu. Ja viss ir ievadīts pareizi, rēķins tiek saglabāts datubāzē, pievienots uzņēmuma profilam un tiek uzskatīts par veiksmīgi izveidotu.

# **DATU STRUKTŪRAS APRAKSTS**

Datu bāzes projektēšanas rezultātā tika veidota vairākas tabulas un starp tām tika definēta tabulu relācija jeb saistības (skat. 7.pielikums 7.att. Datu bāzes fiziskās struktūras diagramma).

Visi dati tiks saglabāti MySQL datu bāzē. Tajā tiks glabāti dati par lietotājiem, sludinājumiem, lomām un citi dati. Datu Bāze sastāv no 13 tābulam.

Tabulas datu bāzē:

* Tabula "stripe\_token” – saišū tabula priekšā maksājumus sistēmai.
* Tabula “companies” - tabulā glabas dati par uzņēmumiem.
* Tabula “invoices” - tabulā glabas dati maksājumim un darījumiem.
* Tabula “questions” - tabulā glabas dati par jautājumiem.
* Tabula “posts” - tabulā glabas dati ziņam.
* Tabula “users” - tabulā glabas dati par jautājumiem.

Pārējie ir palīgtabulas:

* migrations.
* companies\_password\_reset\_tokens.
* invoice\_services.
* password\_reset.
* password\_reset\_tokens.
* failed\_jobs.

Tabula "**uzņēmumi**" ir informācija par uzņēmumiem.

1. tabula uzņēmumi

Tabulas “**companies**” sturktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nr.** | **Nosaukums** | **Tips** | **Izmērs** | **Apraksts** |
| 1. | id | bigint | - | Unikālāis kods |
| 2. | name | varchar | 255 | Kompanijas nosaukums |
| 3. | reg.number | varchar | 255 | Kompanijas reģistrācijas numurs |
| 4. | vat.number | varchar | 255 | PVN numurs |
| 5. | address | text | - | Kompanijas atrašanās vietā |
| 6. | description | text | - | Apraksts par kompaniju |
| 7. | email | varchar | 255 | Kompanijas e-pasts |
| 8. | phone | varchar | 255 | Kompanijas tālrunis |
| 9. | api.code | varchar | 255 | Kods priekšā uzņēmuma pieslēgšānai |
| 10. | password | varchar | 255 | Uzņēmuma parole, lai pieslegties profila |
| 11. | created\_at | timestamp | - | Kad tiek uztaisīts |
| 12. | updated\_at | timestamp | - | Kad tiek atjaunināts |

Tabula "**jautājumi**" ir informācija par jautājumiem.

2.tabula jautājumi

Tabulas “**questions**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nr.** | **Nosaukums** | **Tips** | **Izmērs** | **Apraksts** |
| 1. | id | bigint | - | Unikālāis kods, no kura atsutīts jautājums |
| 2. | email | varchar | 255 | E-pasts priekš jautājumam |
| 3. | question\_text | text | - | Teksts, pats jautājums |
| 4. | ip | varchar | 255 | Lokācija no kuriemes bija atsutīts jautājums |
| 5. | verified | tinyint | 1 | Aktīvs vai jau aizverts jautājums |
| 6. | created\_at | timestamp | - | Kad tiek uzrakstīts |
| 7. | updated\_at | timestamp | - | Kad tiek atjaunināts |

Tabulā “**ziņas**” ir informācija par ziņam:

3.tabula ziņas

Tabulas “**posts**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nr.** | **Nosaukums** | **Tips** | **Izmērs** | **Apraksts** |
| 1. | id | bigint | 255 | Unikālāis kods |
| 2. | title | varchar | 255 | Nosaukums |
| 4. | slug | varchar | 255 | Sludinājuma apkšnosaukums |
| 5. | text | longtext | - | Saturs |
| 6. | active | tinyint | 1 | Ziņa pieejamība statuss |
| 7. | user\_id | bigint | - | Atsauce uz user tābulu |
| 8. | created\_at | timestamp | - | Kad tiek uzrakstīts |
| 9. | updated\_at | timestamp | - | Kad tiek atjaunināts |
| 10. | image | varchar | 255 | Attēls prikš ziņai |

Tabulā “**lietotāji**” ir informācija par lietotājiem:

4.tabula lietotāji

Tabulas “**users**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nr.** | **Nosaukums** | **Tips** | **Izmērs** | **Apraksts** |
| 1. | id | bigint | - | Unikālāis kods |
| 2. | name | varchar | 255 | Lietotāja vārbs |
| 3. | email | varchar | 255 | Lietotāja e-pasts |
| 4. | email\_verified\_at | timestamp | - | - |
| 5. | password | varchar | 255 | Lietotāja parole |
| 6. | Remember\_token | varchar | 100 | - |
| 7. | created\_at | timestamp | - | Kad tiek uzrakstīts |
| 8. | updated\_at | timestamp | - | Kad tiek atjaunināts |

Tabulā “**rēķini**” ir informācija par rēķiniem:

5.tabula rēķini

Tabulas “**rēķini**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nr.** | **Nosaukums** | **Tips** | **Izmērs** | **Apraksts** |
| 1. | id | bigint | - | Unikālāis kods |
| 2. | reference | varchar | 255 | Konta dati |
| 3. | company\_id | bigint | - | Atsaucē uz tābulu company |
| 4. | tax\_rate | decimal | 20,2 | Procentu likme |
| 5. | provider\_name | varchar | 255 | Vārds |
| 6. | provider\_email | varchar | 255 | E-pasts |
| 7. | provider\_phone | varchar | 255 | Tālrunis |
| 8. | provider\_company | varchar | 255 | Uzņēmums |
| 9. | provider\_vat\_number | varchar | 255 | Pilsētas unikālais numurs |
| 10. | provider\_reg\_number | varchar | 255 | Reģistrācijas numurs |
| 11. | provider\_iban | varchar | 255 | Konta numurs |
| 12. | provider\_swift | varchar | 255 | Bankas kods |
| 13. | client\_type | int | 255 | Klientu tips |
| 14. | client\_name | varchar | 255 | Vārds |
| 15. | client\_email | varchar | 255 | E-pasts |
| 16. | client\_phone | varchar | 255 | Tālrunis |
| 17. | client\_company | varchar | 255 | klienta uzņēmuma nosaukums. |
| 18. | client\_vat\_number | varchar | 255 | Klientam nav vat (PVN) |
| 19. | client\_reg\_number | varchar | 255 | Reģistrācijas numurs |
| 20. | client\_iban | varchar | 255 | Konta numurs |
| 21. | client\_swift | varchar | 255 | Bankas kods |
| 22. | paid | tinyint | 1 | Apmaksā |
| 23. | transaction\_code | varchar | 255 | Tranzakcijas numurs |
| 24. | created\_at | timestamp | - | Kad tiek atjaunināts |
| 25. | updated\_at | timestamp | - | Kad tiek atjaunināts |

# **LIETOTĀJA CEĻVEDIS**

## **Sistēmas prasības aparatūrai un programmatūrai**

Sistēma tika realizēta tā, lai tā būtu maksimāli elastīga, atbilstoša specifisku vajadzību nodrošināšanai un viena no svarīgākajām sistēmas prasībām ir interneta pieslēgums. Stabils interneta savienojums nozīme - jo labāks ir savienojums, jo labāk darbojas programma. jo man ir tīmekļa vietne. Vietni var palaist no jebkuras pārlūkprogrammas:

* Safari.
* Google Chrome.
* Windows Edge.
* Opera.
* Mozilla Firefox.

## **6.2. Sistēmas instalācija un palaišana**

**PHP, LARAVEL un Composer instalēšana:**

Pirms LARAVEL instalēšanas datorā ir jāinstalē PHP un Composer. Darbs tika rakstīts, izmantojot operētājsistēmu Windows, un, lai instalētu Composer ar PHP operētājsistēmā Windows, jādodas uz oficiālo PHP vietni (<https://www.php.net/downloads.php>) un lejupielādēt instalācijas failu, tas pats ar Composer, lejupielādējot instalācijas failu no oficiālā PHP vietne (<https://getcomposer.org/download/>) vai instalējot no komandrindas (instalēšanas komandas ņemt no <https://getcomposer.org/download/>). LARAVEL instalēt no komandrindas, izmantojot komandas no oficiālā vietne (<https://laravel.com/docs/4.2>).

**LARAVEL projekta izveide:**

**Finanšu sistēma.**

Sistēmas prasības:

PHP 8.0 (vai jaunāka versija).

Instalējiet šādus PHP paplašinājumus: curl, fileinfo, gd, gettext, intl, mbstring, openssl, pdo\_mysql, pdo\_sqlite.

**Uzstādīšana:**

Klonējiet šo repozitoriju mērķa mapē.

Git clone https://github.com/pikcrvt-prakse/dvb-2024-Valik63.git.

Izveidojiet datu bāzi ar nosaukumu “praktika” (izmantojiet savu MySQL serveri).

Ja ir iestatījumi e-pasta sūtīšanai, varat ar tiem atjaunināt .env faila sadaļu PASTS. Šajā gadījumā .env sadaļai PASTS ir jāizskatās šādi:

*MAIL\_MAILER=smtp*

*MAIL\_HOST=sandbox.smtp.mailtrap.io*

*MAIL\_PORT=2525*

*MAIL\_USERNAME=ffb544fec653c6*

*MAIL\_PASSWORD=5b481cf83efe40*

*MAIL\_ENCRYPTION=null*

*MAIL\_FROM\_ADDRESS="hello@example.com"*

*MAIL\_FROM\_NAME="${APP\_NAME}"*

Pēc tam instalējiet vietņu atkarības, izpildot šādas komandas:

**Composer instalācija:**

composer install

php artisan key:generate

php artisan migrate --seed

php artisan storage:link

**Lietojumprogrammu saites:**

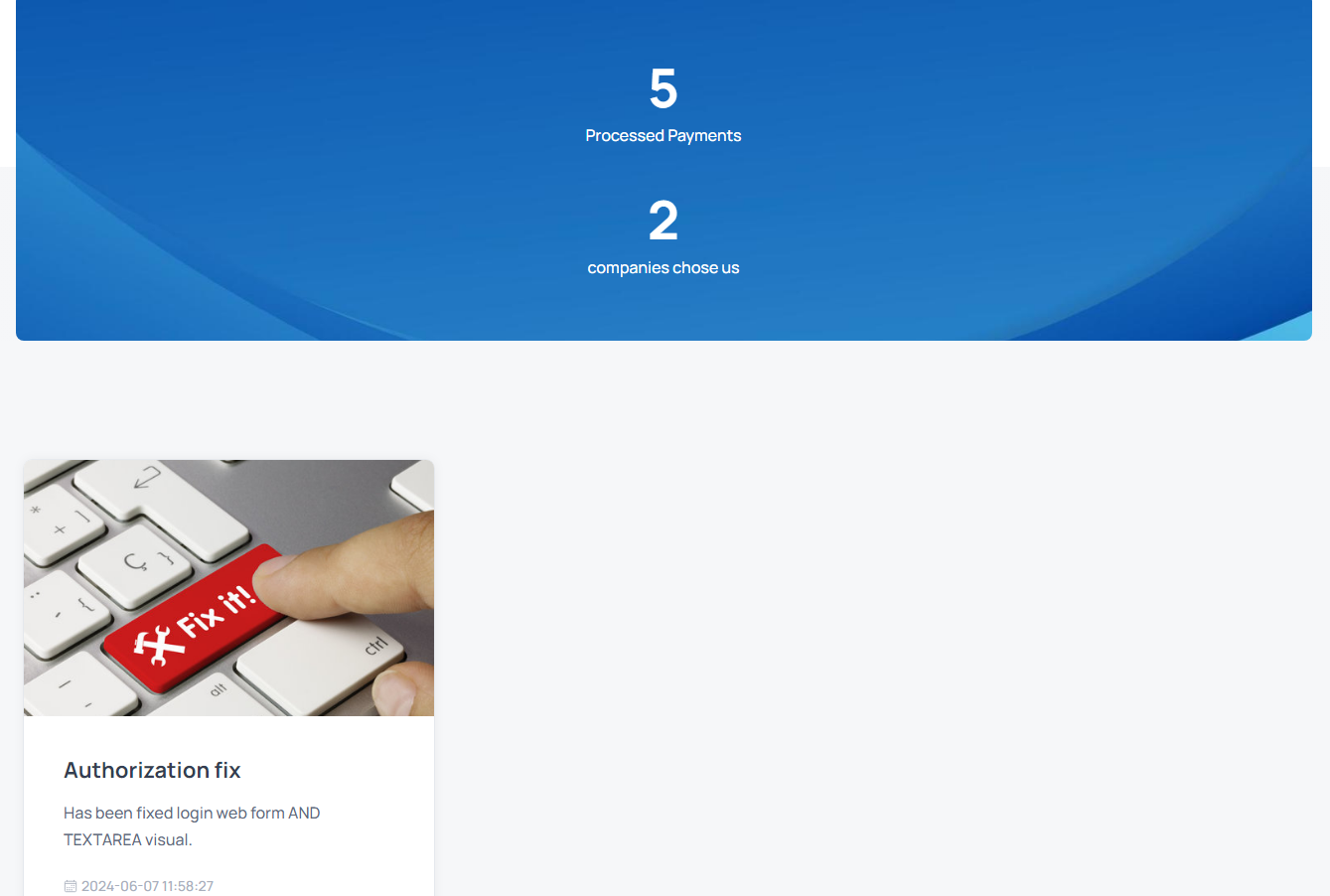
Saite uz admin paneli (noklusējuma e-pasts: test@gmail.com, parole - 12345678): http://eurocipari.com:8000/login

Būs nepieciešama arī datubāze. Manā gadījumā tiek izmantota mysql 8.0. (<https://dev.mysql.com/downloads/installer/>).

## **6.3. Programmas apraksts**

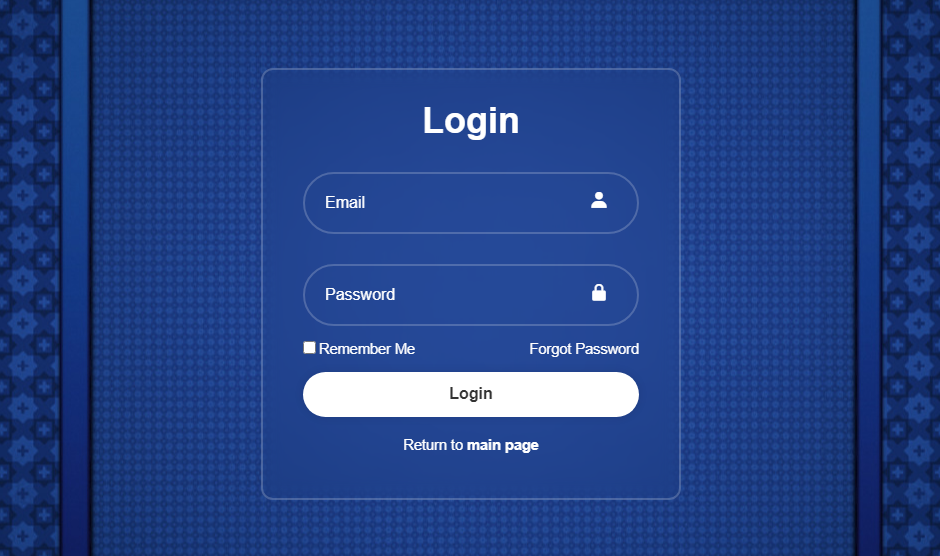
Atverot vietni, varat redzēt vietnes informācijas daļu, pogu, lai ieietu panelī jautājuma nosūtīšanai, un pogu, lai pieteiktos uzņēmuma kontā. Galvenā lapa ir informatīva rakstura (skat. 11.att. galvenā lapa).

11.att. galvenā lapa

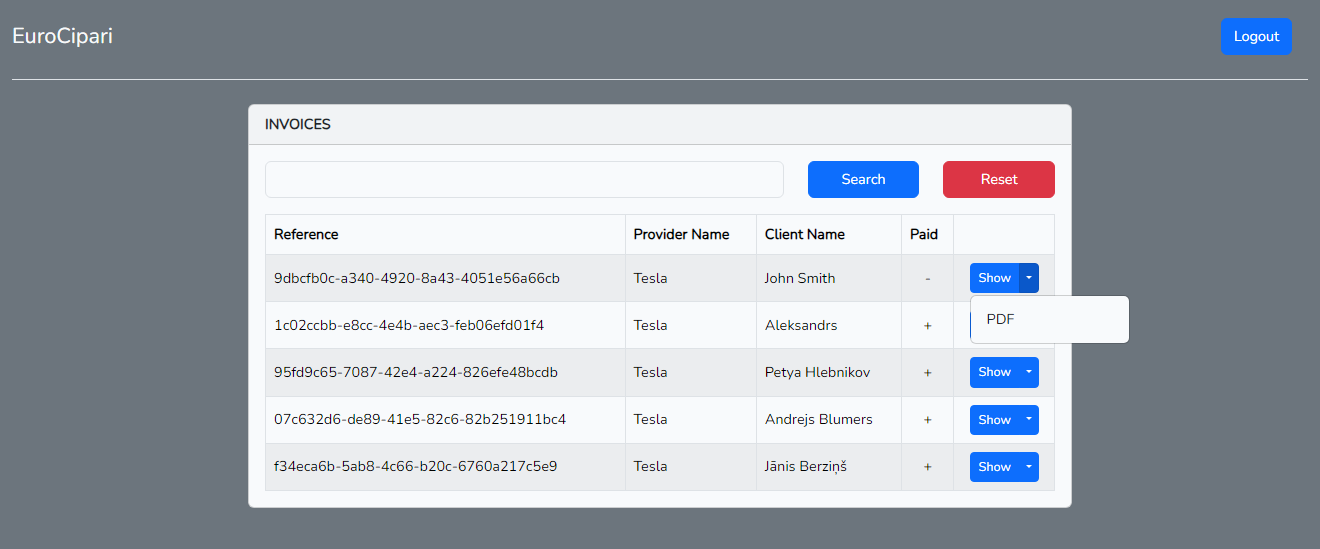
Galvenās lapas apakšā varat redzēt ziņu bloku ar jaunākajiem vietnes atjauninājumiem. Tāpat virs bloga ir automātisks aprēķins, cik uzņēmums ir pieslēdzies sistēmai un cik maksājumi ir izgājuši caur manu sistēmu (skat. 12.att. galvenā lapa apakšā daļa).

12.att. galvenā lapa apakšā daļa

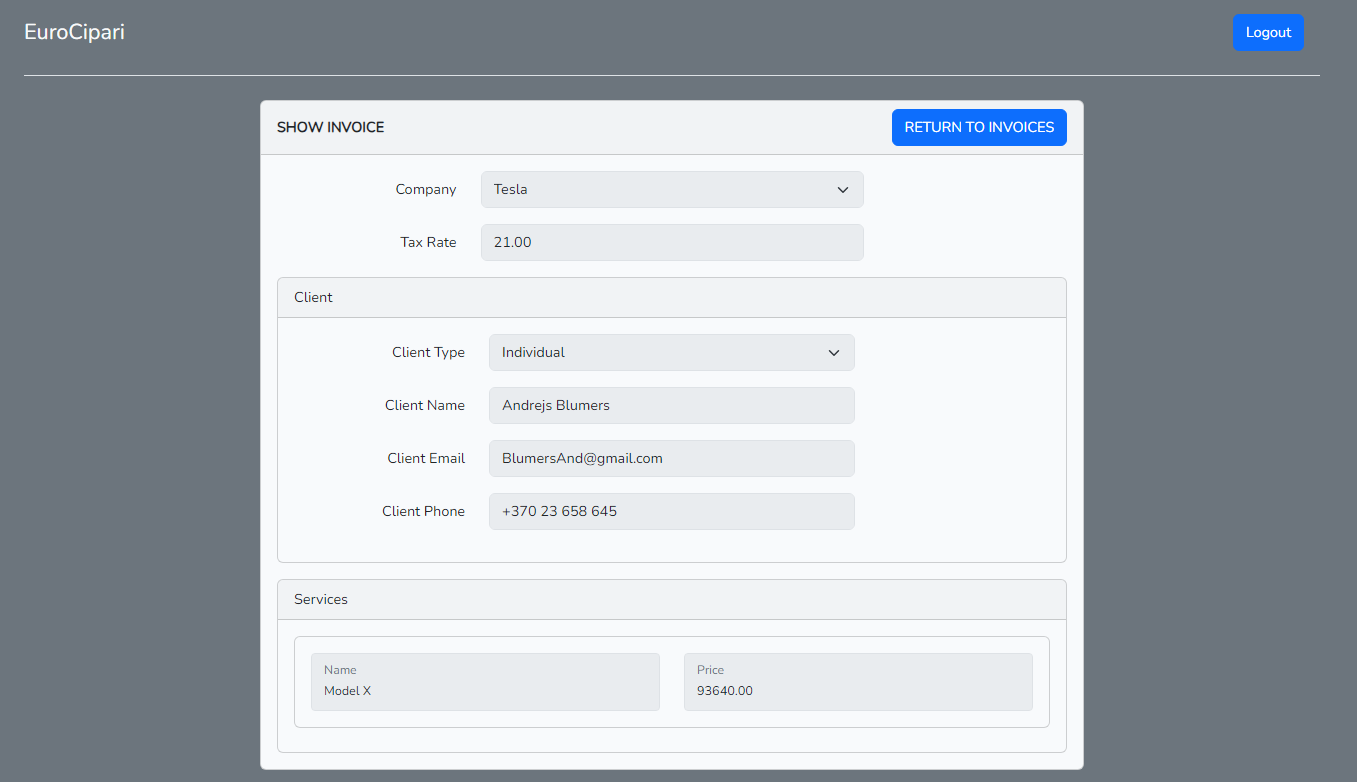
**Uzņēmumi:**

Uzklikšķinot uz uzņēmuma paneļa pogu sākuma lapa, uzņēmumi var pieteikties, izmantojot tiem piešķirto parole ar epastu (skat. 13.att.uzņēmuma pieteikšana).

13.att. uzņēmuma pieteikšana

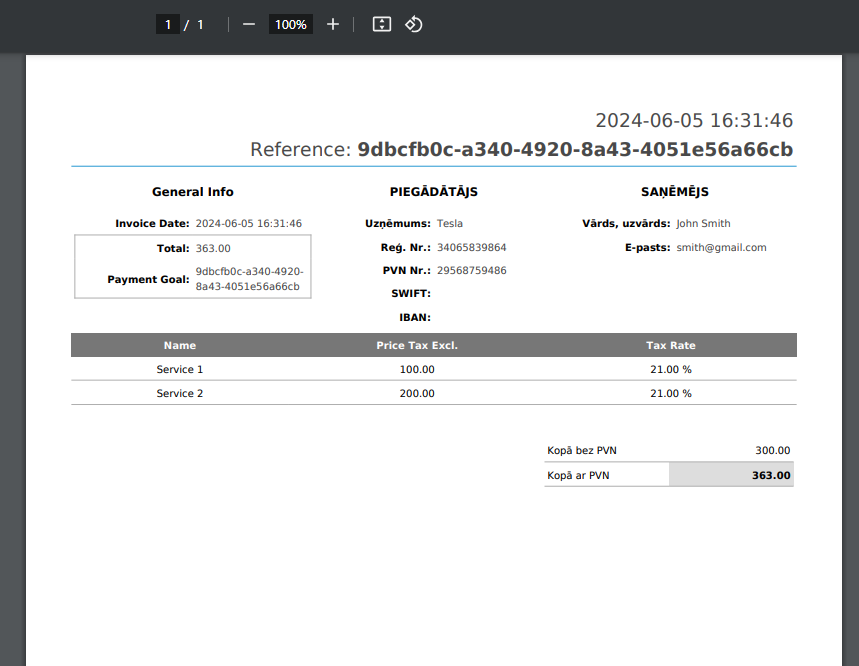
Pēc pieteikšanās jūsu profilā uzņēmums var redzēt visu informāciju par saviem kontiem. Detalizētāku informāciju par rēķinu uzņēmums var uzzināt, noklikšķinot uz skata pogu “show”. Pieejams arī PDF formats un lai izietu no sava konta, jānoklikšķina uz pogu izrakstīties, jeb Logout (skat. 14.att. uzņēmuma profils).

14.att. uzņēmuma profils

Rēķina informācija, kurā var redzēt personu, kas kaut ko iegādājās, un preci, ko viņš iegādājās (skat. 15.att.uzņēmuma rēķini).

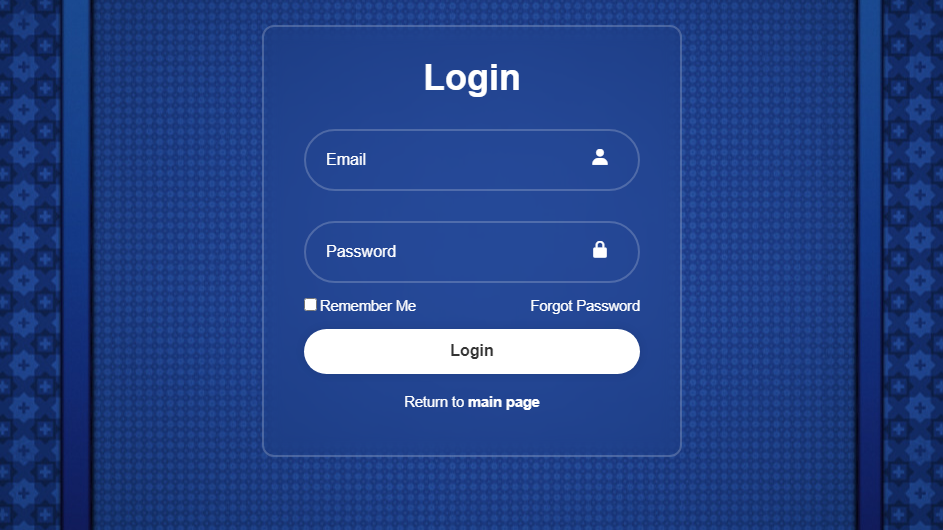
15.att. uzņēmuma ŗēķini

Ja nepieciešams saglabāt maksājumu vai uzzināt vēl detalizētāku informāciju, tad maksājumu iespējams saglabāt pdf formātā. Lai to izdarītu, uzņēmuma profila galvenajā lapā jānoklikšķina uz PDF (skat. 16.att. PDF).

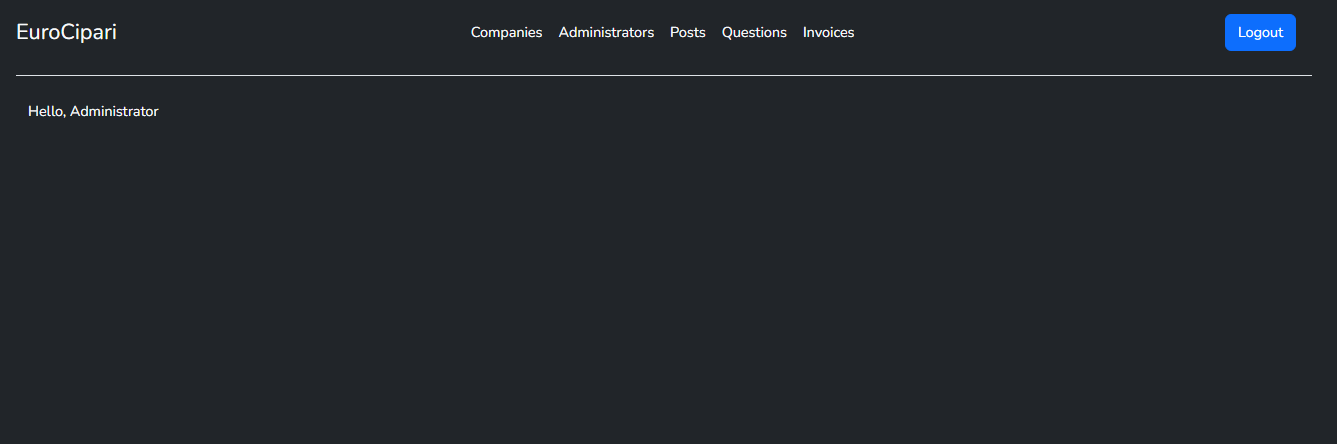


16.att. PDF

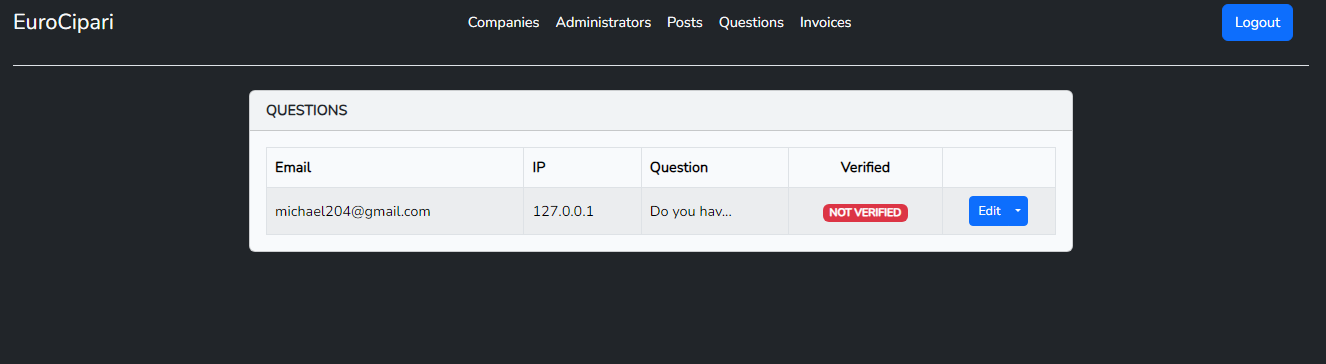
**Administrators:**

Tā kā parastajiem lietotājiem, tāpat kā lielākajai daļai lielu projektu, nav piekļuves administratora panelim, meklēšanas joslā ir jāievada “/admin”, lai ievadītu paneli, un pēc tā ievadīšanas tiksiet novirzīts uz administratora pieteikšanās paneli (skat. 17.att. admin pieteikšana).

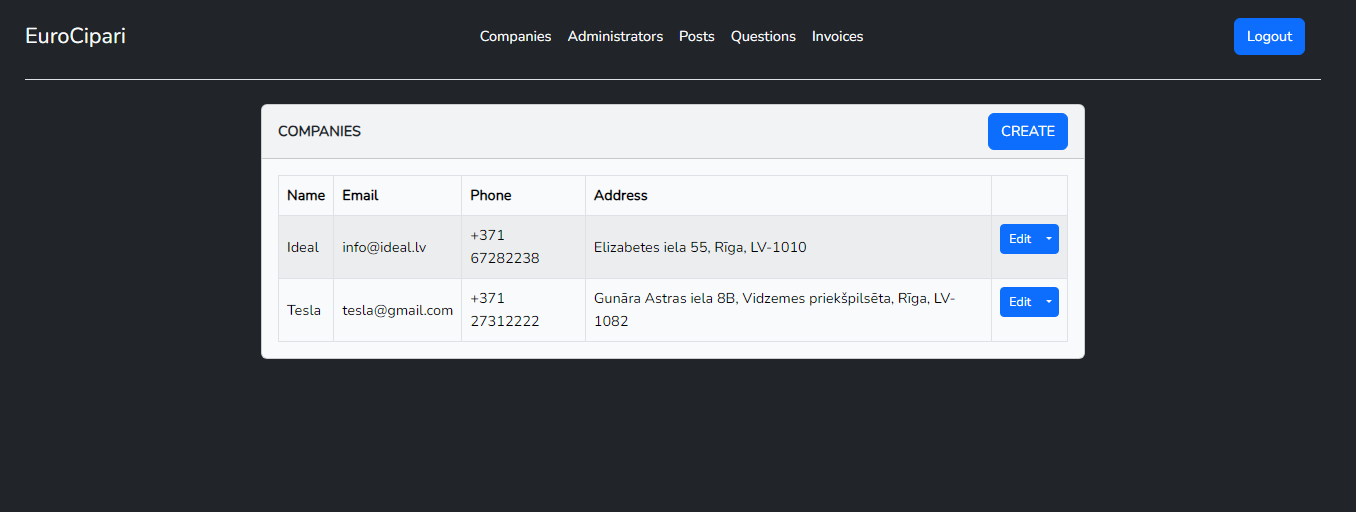
17.att. admin pieteikšana

Pēc autorizācijas administrators nonāk savā panelī, kur var visu administrēt. Pārvaldīt visus uzņēmumus, pārvaldīt administratorus, pārvaldīt maksājumus un atbildēt uz klientu jautājumiem, pārvaldīt ziņas (skat. 18.att. admin panelis).

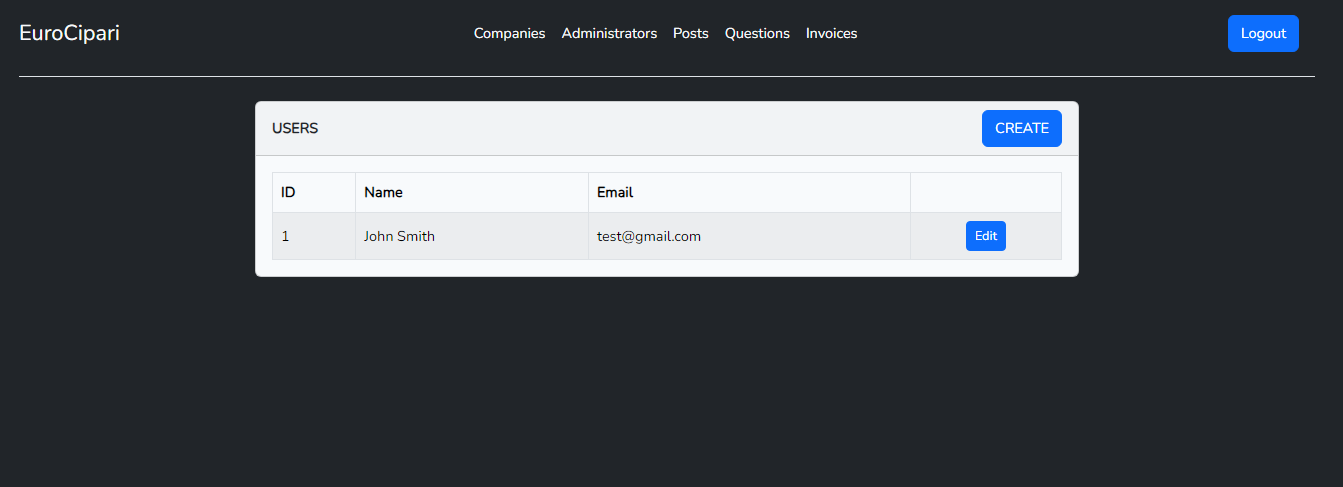
18.att. admin panelis

 Administratori var skatīt nosūtītos jautājumus jautājumu cilnē, viņi var arī nekavējoties atzīmēt jautājuma statusu un nosūtīt lietotājiem atbildi (skat. 19.att. jautājumi).

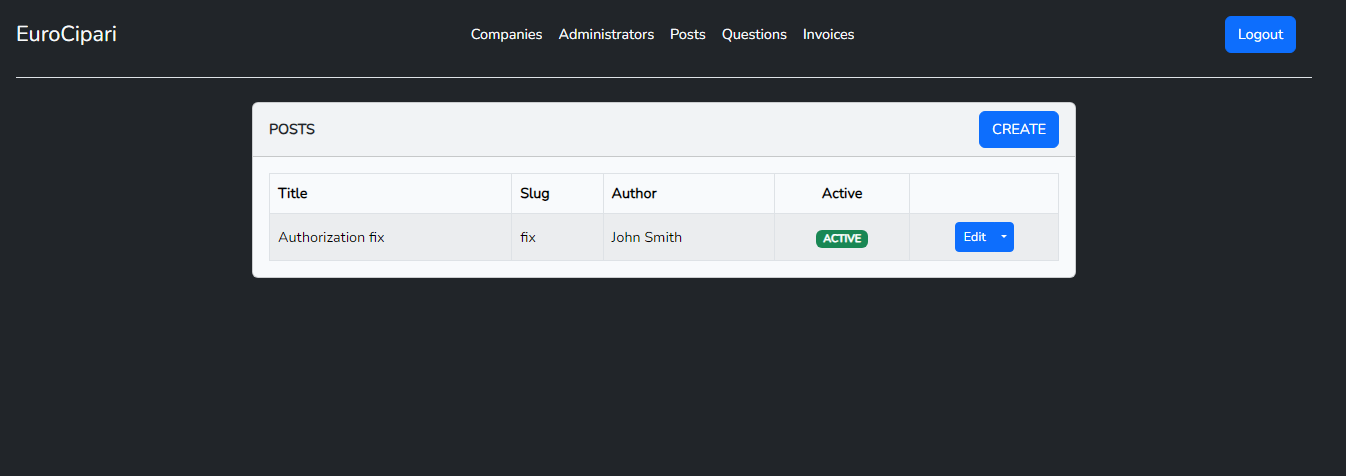
19.att. jautājumi

Uzņēmuma cilnē administrators var pilnībā pārvaldīt visus uzņēmumus, kā arī rediģēt vai dzēst esošos uzņēmumus, atvienojot tos no sistēmas (skat. 20.att. uzņēmumi).

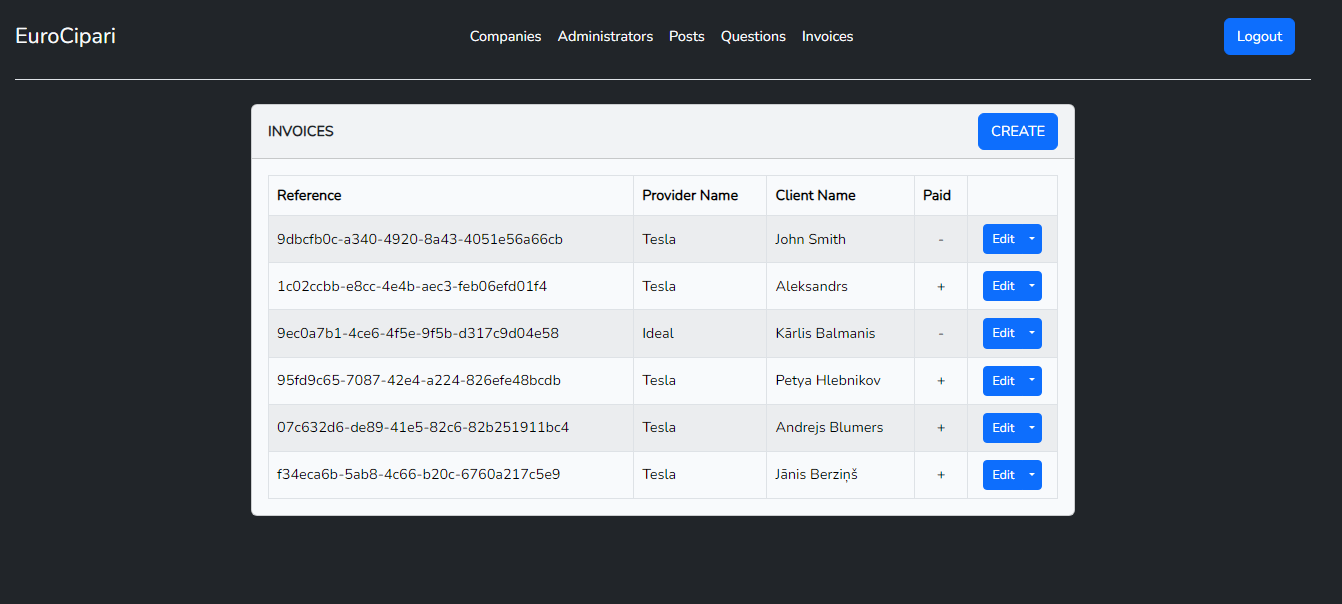
20.att. uzņēmumi

Administratora cilnē galvenais administrators var pilnībā pārvaldīt visus administratorus, kā arī rediģēt vai dzēst esošos (skat. 21.att. administratori).

21.att. administratori

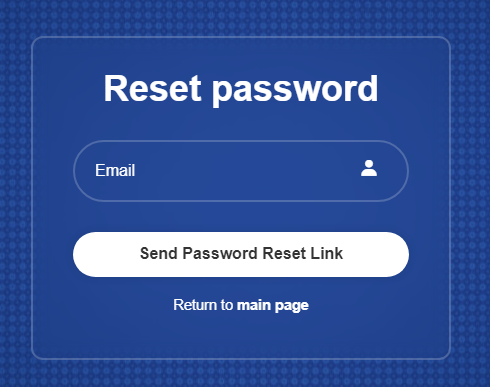
Cilnē Ziņas administrators atjaunina informāciju vietnē, izveidojot ziņas par sistēmas atjauninājumiem vai pievienojot dažas svarīgas ziņas un informāciju (skat. 22.att. ziņas).

22.att. ziņas

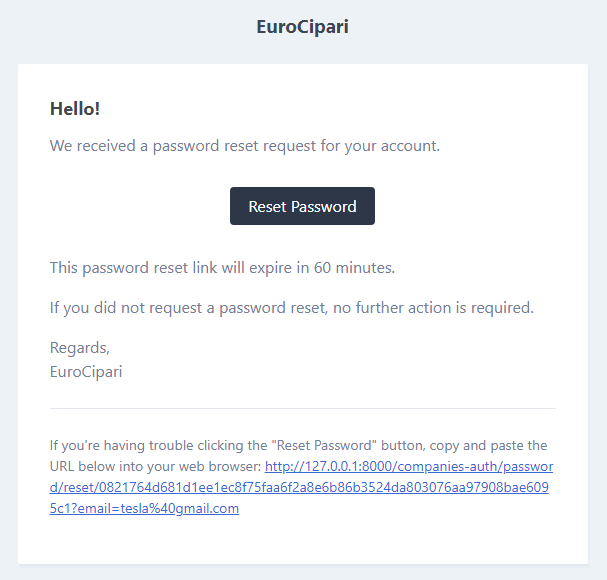
Maksājumu cilnē administrators var redzēt visus maksājumus, viņš var rediģēt maksājumu, ja lietotājs sazinās un saka, ka kaut kas norādīts nepareizi, var apskatīt un rediģēt arī pašus maksājumus, var dzēst nevajadzīgos maksājumus un izsekot maksājumu statusam. No šī paneļa administratoram ir iespējams manuāli izveidot maksājumu un, noklikšķinot uz pogas, nosūtīt to uz e-pastu, nosūtīt klientam vēstuli ar lūgumu samaksāt maksājumu (skat. 23.att. maksājumi).

23.att. maksājumi

**Parole atjaunonājums priekš lietotājiem un administratoriem:**

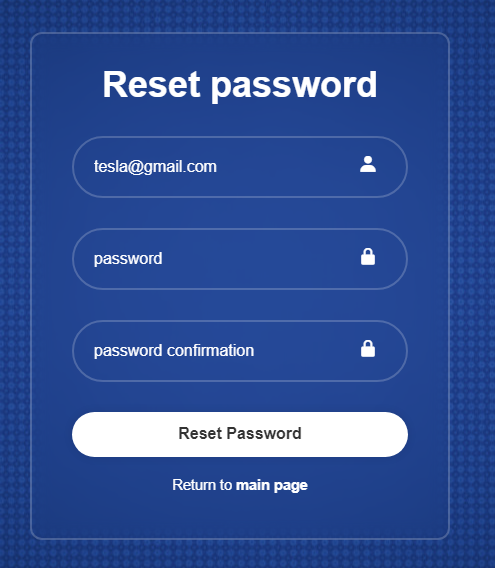
Noklikšķinot uz pogu “Atjaunot paroli” konta pieteikšanās logā, tiks novirzīts uz logu, kurā būs jāievada e-pasta adrese (skat. 24.att. aizmirsu paroli).

24.att. aizmirsu paroli

Pēc tam tiks nosūtīts epasts ar saiti, lai atjaunotu paroli (skat. 25.att. paroles atjaunošana).

25.att. paroles atjaunošana

Lai saņemtu jaunu paroli, pēc noklikšķināšanas uz saites no vēstules, atvērtajā logā ievadiet savu e-pastu, jauno paroli un paroles apstiprinājumu, pēc kura parole tiks atjaunināts (skat. 26.att. paroles atjaunošana vietnes izskats).

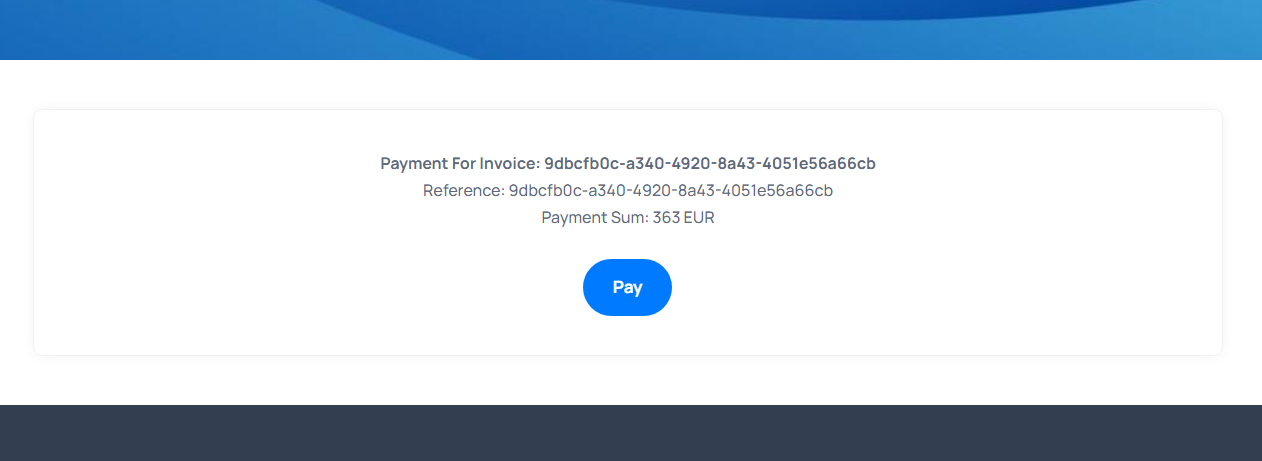


26.att. paroles atjaunošana vietnes izskats

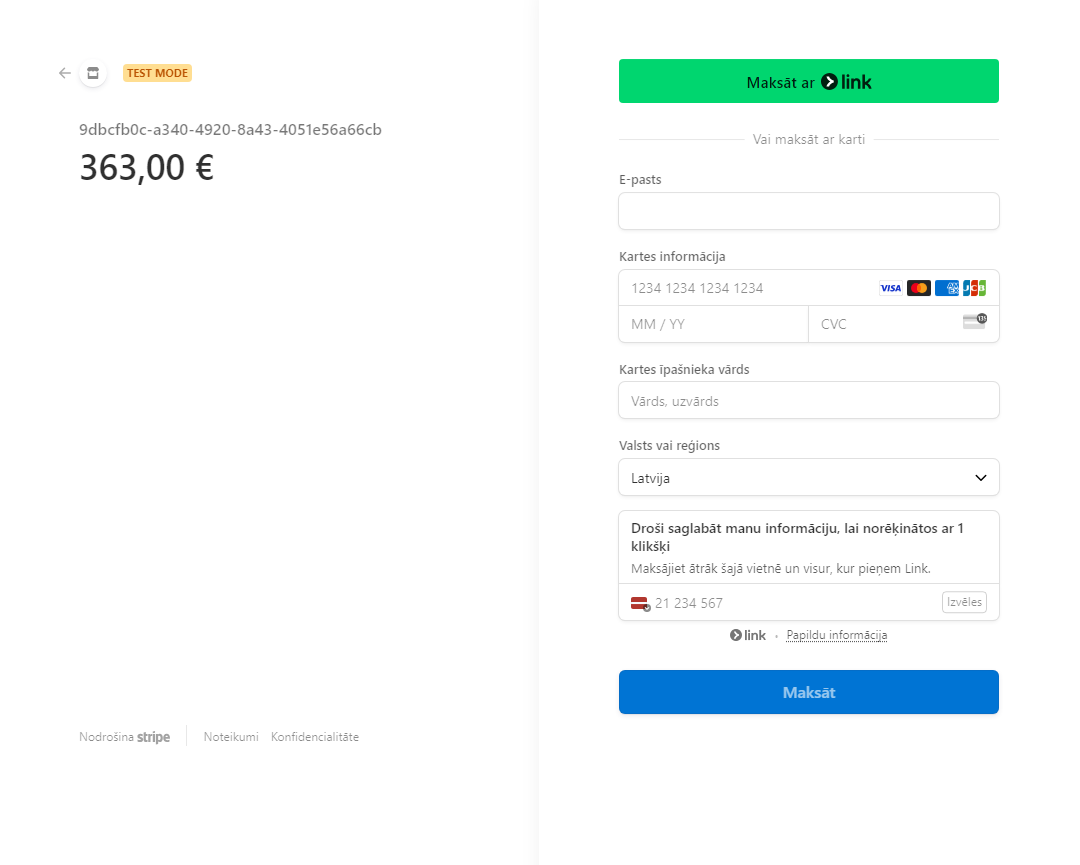
**Rēķinu apmaksa:**

Kad ir izveidots jauns maksājums, pēc tā izveidošanas uz e-pastu tiek nosūtīta vēstule ar linku šī rēķina apmaksai, vēstulē norādīta maksājuma summa un numurs (skat. 27.att. maksājuma vēstule).

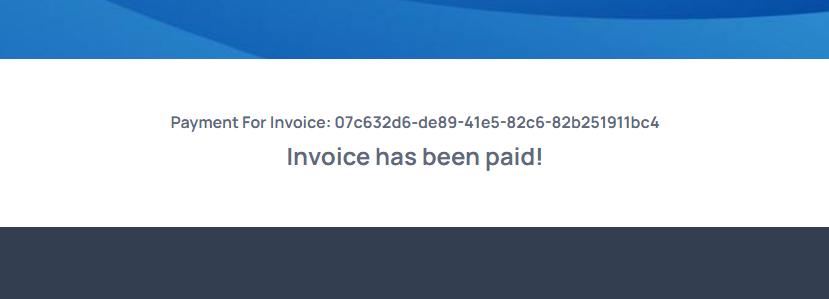
27.att. maksājuma vēstule

Noklikšķinot uz vēstulē esošo linku, atvērsies logs, kurā tiks norādīts maksājuma numurs un maksājuma summa. Ja viss ir pareizi un dati sakrīt, tad varat noklikšķināt uz apmaksāt (skat. 27.att. maksājuma parbaude).

27.att. maksājuma parbaude

Pēc datu pārbaudes un nospiešanas uz pogas apmaksāt, atvērsies maksājuma logs, kurā būs jāievada kartes dati un lietotāja dati, lai apmaksātu rēķinu (skat. 28.att. bankas datu ievadīšana).

28.att. bankas datu ievadīšana

Pēc veiksmīga maksājuma jūs tiksit novirzīts uz logu vietnes lapā, kurā jums tiks paziņots, ka maksājums ir veiksmīgi apmaksāts (skat. 29.att. veiksmīgi apmaksāts).

29.att. veiksmīgi apmaksāts

## **6.4. Testa piemērs**

6.tabula testēšana

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nr.** | **Prasības numurs** | **Prasības nosaukums** | **Ievaddati/situācijas apraksts** | **Sagaidāmais rezultāts** | **Statuss** |
| 1. | 1. | Lietotāja autentifikācija | Pareizi uzņēmuma dati: e-pasts [tesla@gmail.com](mailto:tesla@gmail.com), parole teslatesla123 | Dati ievadīti pareizi, sistēma dot iespēju pieslēgties uzņēmumam | Pareizi |
| 2. | e-pasts tesslaa@gmail.com | e-pasts ievadīts nepareizi, sistēma izvada kļūdu | pareizi |
| 3. | Parole tesilatesla123 | Parole ievadīts nepareizi, sistēma izvada kļūdu | pareizi |
| 4. | e-pasts tesla@gmail.com | e-pasts jau eksiste, sistēma izvada par to ziņojumu | pareizi |
| 5. | 2. | Uzņēmums un tā pārstāvji | Funkcija atgriež visus darījumus lietotāja profila | Darījumu vēsture ir apskātama profilā | pareizi |
| 6. | To pašu var apskatīt caur API | Izmantojot kaut kādu rīku, var to apskatīt arī caur API | pareizi |
| 7. | 3. | Darījumu veikšana | Link uz apmāksu atsutīts uz e-pastu nospiežot “Send email” | Uz e-pastu pināk vēstule ar linku uz apmāksu | pareizi |
| 8. | Apmaksa statuss | Kad rēķins ir apmaksāts, nevar to pašu atkartoti apmaksāt un statuss atzimēts ar + | pareizi |
| 9. | 4. | Administrācijas funkcijas | Parvaldība | Nodzesot uzņēmumu vai administratoru, viņš vairs nevar pieslēgties | pareizi |
| 10. | Rēķini | Ja ar automatisko rēķinu problēma, viņu var pievienot ar rokam | pareizi |

Tika veiktas 10 pārbaudes, no kurām tika identificēti programmas un tās funkciju plusi un mīnusi, šī rezultāta sasniegšanai tika novērsti un laboti visi trūkumi, lai visur sasniegtu rezultātu “pareizi”.

# **NOBEIGUMS**

Visi mērķi un uzdevumi ir sasniegti, jo ieplānotais rezultāts ir izpildīts uz 100%. Finanšu sitēmas vietne tika izveidota, izmantojot HTML, CSS, JavaScript un PHP, un es visu rakstīju, izmantojot LARAVEL frameworku.

Darba gaitā tika veikta programmatūras produkta analīze un prasības, izstrādāts programmatūras produkta projekts, izstrādāts ER datu modelis, datu tabulas, interfeiss, izstrādāta programmatūras produkta loģika. Izstrādāta un izstrādātā programmatūra ir tīmekļa lietojumprogramma, kas ļauj vieglāk apmaksāt rēķinus un apskatīt savus rēķinus.

Turpmāka attīstība:

1. Balstoties uz projekta komentāriem un idejām, nākotnē plānots, ka tiks veiktas īsziņas ar reālu uzņēmumu, lai pēc pasūtījuma varētu sākt pārbaudīt automatizācijas procesu un automātisku rēķina nosūtīšanu uzņēmumam caur realajiem firmam.

2. Pievienot arī unikālus identifikatorus jautājumiem, lai tos būtu vieglāk atrast un pārvaldīt, ja kaut kas notiek.

3. Ja viss izdodas, tad arī publicēt vietni internetā, iegādājoties serveri un domēnu.

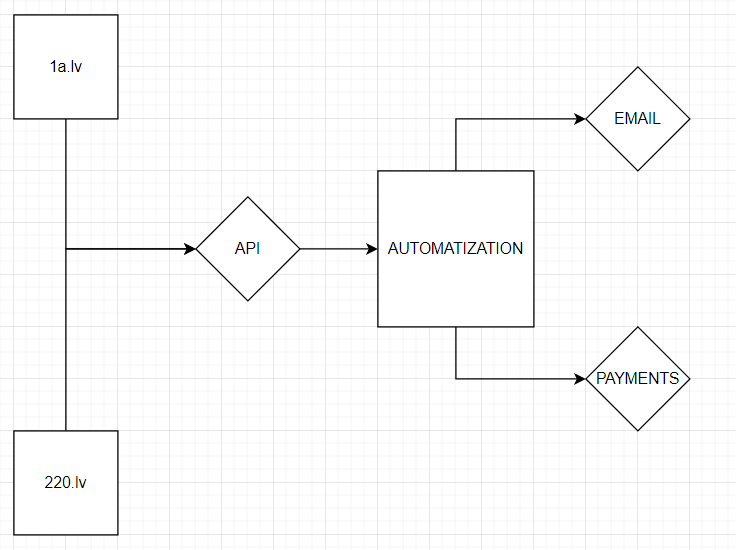
# **INFORMĀCIJAS AVOTI**

1. Laravel Documentation – <https://laravel.com/docs/10.x/readme> - (Resurss apskatīts 06.21.2024)[1].
2. Bootstrap Documentation – <https://getbootstrap.com/docs/4.1/getting-started/introduction/> - (Resurss apskatīts 06.20.2024)[2].
3. Shemu izveide – <https://app.diagrams.net/> - (Resurss apskatīts 12.13.2024)[3].
4. Skiču un dizainu veidošana – <https://www.figma.com/> - (Resurss apskatīts 15.10.2023)[4].
5. Par postman programmu – <https://www.postman.com/> - (Resurss apskatīts 10.12.2023)[5].
6. Kļūdas labošanas iespējas meklēšanas – <https://www.youtube.com/> - (Resurss apskatīts 10.05.2024)[6].
7. Informācija par css, html, js, php – <https://www.w3school.com/> - (Resurss apskatīts 10.12.2023)[7].
8. Vizuālās daļas daļa tika uzrakstīta ar YouTube – <https://www.youtube.com/watch?v=5Bl3CCizSRQ&t=424s&ab_channel=codewithsadee> – (Resurss apskatīts 03.22.2024)[8].
9. E-pasts – <https://mailtrap.io/> - (Resurss apskatīts 31.05.2024)[9].
10. Rēķini – <https://stripe.com/en-lv> - (Resurss apskatīts 31.05.2024)[10].

# PIELIKUMI

## 1.pielikums. b2b sistēmas shēma

**B2B**

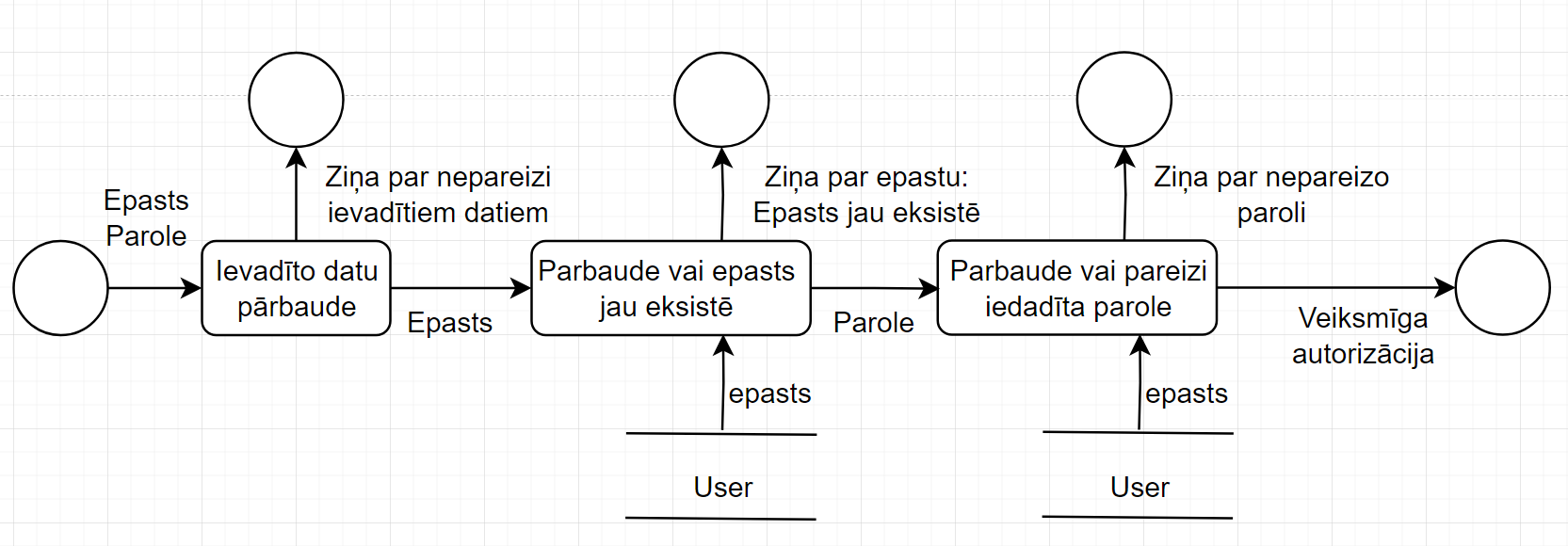
****Pārdošanu starp juridiskām personām (starp uzņēmumiem, firmām un organizācijām) parasti dēvē par B2B. B2B nozīmē “business to business” (pārdošana “no uzņēmuma citam uzņēmumam”).

1.att. b2b sistēmas shēma

|  |
| --- |
| **Pieteikšanās shēma, autorizācija** |

|  |
| --- |
| 2.pielikums. Pieteikšanās shēma, autorizācija |

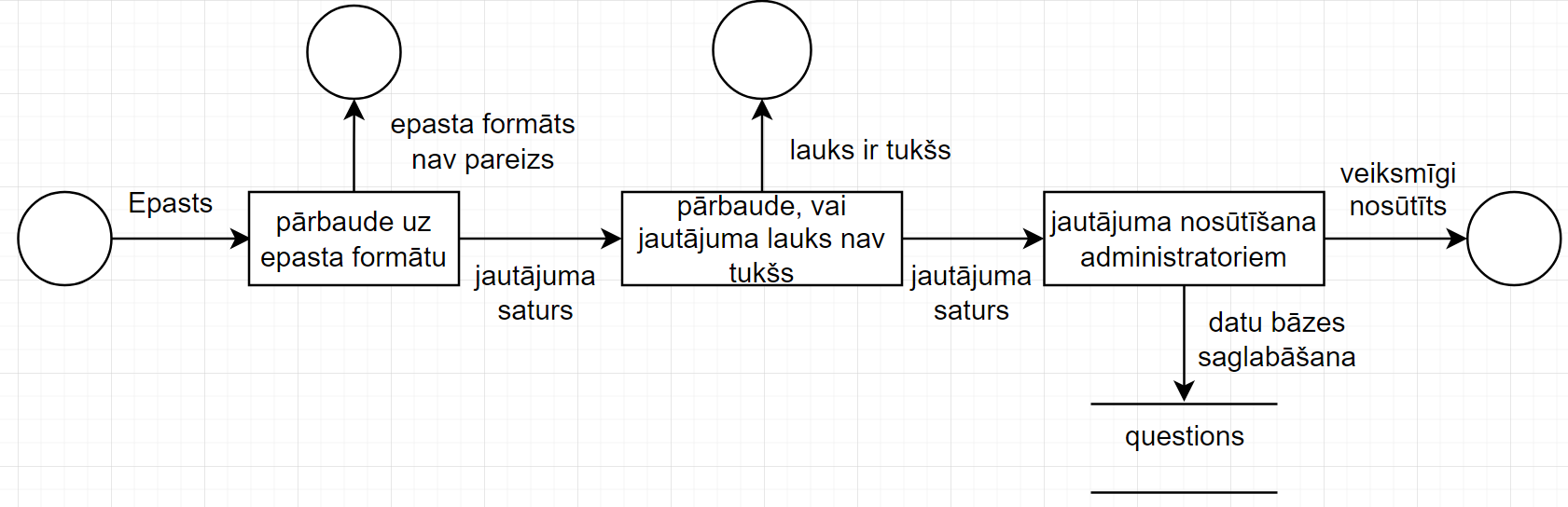
|  |
| --- |
| 2.att. autorizācijas datu plūsmas diagramma |



|  |
| --- |
| 3.pielikums. Jautājumu shēma |

|  |
| --- |
| 3.att. jautājumu datu plūsmu diagramma |

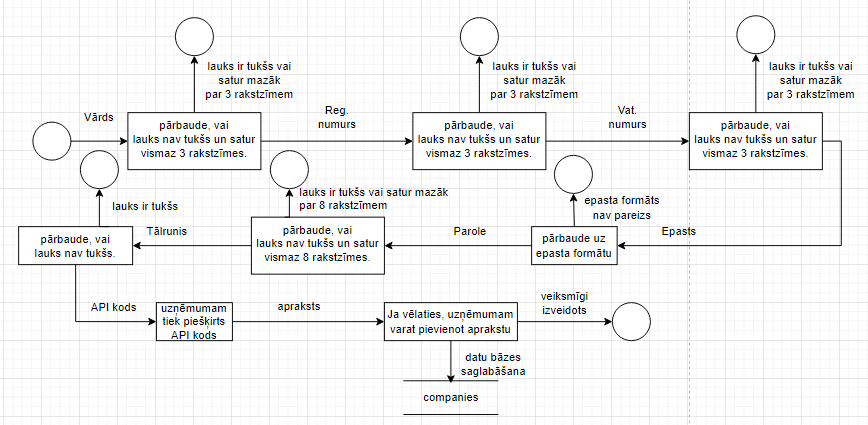
|  |
| --- |
| **Jautājumu shēma** |



|  |
| --- |
| 4.att. uzņēmumu datu plūsmu diagramma |

|  |
| --- |
| **Uzņēmumu shēma** |

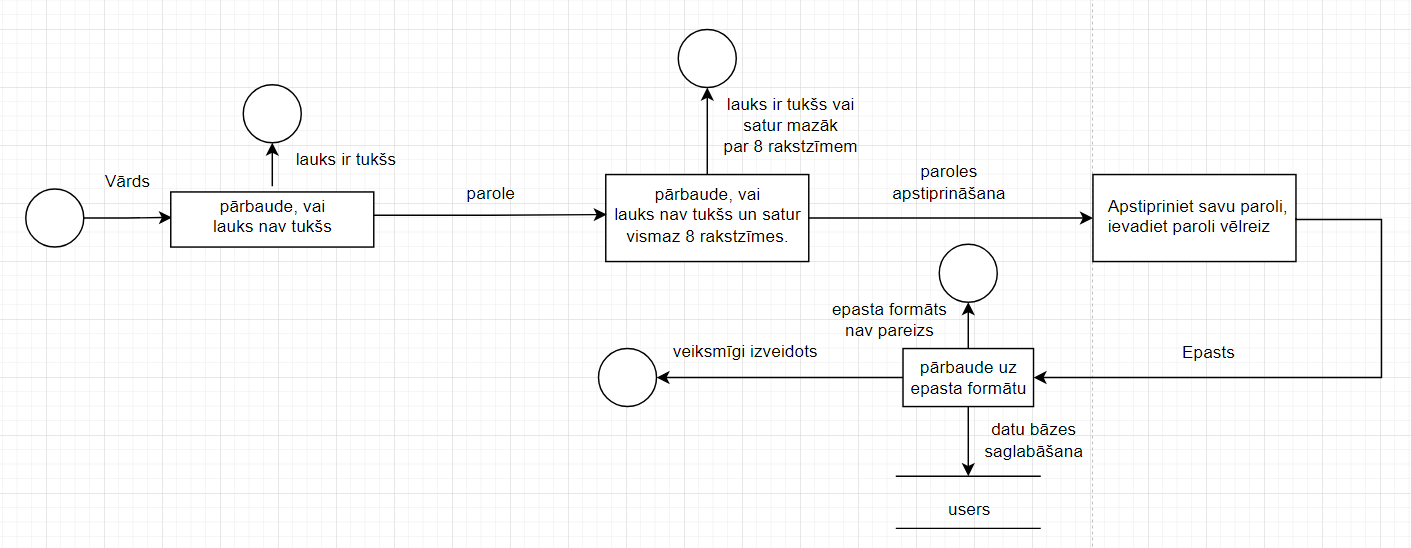
|  |
| --- |
| 4.pielikums. Uzņēmumu shēma |



|  |
| --- |
| 5.att. lietotāju datu plūsmu diagramma |

|  |
| --- |
| **Lietotāju shēma** |

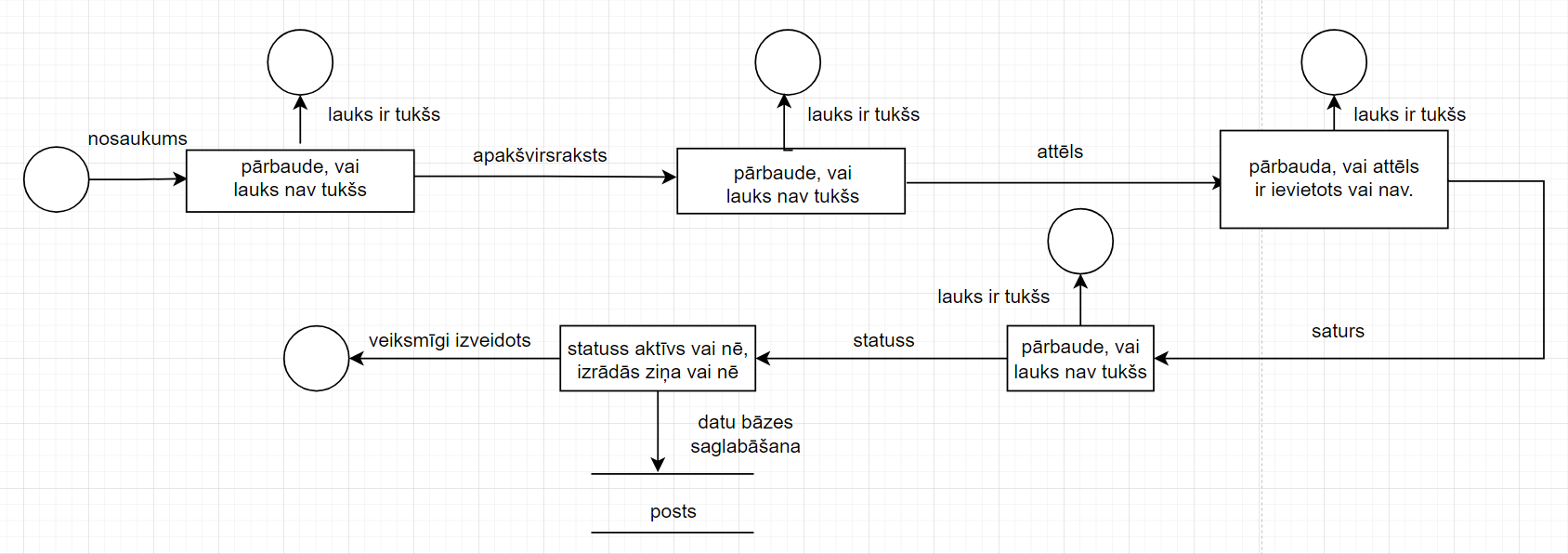
|  |
| --- |
| 5.pielikums. Lietotāju shēma |



|  |
| --- |
| 6.pielikums. Ziņu shēma |

|  |
| --- |
| 6.att. ziņu datu plūsmu diagramma |

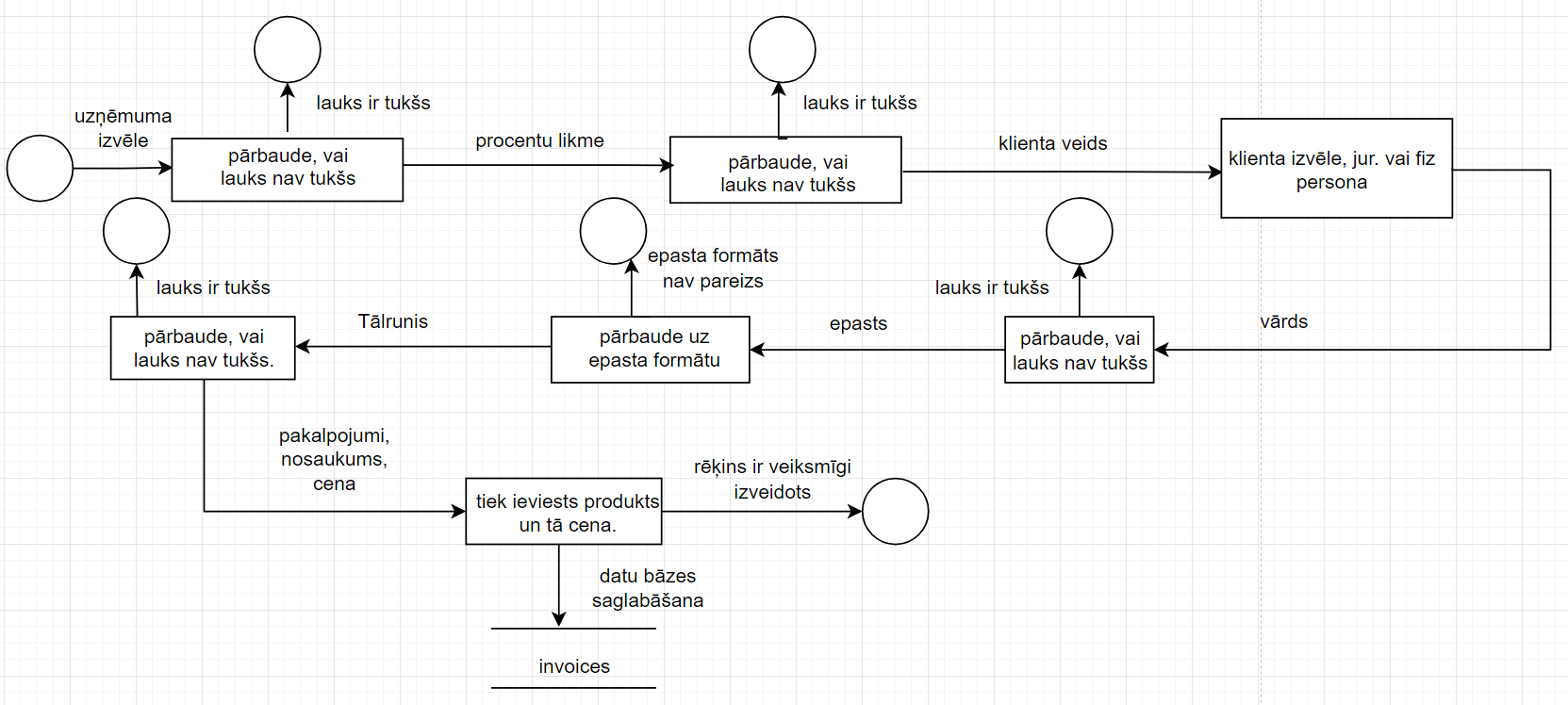
|  |
| --- |
| **Ziņu shēma** |



|  |
| --- |
| 7.pielikums. Maksājumu shēma |

|  |
| --- |
| 7.att. uzņēmumu datu plūsmu diagramma |

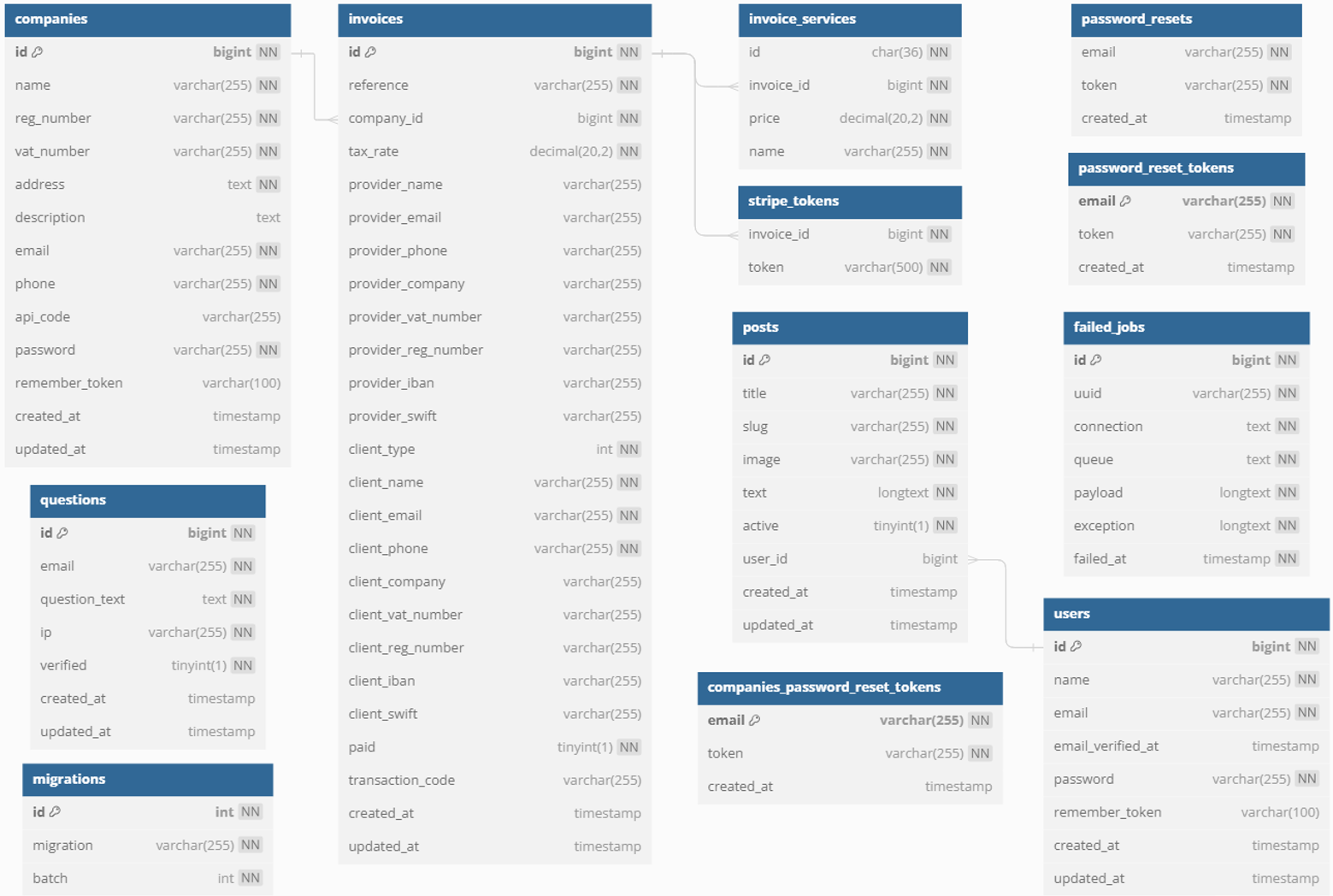
|  |
| --- |
| **Maksājumu shēma** |



|  |
| --- |
| 8.pielikums. Fiziskās struktūra |

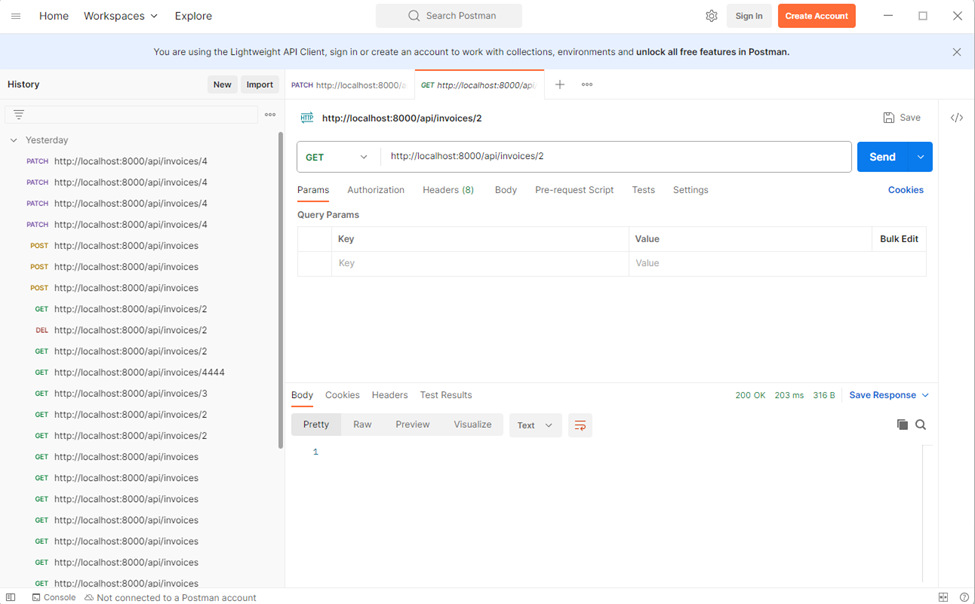
|  |
| --- |
| 8.att. datu bāzes fiziskās struktūras diagramma |

|  |
| --- |
| **Fiziskās struktūra** |



## 9.pielikums. API-Postman

**POSTMAN**

Lai paradīt darbības principu ar API no uzņēmumu puses, var izmantot Postman aplikācija. Postman ir emulācija no uzņēmuma puses maksājumu apskatei, rēķina izveidošanai un postman ir nepieciešams, lai parādītu un demonstrētu darbu sistēmā no uzņēmuma puses.

9.att. postman

## 10.pielikums. Programmas pirmteksts

**PROGRAMMAS KODS**

**app/Console/Kernel.php:**

<?php

namespace App\Console;

use Illuminate\Console\Scheduling\Schedule;

use Illuminate\Foundation\Console\Kernel as ConsoleKernel;

class Kernel extends ConsoleKernel

{

/\*\*

\* Define the application's command schedule.

\*/

protected function schedule(Schedule $schedule): void

{

// $schedule->command('inspire')->hourly();

}

/\*\*

\* Register the commands for the application.

\*/

protected function commands(): void

{

$this->load(\_\_DIR\_\_.'/Commands');

require base\_path('routes/console.php');

}

}

**app/Exceptions/Handler.php:**

<?php

namespace App\Exceptions;

use Illuminate\Foundation\Exceptions\Handler as ExceptionHandler;

use Throwable;

class Handler extends ExceptionHandler

{

/\*\*

\* The list of the inputs that are never flashed to the session on validation exceptions.

\*

\* @var array<int, string>

\*/

protected $dontFlash = [

'current\_password',

'password',

'password\_confirmation',

];

/\*\*

\* Register the exception handling callbacks for the application.

\*/

public function register(): void

{

$this->reportable(function (Throwable $e) {

//

});

}

}

**app/Http/Controllers/Admin/AdminController.php:**

<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;

use App\Http\Requests\Admin\User\UserStoreRequest;

use App\Http\Requests\Admin\User\UserUpdateRequest;

use App\Models\User;

use App\Services\UserService;

use Illuminate\Contracts\Foundation\Application;

use Illuminate\Contracts\View\Factory;

use Illuminate\Contracts\View\View;

use Illuminate\Http\RedirectResponse;

class AdminController extends Controller

{

/\*\*

\* @var UserService

\*/

private UserService $userService;

/\*\*

\* @param UserService $userService

\*/

public function \_\_construct(UserService $userService)

{

$this->userService = $userService;

}

/\*\*

\* Display a listing of the resource.

\*

\* @return Application|Factory|View

\*/

public function index(): View|Factory|Application

{

$users = User::orderBy('id', 'desc')->paginate(10);

$currentUser = $this->userService->getCurrentUser();

return view('admin.user.index', compact('users', 'currentUser'));

}

/\*\*

\* Show the form for creating a new resource.

\*

\* @return Factory|View|Application

\*/

public function create(): Factory|View|Application

{

return view('admin.user.create');

}

/\*\*

\* Store a newly created resource in storage.

\*

\* @param UserStoreRequest $request

\* @return RedirectResponse

\*/

public function store(UserStoreRequest $request): RedirectResponse

{

$user = $this->userService->store($request->validated());

return redirect()->route('admin.users.edit', $user)->with('success', \_\_('User created successfully.'));

}

/\*\*

\* Show the form for editing the specified resource.

\*

\* @param User $user

\* @return Application|Factory|View

\*/

public function edit(User $user): View|Factory|Application

{

return view('admin.user.edit', compact('user'));

}

/\*\*

\* Update the specified resource in storage.

\*

\* @param UserUpdateRequest $request

\* @param User $user

\* @return RedirectResponse

\*/

public function update(UserUpdateRequest $request, User $user): RedirectResponse

{

$this->userService->update($user, $request->validated());

return redirect()->route('admin.users.edit', $user)->with('success', \_\_('User updated successfully.'));

}

/\*\*

\* Remove the specified resource from storage.

\*

\* @param User $user

\* @return RedirectResponse

\*/

public function destroy(User $user): RedirectResponse

{

// prevent possibility when user deletes himself

$currentUser = $this->userService->getCurrentUser();

if ($currentUser && $currentUser->id === $user->id) {

return redirect()->back()->with('error', \_\_('It is not possible to delete yourself.'));

}

$this->userService->destroy($user);

return redirect()->route('admin.users.index')->with('success', \_\_('User deleted successfully.'));

}

}

**app/Http/Controllers/Admin/CompanyController.php:**

<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;

use App\Http\Requests\Admin\Company\CompanyCreateRequest;

use App\Http\Requests\Admin\Company\CompanyUpdateRequest;

use App\Models\Company;

use App\Services\CompanyService;

use App\Services\UserService;

use Illuminate\Contracts\Foundation\Application;

use Illuminate\Contracts\View\Factory;

use Illuminate\Contracts\View\View;

use Illuminate\Http\RedirectResponse;

class CompanyController extends Controller

{

/\*\*

\* @var CompanyService

\*/

private CompanyService $companyService;

/\*\*

\* @param CompanyService $companyService

\*/

public function \_\_construct(CompanyService $companyService)

{

$this->companyService = $companyService;

}

/\*\*

\* Display a listing of the resource.

\*

\* @return Application|Factory|View

\*/

public function index(): View|Factory|Application

{

$companies = Company::orderBy('id', 'desc')->paginate(10);

return view('admin.company.index', compact('companies'));

}

/\*\*

\* Show the form for creating a new resource.

\*

\* @return Factory|View|Application

\*/

public function create(): Factory|View|Application

{

return view('admin.company.create');

}

/\*\*

\* Store a newly created resource in storage.

\*

\* @param CompanyCreateRequest $request

\* @return RedirectResponse

\*/

public function store(CompanyCreateRequest $request): RedirectResponse

{

$company = $this->companyService->create($request->validated());

return redirect()->route('admin.companies.edit', $company)->with('success', \_\_('Company created successfully.'));

}

/\*\*

\* Show the form for editing the specified resource.

\*

\* @param Company $company

\* @return Application|Factory|View

\*/

public function edit(Company $company): View|Factory|Application

{

return view('admin.company.edit', compact('company'));

}

/\*\*

\* Update the specified resource in storage.

\*

\* @param CompanyUpdateRequest $request

\* @param Company $company

\* @return RedirectResponse

\*/

public function update(CompanyUpdateRequest $request, Company $company): RedirectResponse

{

$this->companyService->update($company, $request->validated());

return redirect()->route('admin.companies.edit', $company)->with('success', \_\_('Company updated successfully.'));

}

/\*\*

\* Remove the specified resource from storage.

\*

\* @param Company $company

\* @return RedirectResponse

\*/

public function destroy(Company $company): RedirectResponse

{

$this->companyService->destroy($company);

return redirect()->route('admin.companies.index')->with('success', \_\_('Company deleted successfully.'));

}

}

**app/Http/Controllers/Admin/DashboardController.php:**

<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;

use Illuminate\Contracts\View\Factory;

use Illuminate\Contracts\View\View;

use Illuminate\Foundation\Application;

class DashboardController extends Controller

{

public function index(): View|Application|Factory|\Illuminate\Contracts\Foundation\Application

{

return view('admin/dashboard/index');

}

}

**app/Http/Controllers/Admin/InvoiceController.php:**

<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;

use App\Http\Requests\Admin\Invoice\InvoiceStoreRequest;

use App\Http\Requests\Admin\Invoice\InvoiceUpdateRequest;

use App\Models\Company;

use App\Models\Invoice;

use App\Models\User;

use App\Services\InvoiceService;

use App\Services\MailSendingService;

use DomainException;

use Illuminate\Contracts\Foundation\Application;

use Illuminate\Contracts\Routing\ResponseFactory;

use Illuminate\Contracts\View\Factory;

use Illuminate\Contracts\View\View;

use Illuminate\Http\RedirectResponse;

use Illuminate\Http\Response;

class InvoiceController extends Controller

{

/\*\*

\* @var InvoiceService

\*/

private InvoiceService $invoiceService;

/\*\*

\* @var MailSendingService

\*/

private MailSendingService $mailSendingService;

/\*\*

\* @param InvoiceService $invoiceService

\* @param MailSendingService $mailSendingService

\*/

public function \_\_construct(InvoiceService $invoiceService, MailSendingService $mailSendingService)

{

$this->invoiceService = $invoiceService;

$this->mailSendingService = $mailSendingService;

}

/\*\*

\* Display a listing of the resource.

\*

\* @return Application|Factory|View

\*/

public function index(): View|Factory|Application

{

$invoices = Invoice::orderBy('id', 'desc')->paginate(10);

return view('admin.invoice.index', compact('invoices'));

}

/\*\*

\* Show the form for creating a new resource.

\*

\* @return Factory|View|Application

\*/

public function create(): Factory|View|Application

{

$companies = Company::all();

$clientTypes = Invoice::getClientTypes();

return view('admin.invoice.create', compact('companies', 'clientTypes'));

}

/\*\*

\* Store a newly created resource in storage.

\*

\* @param InvoiceStoreRequest $request

\* @return RedirectResponse

\*/

public function store(InvoiceStoreRequest $request): RedirectResponse

{

$invoice = $this->invoiceService->store($request->validated());

return redirect()->route('admin.invoices.edit', $invoice)->with('success', \_\_('Invoice created successfully.'));

}

/\*\*

\* Show the form for editing the specified resource.

\*

\* @param Invoice $invoice

\* @return Application|Factory|View

\*/

public function edit(Invoice $invoice): View|Factory|Application

{

$companies = Company::all();

$clientTypes = Invoice::getClientTypes();

return view('admin.invoice.edit', compact('invoice', 'companies', 'clientTypes'));

}

/\*\*

\* Update the specified resource in storage.

\*

\* @param InvoiceUpdateRequest $request

\* @param Invoice $invoice

\* @return RedirectResponse

\*/

public function update(InvoiceUpdateRequest $request, Invoice $invoice): RedirectResponse

{

$this->invoiceService->update($invoice, $request->validated());

return redirect()->route('admin.invoices.edit', $invoice)->with('success', \_\_('Invoice updated successfully.'));

}

/\*\*

\* Remove the specified resource from storage.

\*

\* @param Invoice $invoice

\* @return RedirectResponse

\*/

public function destroy(Invoice $invoice): RedirectResponse

{

$this->invoiceService->destroy($invoice);

return redirect()->route('admin.invoices.index')->with('success', \_\_('Invoice deleted successfully.'));

}

/\*\*

\* @param Invoice $invoice

\* @return Response|Application|ResponseFactory

\*/

public function show(Invoice $invoice): Response|Application|ResponseFactory

{

return response($this->invoiceService->show($invoice))->header('Content-Type', 'application/pdf');

}

public function sendEmail(Invoice $invoice): RedirectResponse

{

try {

$this->mailSendingService->sendUnpaidInvoiceMessage($invoice);

} catch (DomainException $e) {

return redirect()->back()->with('error', $e->getMessage());

}

return redirect()->back()->with('success', \_\_('Email has been sent successfully.'));

}

}

**app/Http/Controllers/Admin/PostController.php:**

<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;

use App\Http\Requests\Admin\Post\PostStoreRequest;

use App\Http\Requests\Admin\Post\PostUpdateRequest;

use App\Models\Post;

use App\Services\PostService;

use App\Services\UserService;

use Illuminate\Contracts\Foundation\Application;

use Illuminate\Contracts\View\Factory;

use Illuminate\Contracts\View\View;

use Illuminate\Http\RedirectResponse;

class PostController extends Controller

{

/\*\*

\* @var PostService

\*/

private PostService $postService;

/\*\*

\* @var UserService

\*/

private UserService $userService;

/\*\*

\* @param PostService $postService

\* @param UserService $userService

\*/

public function \_\_construct(PostService $postService, UserService $userService)

{

$this->postService = $postService;

$this->userService = $userService;

}

/\*\*

\* Display a listing of the resource.

\*

\* @return Application|Factory|View

\*/

public function index(): View|Factory|Application

{

$posts = Post::orderBy('id', 'desc')->with('user')->paginate(10);

return view('admin.post.index', compact('posts'));

}

/\*\*

\* Show the form for creating a new resource.

\*

\* @return Factory|View|Application

\*/

public function create(): Factory|View|Application

{

return view('admin.post.create');

}

/\*\*

\* Store a newly created resource in storage.

\*

\* @param PostStoreRequest $request

\* @return RedirectResponse

\*/

public function store(PostStoreRequest $request): RedirectResponse

{

$currentUser = $this->userService->getCurrentUser();

if (!$currentUser || !$currentUser->id) {

return redirect()->back()->with('error', \_\_('It is not possible to get user for post creation.'));

}

$post = $this->postService->store($request->validated(), $currentUser);

return redirect()->route('admin.posts.edit', $post)->with('success', \_\_('Post created successfully.'));

}

/\*\*

\* Show the form for editing the specified resource.

\*

\* @param Post $post

\* @return Application|Factory|View

\*/

public function edit(Post $post): View|Factory|Application

{

return view('admin.post.edit', compact('post'));

}

/\*\*

\* Update the specified resource in storage.

\*

\* @param PostUpdateRequest $request

\* @param Post $post

\* @return RedirectResponse

\*/

public function update(PostUpdateRequest $request, Post $post): RedirectResponse

{

$this->postService->update($post, $request->validated());

return redirect()->route('admin.posts.edit', $post)->with('success', \_\_('Post updated successfully.'));

}

/\*\*

\* Remove the specified resource from storage.

\*

\* @param Post $post

\* @return RedirectResponse

\*/

public function destroy(Post $post): RedirectResponse

{

$this->postService->destroy($post);

return redirect()->route('admin.posts.index')->with('success', \_\_('Post deleted successfully.'));

}

}

**app/Http/Controllers/Admin/QuestionController.php:**

<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;

use App\Http\Requests\Admin\Question\QuestionResponseRequest;

use App\Http\Requests\Admin\Question\QuestionUpdateRequest;

use App\Mail\QuestionResponseMail;

use App\Models\Question;

use App\Services\QuestionService;

use Illuminate\Contracts\Foundation\Application;

use Illuminate\Contracts\View\Factory;

use Illuminate\Contracts\View\View;

use Illuminate\Http\RedirectResponse;

use Illuminate\Support\Facades\Mail;

class QuestionController extends Controller

{

/\*\*

\* @var QuestionService

\*/

private QuestionService $questionService;

/\*\*

\* @param QuestionService $questionService

\*/

public function \_\_construct(QuestionService $questionService)

{

$this->questionService = $questionService;

}

/\*\*

\* Display a listing of the resource.

\*

\* @return Application|Factory|View

\*/

public function index(): View|Factory|Application

{

$questions = Question::orderBy('id', 'desc')->paginate(10);

return view('admin.question.index', compact('questions'));

}

/\*\*

\* Show the form for editing the specified resource.

\*

\* @param Question $question

\* @return Application|Factory|View

\*/

public function edit(Question $question): View|Factory|Application

{

return view('admin.question.edit', compact('question'));

}

/\*\*

\* Update the specified resource in storage.

\*

\* @param QuestionUpdateRequest $request

\* @param Question $question

\* @return RedirectResponse

\*/

public function update(QuestionUpdateRequest $request, Question $question): RedirectResponse

{

$this->questionService->update($question, $request->validated());

return redirect()->route('admin.questions.edit', $question)

->with('success', \_\_('Question updated successfully.'));

}

/\*\*

\* Remove the specified resource from storage.

\*

\* @param Question $question

\* @return RedirectResponse

\*/

public function destroy(Question $question): RedirectResponse

{

$this->questionService->destroy($question);

return redirect()->route('admin.questions.index')

->with('success', \_\_('Question deleted successfully.'));

}

public function response( Question $question, QuestionResponseRequest $request): RedirectResponse

{

$response = $request->validated("response");

Mail::to($question->email)

->send(new QuestionResponseMail($question, $response));

return redirect()->back()

->with('success', \_\_('Question answered successfully.'));

}

}

**app/Http/API/InvoiceController.php:**

<?php

namespace App\Http\Controllers\API;

use App\Http\Controllers\Controller;

use App\Http\Requests\Admin\Invoice\InvoiceUpdateRequest;

use App\Http\Requests\API\InvoiceCreateRequest;

use App\Models\Company;

use App\Models\Invoice;

use App\Services\InvoiceService;

use App\Services\MailSendingService;

use DomainException;

use Illuminate\Http\JsonResponse;

use Illuminate\Http\Request;

class InvoiceController extends Controller

{

/\*\*

\* @var InvoiceService

\*/

private InvoiceService $invoiceService;

/\*\*

\* @var MailSendingService

\*/

private MailSendingService $mailSendingService;

/\*\*

\* @var Company|null

\*/

private ?Company $company;

/\*\*

\* @param InvoiceService $invoiceService

\*/

public function \_\_construct(InvoiceService $invoiceService, MailSendingService $mailSendingService)

{

$this->invoiceService = $invoiceService;

$this->mailSendingService = $mailSendingService;

$this->company = $this->getCurrentCompany();

}

/\*\*

\* Display a listing of the resource.

\*/

public function index(): JsonResponse

{

return response()->json(Invoice::where('company\_id', $this->company?->id)->get());

}

/\*\*

\* Store a newly created resource in storage.

\*/

public function store(InvoiceCreateRequest $request)

{

$data = $request->validated();

$data['company\_id'] = $this->company?->id;

$result = $this->invoiceService->store($data);

$this->mailSendingService->sendUnpaidInvoiceMessage($result);

return $result;

}

/\*\*

\* Display the specified resource.

\*/

public function show(string $id)

{

return Invoice::where([

'company\_id' => $this->company?->id,

'id' => $id

])->first();

}

/\*\*

\* Update the specified resource in storage.

\*/

public function update(InvoiceUpdateRequest $request, string $id)

{

$invoice = Invoice::where([

'company\_id' => $this->company?->id,

'id' => $id

])->first();

if (!$invoice) {

return response()->json(['message' => \_\_('Provided invoice id is wrong!')],404);

}

$data = $request->validated();

$data['company\_id'] = $this->company->id;

$this->invoiceService->update($invoice, $data);

}

/\*\*

\* Remove the specified resource from storage.

\*/

public function destroy(string $id)

{

$invoice = Invoice::where([

'company\_id' => $this->company?->id,

'id' => $id

])->first();

if ($invoice) {

$invoice->delete();

}

return response()->json(['message' => \_\_('Invoice has been deleted successfully.')]);

}

/\*\*

\* @return Company|null

\*/

private function getCurrentCompany(): ?Company

{

$apiCode = (string)request()->header('X-API-Code');

return Company::where('api\_code', $apiCode)->first();

}

}

**app/Http/Auth/ConfirmPasswordController.php:**

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Foundation\Auth\ConfirmsPasswords;

class ConfirmPasswordController extends Controller

{

/\*

|--------------------------------------------------------------------------

| Confirm Password Controller

|--------------------------------------------------------------------------

|

| This controller is responsible for handling password confirmations and

| uses a simple trait to include the behavior. You're free to explore

| this trait and override any functions that require customization.

|

\*/

use ConfirmsPasswords;

/\*\*

\* Where to redirect users when the intended url fails.

\*

\* @var string

\*/

protected $redirectTo = RouteServiceProvider::HOME;

/\*\*

\* Create a new controller instance.

\*

\* @return void

\*/

public function \_\_construct()

{

$this->middleware('auth');

}

}

**app/Http/Auth/ForgotPasswordController.php:**

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;

use Illuminate\Foundation\Auth\SendsPasswordResetEmails;

class ForgotPasswordController extends Controller

{

/\*

|--------------------------------------------------------------------------

| Password Reset Controller

|--------------------------------------------------------------------------

|

| This controller is responsible for handling password reset emails and

| includes a trait which assists in sending these notifications from

| your application to your users. Feel free to explore this trait.

|

\*/

use SendsPasswordResetEmails;

}

**app/Http/Auth/ForgotPasswordController.php:**

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Foundation\Auth\AuthenticatesUsers;

class LoginController extends Controller

{

/\*

|--------------------------------------------------------------------------

| Login Controller

|--------------------------------------------------------------------------

|

| This controller handles authenticating users for the application and

| redirecting them to your home screen. The controller uses a trait

| to conveniently provide its functionality to your applications.

|

\*/

use AuthenticatesUsers;

/\*\*

\* Where to redirect users after login.

\*

\* @var string

\*/

protected $redirectTo = RouteServiceProvider::HOME;

/\*\*

\* Create a new controller instance.

\*

\* @return void

\*/

public function \_\_construct()

{

$this->middleware('guest')->except('logout');

}

}

**app/Http/Auth/LoginController.php:**

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Foundation\Auth\AuthenticatesUsers;

class LoginController extends Controller

{

/\*

|--------------------------------------------------------------------------

| Login Controller

|--------------------------------------------------------------------------

|

| This controller handles authenticating users for the application and

| redirecting them to your home screen. The controller uses a trait

| to conveniently provide its functionality to your applications.

|

\*/

use AuthenticatesUsers;

/\*\*

\* Where to redirect users after login.

\*

\* @var string

\*/

protected $redirectTo = RouteServiceProvider::HOME;

/\*\*

\* Create a new controller instance.

\*

\* @return void

\*/

public function \_\_construct()

{

$this->middleware('guest')->except('logout');

}

}

**app/Http/Auth/RegisterController.php:**

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use App\Models\User;

use Illuminate\Foundation\Auth\RegistersUsers;

use Illuminate\Support\Facades\Hash;

use Illuminate\Support\Facades\Validator;

class RegisterController extends Controller

{

/\*

|--------------------------------------------------------------------------

| Register Controller

|--------------------------------------------------------------------------

|

| This controller handles the registration of new users as well as their

| validation and creation. By default this controller uses a trait to

| provide this functionality without requiring any additional code.

|

\*/

use RegistersUsers;

/\*\*

\* Where to redirect users after registration.

\*

\* @var string

\*/

protected $redirectTo = RouteServiceProvider::HOME;

/\*\*

\* Create a new controller instance.

\*

\* @return void

\*/

public function \_\_construct()

{

$this->middleware('guest');

}

/\*\*

\* Get a validator for an incoming registration request.

\*

\* @param array $data

\* @return \Illuminate\Contracts\Validation\Validator

\*/

protected function validator(array $data)

{

return Validator::make($data, [

'name' => ['required', 'string', 'max:255'],

'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],

'password' => ['required', 'string', 'min:8', 'confirmed'],

]);

}

/\*\*

\* Create a new user instance after a valid registration.

\*

\* @param array $data

\* @return \App\Models\User

\*/

protected function create(array $data)

{

return User::create([

'name' => $data['name'],

'email' => $data['email'],

'password' => Hash::make($data['password']),

]);

}

}

**app/Http/Auth/ResetPasswordController.php:**

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Foundation\Auth\ResetsPasswords;

class ResetPasswordController extends Controller

{

/\*

|--------------------------------------------------------------------------

| Password Reset Controller

|--------------------------------------------------------------------------

|

| This controller is responsible for handling password reset requests

| and uses a simple trait to include this behavior. You're free to

| explore this trait and override any methods you wish to tweak.

|

\*/

use ResetsPasswords;

/\*\*

\* Where to redirect users after resetting their password.

\*

\* @var string

\*/

protected $redirectTo = RouteServiceProvider::HOME;

}

**app/Http/Auth/VerificationController.php:**

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Foundation\Auth\VerifiesEmails;

class VerificationController extends Controller

{

/\*

|--------------------------------------------------------------------------

| Email Verification Controller

|--------------------------------------------------------------------------

|

| This controller is responsible for handling email verification for any

| user that recently registered with the application. Emails may also

| be re-sent if the user didn't receive the original email message.

|

\*/

use VerifiesEmails;

/\*\*

\* Where to redirect users after verification.

\*

\* @var string

\*/

protected $redirectTo = RouteServiceProvider::HOME;

/\*\*

\* Create a new controller instance.

\*

\* @return void

\*/

public function \_\_construct()

{

$this->middleware('auth');

$this->middleware('signed')->only('verify');

$this->middleware('throttle:6,1')->only('verify', 'resend');

}

}

**app/Http/Company/DashboardController.php:**

<?php

namespace App\Http\Controllers\Company;

use App\Http\Controllers\Controller;

use App\Models\Company;

use App\Models\Invoice;

use App\Services\InvoiceService;

use Illuminate\Contracts\Foundation\Application;

use Illuminate\Contracts\Routing\ResponseFactory;

use Illuminate\Http\Response;

class DashboardController extends Controller

{

private InvoiceService $invoiceService;

/\*\*

\* @param InvoiceService $invoiceService

\*/

public function \_\_construct(InvoiceService $invoiceService)

{

$this->invoiceService = $invoiceService;

}

public function index()

{

$invoices = Invoice::orderBy('id', 'desc')->where('company\_id', $this->getCompanyId())->paginate(10);

return view('companies-admin.invoices.index', compact('invoices'));

}

public function show(Invoice $invoice)

{

if ((int)$invoice->company\_id !== $this->getCompanyId()) {

abort(403);

}

$companies = Company::where('id', $this->getCompanyId())->get();

$clientTypes = Invoice::getClientTypes();

return view('companies-admin.invoices.show', compact('invoice', 'companies', 'clientTypes'));

}

private function getCompanyId(): int

{

return (int)auth()->guard('company')->user()->id;

}

/\*\*

\* @param Invoice $invoice

\* @return Response|Application|ResponseFactory

\*/

public function showPdf(Invoice $invoice): Response|Application|ResponseFactory

{

if ((int)$invoice->company\_id !== $this->getCompanyId()) {

abort(403);

}

return response($this->invoiceService->show($invoice))->header('Content-Type', 'application/pdf');

}

}

**app/Http/CompanyAuth/ConfirmPasswordController.php:**

<?php

namespace App\Http\Controllers\CompanyAuth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Foundation\Auth\ConfirmsPasswords;

class ConfirmPasswordController extends Controller

{

/\*

|--------------------------------------------------------------------------

| Confirm Password Controller

|--------------------------------------------------------------------------

|

| This controller is responsible for handling password confirmations and

| uses a simple trait to include the behavior. You're free to explore

| this trait and override any functions that require customization.

|

\*/

use ConfirmsPasswords;

/\*\*

\* Where to redirect users when the intended url fails.

\*

\* @var string

\*/

protected $redirectTo = RouteServiceProvider::HOME;

/\*\*

\* Create a new controller instance.

\*

\* @return void

\*/

public function \_\_construct()

{

$this->middleware('company.auth');

}

}

**app/Http/CompanyAuth/ForgotPasswordController.php:**

<?php

namespace App\Http\Controllers\CompanyAuth;

use App\Http\Controllers\Controller;

use Illuminate\Contracts\Auth\PasswordBroker;

use Illuminate\Contracts\View\Factory;

use Illuminate\Foundation\Application;

use Illuminate\Foundation\Auth\SendsPasswordResetEmails;

use Illuminate\Support\Facades\Password;

use Illuminate\Support\Facades\View;

class ForgotPasswordController extends Controller

{

/\*

|--------------------------------------------------------------------------

| Password Reset Controller

|--------------------------------------------------------------------------

|

| This controller is responsible for handling password reset emails and

| includes a trait which assists in sending these notifications from

| your application to your users. Feel free to explore this trait.

|

\*/

use SendsPasswordResetEmails;

/\*\*

\* Create a new controller instance.

\*

\* @return void

\*/

public function \_\_construct()

{

$this->middleware('company.guest');

}

public function showLinkRequestForm(): Factory|Application|\Illuminate\Contracts\View\View|\Illuminate\Contracts\Foundation\Application

{

return view('companies-auth.passwords.email');

}

/\*\*

\* Get the broker to be used during password reset.

\*

\* @return PasswordBroker

\*/

public function broker(): PasswordBroker

{

return Password::broker('companies');

}

}

**app/Http/CompanyAuth/LoginController.php:**

<?php

namespace App\Http\Controllers\CompanyAuth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Contracts\Auth\StatefulGuard;

use Illuminate\Foundation\Auth\AuthenticatesUsers;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

use Illuminate\View\View;

class LoginController extends Controller

{

/\*

|--------------------------------------------------------------------------

| Login Controller

|--------------------------------------------------------------------------

|

| This controller handles authenticating users for the application and

| redirecting them to your home screen. The controller uses a trait

| to conveniently provide its functionality to your applications.

|

\*/

use AuthenticatesUsers;

/\*\*

\* Where to redirect users after login.

\*

\* @var string

\*/

protected string $redirectTo = RouteServiceProvider::HOME;

/\*\*

\* Create a new controller instance.

\*

\* @return void

\*/

public function \_\_construct()

{

$this->middleware('company.guest')->except('logout');

}

/\*\*

\* Show the application's login form.

\*

\* @return View

\*/

public function showLoginForm(): View

{

return view('companies-auth.login');

}

/\*\*

\* @return string

\*/

public function redirectTo(): string

{

return route('companies-admin.index');

}

/\*\*

\* Get the guard to be used during authentication.

\*

\* @return StatefulGuard

\*/

protected function guard(): StatefulGuard

{

return Auth::guard($this->getGuardName());

}

/\*\*

\* @return string

\*/

private function getGuardName(): string

{

return 'company';

}

protected function loggedOut(Request $request)

{

return redirect()->route('companies.login');

}

}

**app/Http/CompanyAuth/ResetPasswordController.php:**

<?php

namespace App\Http\Controllers\CompanyAuth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Contracts\Auth\PasswordBroker;

use Illuminate\Contracts\Auth\StatefulGuard;

use Illuminate\Foundation\Auth\ResetsPasswords;

use Illuminate\Contracts\View\Factory;

use Illuminate\Contracts\View\View;

use Illuminate\Foundation\Application;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

use Illuminate\Support\Facades\Password;

class ResetPasswordController extends Controller

{

/\*

|--------------------------------------------------------------------------

| Password Reset Controller

|--------------------------------------------------------------------------

|

| This controller is responsible for handling password reset requests

| and uses a simple trait to include this behavior. You're free to

| explore this trait and override any methods you wish to tweak.

|

\*/

use ResetsPasswords;

/\*\*

\* Where to redirect users after resetting their password.

\*

\* @var string

\*/

protected $redirectTo = RouteServiceProvider::HOME;

/\*\*

\* @param Request $request

\* @return \Illuminate\Contracts\Foundation\Application|Factory|View|Application

\*/

public function showResetForm(Request $request): Application|View|Factory|\Illuminate\Contracts\Foundation\Application

{

$token = $request->route()->parameter('token');

return view('companies-auth.passwords.reset')->with(

['token' => $token, 'email' => $request->email]

);

}

/\*\*

\* Get the broker to be used during password reset.

\*

\* @return PasswordBroker

\*/

public function broker(): PasswordBroker

{

return Password::broker('companies');

}

/\*\*

\* Get the post register / login redirect path.

\*

\* @return string

\*/

public function redirectTo(): string

{

return route('companies-admin.index');

}

/\*\*

\* Get the guard to be used during password reset.

\*

\* @return StatefulGuard

\*/

protected function guard(): StatefulGuard

{

return Auth::guard('company');

}

}

**app/Http/Controllers/BlogController.php:**

<?php

namespace App\Http\Controllers;

use App\Models\Post;

use Illuminate\Contracts\Foundation\Application;

use Illuminate\Contracts\View\Factory;

use Illuminate\Contracts\View\View;

class BlogController extends Controller

{

/\*\*

\* @param Post $post

\* @return Factory|View|Application

\*/

public function show(Post $post): Factory|View|Application

{

if (!$post->active) {

abort(404);

}

return view('blog.show', compact('post'));

}

}

**app/Http/Controllers/ContactsController.php:**

<?php

namespace App\Http\Controllers;

use App\Http\Requests\QuestionRequest;

use App\Services\QuestionService;

use DomainException;

use Illuminate\Contracts\Foundation\Application;

use Illuminate\Contracts\View\Factory;

use Illuminate\Contracts\View\View;

use Illuminate\Http\RedirectResponse;

class ContactsController extends Controller

{

private QuestionService $questionService;

/\*\*

\* @param QuestionService $questionService

\*/

public function \_\_construct(QuestionService $questionService)

{

$this->questionService = $questionService;

}

/\*\*

\* @return Factory|View|Application

\*/

public function index(): Factory|View|Application

{

return view('contacts.form');

}

/\*\*

\* @param QuestionRequest $questionRequest

\* @return RedirectResponse

\*/

public function submit(QuestionRequest $questionRequest): RedirectResponse

{

try {

$this->questionService->create($questionRequest->validated(), $questionRequest->ip(), false);

} catch (DomainException $e) {

return redirect()->back()->withInput()->with(['error' => $e->getMessage()]);

}

return redirect()->back()->with(['success' => \_\_('Your question has been submitted successfully.')]);

}

}

**app/Http/Controllers/Controller.php:**

<?php

namespace App\Http\Controllers;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;

use Illuminate\Foundation\Validation\ValidatesRequests;

use Illuminate\Routing\Controller as BaseController;

class Controller extends BaseController

{

use AuthorizesRequests, ValidatesRequests;

}

**app/Http/Controllers/HomeController.php:**

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class HomeController extends Controller

{

/\*\*

\* Create a new controller instance.

\*

\* @return void

\*/

public function \_\_construct()

{

$this->middleware('auth');

}

/\*\*

\* Show the application dashboard.

\*

\* @return \Illuminate\Contracts\Support\Renderable

\*/

public function index()

{

return view('admin-layout');

}

}

**app/Http/Controllers/MailController.php:**

<?php

namespace App\Http\Controllers;

use App\Models\Company;

use App\Models\Invoice;

use App\Models\Post;

use Illuminate\Contracts\View\Factory;

use Illuminate\Contracts\View\View;

use Illuminate\Foundation\Application;

use Illuminate\Http\Request;

class MainController extends Controller{

public function main(): View|Application|Factory|\Illuminate\Contracts\Foundation\Application

{

$posts = Post::orderBy('id', 'desc')->where('active', true)->with('user')->paginate(9);

$companiesQuantity = Company::all()->count();

$paymentsQuantity = Invoice::where('paid', true)->count();

return view('main', compact('posts', 'companiesQuantity', 'paymentsQuantity'));

}

public function about(): View|Application|Factory|\Illuminate\Contracts\Foundation\Application

{

return view('about');

}

public function services(): View|Application|Factory|\Illuminate\Contracts\Foundation\Application

{

return view('services');

}

public function Account\_management(): View|Application|Factory|\Illuminate\Contracts\Foundation\Application

{

return view('Account\_management');

}

public function Managing\_administrators(): View|Application|Factory|\Illuminate\Contracts\Foundation\Application

{

return view('Managing\_administrators');

}

public function Company\_management(): View|Application|Factory|\Illuminate\Contracts\Foundation\Application

{

return view('Company\_management');

}

}

**app/Http/Controllers/PaymentController.php:**

<?php

namespace App\Http\Controllers;

use App\Models\Invoice;

use App\Services\StripeService;

use DomainException;

use Illuminate\Contracts\View\Factory;

use Illuminate\Contracts\View\View;

use Illuminate\Foundation\Application;

use Illuminate\Http\RedirectResponse;

class PaymentController extends Controller

{

private StripeService $stripeService;

/\*\*

\* @param StripeService $stripeService

\*/

public function \_\_construct(StripeService $stripeService)

{

$this->stripeService = $stripeService;

}

/\*\*

\* @param Invoice $invoice

\* @return \Illuminate\Contracts\Foundation\Application|Factory|View|Application

\*/

public function showPaymentForm(Invoice $invoice): Application|View|Factory|\Illuminate\Contracts\Foundation\Application

{

if ($invoice->paid) {

return view('payment.invoice-paid', compact('invoice'));

} else {

return view('payment.payment-form', compact('invoice'));

}

}

/\*\*

\* @param Invoice $invoice

\* @return RedirectResponse

\*/

public function executePayment(Invoice $invoice): RedirectResponse

{

if ($invoice->paid) {

return redirect()->route('payment.show-form', $invoice)

->with('error', \_\_('Invoice has been already paid'));

}

try {

$redirectUrl = $this->stripeService->getRedirectUrl($invoice);

} catch (DomainException $e) {

return redirect()->route('payment.show-form', $invoice)->with('error', $e->getMessage());

}

return redirect()->to($redirectUrl);

}

public function finish(Invoice $invoice)

{

try {

$this->stripeService->finalizeStripePayment($invoice);

} catch (DomainException $e) {

return redirect()->route('payment.show-form', $invoice)->with('error', $e->getMessage());

}

if (!$invoice->paid) {

return redirect()->route('payment.show-form', $invoice)

->with('error', \_\_('Payment is not accepted'));

}

return redirect()->route('payment.show-form', $invoice);

}

/\*\*

\* @param Invoice $invoice

\* @return RedirectResponse

\*/

public function cancel(Invoice $invoice): RedirectResponse

{

return redirect()->route('payment.show-form', $invoice);

}

}

**app/Http/Middleware/Company/Authenticate.php:**

<?php

namespace App\Http\Middleware\Company;

use Closure;

use Illuminate\Auth\AuthenticationException;

use Illuminate\Auth\Middleware\Authenticate as Middleware;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

class Authenticate extends Middleware

{

/\*\*

\* Handle an incoming request.

\*

\* @param Request $request

\* @param Closure $next

\* @param string[] ...$guards

\* @return mixed

\*

\* @throws AuthenticationException

\*/

public function handle($request, Closure $next, ...$guards): mixed

{

$guard = 'company';

if (!Auth::guard($guard)->check()) {

throw new AuthenticationException(\_\_('Unauthenticated.'), [$guard], $this->redirectTo($request));

}

return $next($request);

}

/\*\*

\* Get the path the user should be redirected to when they are not authenticated.

\*/

protected function redirectTo(Request $request): ?string

{

return $request->expectsJson() ? null : route('companies.login');

}

}

**app/Http/Middleware/Company/RedirectIfAuthenticated.php:**

<?php

namespace App\Http\Middleware\Company;

use App\Providers\RouteServiceProvider;

use Closure;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

use Symfony\Component\HttpFoundation\Response;

class RedirectIfAuthenticated

{

/\*\*

\* Handle an incoming request.

\*

\* @param Closure(Request): (Response) $next

\*/

public function handle(Request $request, Closure $next): Response

{

if (Auth::guard('company')->check()) {

return redirect()->route('companies-admin.index');

}

return $next($request);

}

}

**app/Http/Middleware/APIAuthorizationMiddleware.php:**

<?php

namespace App\Http\Middleware;

use App\Models\Company;

use Closure;

use Illuminate\Http\Request;

use Symfony\Component\HttpFoundation\Response;

class APIAuthorizationMiddleware

{

/\*\*

\* Handle an incoming request.

\*

\* @param \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response) $next

\*/

public function handle(Request $request, Closure $next): Response

{

$apiCode = trim($request->header('X-API-Code'));

if ($apiCode === '' || Company::where('api\_code', $apiCode)->count() === 0) {

return response()->json(['error' => \_\_('Wrong or empty X-API-Code has been provided.')],401);

}

return $next($request);

}

}

**app/Http/Middleware/Authenticate.php:**

<?php

namespace App\Http\Middleware;

use Illuminate\Auth\Middleware\Authenticate as Middleware;

use Illuminate\Http\Request;

class Authenticate extends Middleware

{

/\*\*

\* Get the path the user should be redirected to when they are not authenticated.

\*/

protected function redirectTo(Request $request): ?string

{

return $request->expectsJson() ? null : route('login');

}

}

**app/Http/Middleware/EncryptCookies.php:**

<?php

namespace App\Http\Middleware;

use Illuminate\Cookie\Middleware\EncryptCookies as Middleware;

class EncryptCookies extends Middleware

{

/\*\*

\* The names of the cookies that should not be encrypted.

\*

\* @var array<int, string>

\*/

protected $except = [

//

];

}

**app/Http/Middleware/PreventRequestsDuringMaintenance.php:**

<?php

namespace App\Http\Middleware;

use Illuminate\Foundation\Http\Middleware\PreventRequestsDuringMaintenance as Middleware;

class PreventRequestsDuringMaintenance extends Middleware

{

/\*\*

\* The URIs that should be reachable while maintenance mode is enabled.

\*

\* @var array<int, string>

\*/

protected $except = [

//

];

}

**app/Http/Middleware/RedirectIfAuthenticated.php:**

<?php

namespace App\Http\Middleware;

use App\Providers\RouteServiceProvider;

use Closure;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

use Symfony\Component\HttpFoundation\Response;

class RedirectIfAuthenticated

{

/\*\*

\* Handle an incoming request.

\*

\* @param \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response) $next

\*/

public function handle(Request $request, Closure $next, string ...$guards): Response

{

$guards = empty($guards) ? [null] : $guards;

foreach ($guards as $guard) {

if (Auth::guard($guard)->check()) {

return redirect(RouteServiceProvider::HOME);

}

}

return $next($request);

}

}

**app/Http/Middleware/TrimStrings.php:**

<?php

namespace App\Http\Middleware;

use Illuminate\Foundation\Http\Middleware\TrimStrings as Middleware;

class TrimStrings extends Middleware

{

/\*\*

\* The names of the attributes that should not be trimmed.

\*

\* @var array<int, string>

\*/

protected $except = [

'current\_password',

'password',

'password\_confirmation',

];

}

**app/Http/Middleware/TrustHosts.php:**

<?php

namespace App\Http\Middleware;

use Illuminate\Http\Middleware\TrustHosts as Middleware;

class TrustHosts extends Middleware

{

/\*\*

\* Get the host patterns that should be trusted.

\*

\* @return array<int, string|null>

\*/

public function hosts(): array

{

return [

$this->allSubdomainsOfApplicationUrl(),

];

}

}

**app/Http/Middleware/TrustProxies.php:**

<?php

namespace App\Http\Middleware;

use Illuminate\Http\Middleware\TrustProxies as Middleware;

use Illuminate\Http\Request;

class TrustProxies extends Middleware

{

/\*\*

\* The trusted proxies for this application.

\*

\* @var array<int, string>|string|null

\*/

protected $proxies;

/\*\*

\* The headers that should be used to detect proxies.

\*

\* @var int

\*/

protected $headers =

Request::HEADER\_X\_FORWARDED\_FOR |

Request::HEADER\_X\_FORWARDED\_HOST |

Request::HEADER\_X\_FORWARDED\_PORT |

Request::HEADER\_X\_FORWARDED\_PROTO |

Request::HEADER\_X\_FORWARDED\_AWS\_ELB;

}

**app/Http/Middleware/ValidateSignature.php:**

<?php

namespace App\Http\Middleware;

use Illuminate\Routing\Middleware\ValidateSignature as Middleware;

class ValidateSignature extends Middleware

{

/\*\*

\* The names of the query string parameters that should be ignored.

\*

\* @var array<int, string>

\*/

protected $except = [

// 'fbclid',

// 'utm\_campaign',

// 'utm\_content',

// 'utm\_medium',

// 'utm\_source',

// 'utm\_term',

];

}

**app/Http/Middleware/VerifyCsrfToken.php:**

<?php

namespace App\Http\Middleware;

use Illuminate\Foundation\Http\Middleware\VerifyCsrfToken as Middleware;

class VerifyCsrfToken extends Middleware

{

/\*\*

\* The URIs that should be excluded from CSRF verification.

\*

\* @var array<int, string>

\*/

protected $except = [

//

];

}

**app/Http/Requests/Admin/Company/CompanyCreateRequest.php:**

<?php

namespace App\Http\Requests\Admin\Company;

use Illuminate\Foundation\Http\FormRequest;

class CompanyCreateRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*/

public function authorize(): bool

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array<string, \Illuminate\Contracts\Validation\ValidationRule|array<mixed>|string>

\*/

public function rules(): array

{

return [

'name' => 'required|string|min:3|max:255',

'reg\_number' => 'required|string|min:3|max:255|unique:companies,reg\_number',

'vat\_number' => 'required|string|min:3|max:255|unique:companies,vat\_number',

'address' => 'required|string|min:3|max:255',

'email' => 'required|string|email|min:3|max:255|unique:companies,email',

'phone' => 'required|string|min:3|max:255|unique:companies,phone',

'description' => 'nullable|string|max:9999',

'api\_code' => 'nullable|string|max:9999',

'password' => 'required|string|min:8|max:100',

];

}

}

**app/Http/Requests/Admin/Company/CompanyUpdateRequest.php:**

<?php

namespace App\Http\Requests\Admin\Company;

use Illuminate\Foundation\Http\FormRequest;

class CompanyUpdateRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*/

public function authorize(): bool

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array<string, \Illuminate\Contracts\Validation\ValidationRule|array<mixed>|string>

\*/

public function rules(): array

{

$companyId = isset($this->company->id) ? (int) $this->company->id : 0;

return [

'name' => 'required|string|min:3|max:255',

'reg\_number' => 'required|string|min:3|max:255|unique:companies,reg\_number,' . $companyId ,

'vat\_number' => 'required|string|min:3|max:255|unique:companies,vat\_number,' . $companyId ,

'address' => 'required|string|min:3|max:255',

'email' => 'required|string|email|min:3|max:255|unique:companies,email,' . $companyId ,

'phone' => 'required|string|min:3|max:255|unique:companies,phone,' . $companyId ,

'description' => 'nullable|string|max:9999',

'api\_code' => 'nullable|string|max:9999',

'password' => 'nullable|string|min:8|max:100',

];

}

}

**app/Http/Requests/Admin/Invoice/InvoiceStoreRequest.php:**

<?php

namespace App\Http\Requests\Admin\Invoice;

use App\Models\Invoice;

use Illuminate\Foundation\Http\FormRequest;

class InvoiceStoreRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*

\* @return bool

\*/

public function authorize()

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array<string, mixed>

\*/

public function rules()

{

$clientTypes = implode(',', array\_keys(Invoice::getClientTypes()));

$clientType = !empty($this->client\_type) && is\_numeric($this->client\_type) ? (int) $this->client\_type : 0;

$rules = [

'company\_id' => 'required|int|exists:companies,id',

'tax\_rate' => 'required|numeric|min:0|max:100',

'client\_type' => 'required|int|in:' . $clientTypes,

'client\_name' => 'required|string|max:255',

'client\_email' => 'required|string|email|max:255',

'client\_phone' => 'required|string|max:255',

'services' => 'required|array',

'services.\*.name' => 'required|string|max:255',

'services.\*.price' => 'required|numeric|min:0|max:9999999999',

];

if ($clientType === Invoice::CLIENT\_TYPE\_LEGAL\_ENTITY) {

$rules = array\_merge($rules, [

'client\_company' => 'required|string|max:255',

'client\_vat\_number' => 'required|string|max:255',

'client\_reg\_number' => 'required|string|max:255',

'client\_iban' => 'required|string|max:255',

'client\_swift' => 'required|string|max:255',

]);

}

return $rules;

}

public function attributes()

{

return [

'services' => \_\_('services'),

'services.\*.name' => \_\_('service name'),

'services.\*.price' => \_\_('service price'),

];

}

}

**app/Http/Requests/Admin/Invoice/InvoiceUpdateRequest.php:**

<?php

namespace App\Http\Requests\Admin\Invoice;

use App\Models\Invoice;

use Illuminate\Foundation\Http\FormRequest;

class InvoiceUpdateRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*

\* @return bool

\*/

public function authorize()

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array<string, mixed>

\*/

public function rules()

{

$clientTypes = implode(',', array\_keys(Invoice::getClientTypes()));

$clientType = !empty($this->client\_type) && is\_numeric($this->client\_type) ? (int) $this->client\_type : 0;

$rules = [

'company\_id' => 'required|int|exists:companies,id',

'tax\_rate' => 'required|numeric|min:0|max:100',

'client\_type' => 'required|int|in:' . $clientTypes,

'client\_name' => 'required|string|max:255',

'client\_email' => 'required|string|email|max:255',

'client\_phone' => 'required|string|max:255',

'services' => 'required|array',

'services.\*.name' => 'required|string|max:255',

'services.\*.price' => 'required|numeric|min:0|max:9999999999',

];

if ($clientType === Invoice::CLIENT\_TYPE\_LEGAL\_ENTITY) {

$rules = array\_merge($rules, [

'client\_company' => 'required|string|max:255',

'client\_vat\_number' => 'required|string|max:255',

'client\_reg\_number' => 'required|string|max:255',

'client\_iban' => 'required|string|max:255',

'client\_swift' => 'required|string|max:255',

]);

}

return $rules;

}

public function attributes()

{

return [

'services' => \_\_('services'),

'services.\*.name' => \_\_('service name'),

'services.\*.price' => \_\_('service price')

];

}

}

**app/Http/Requests/Admin/Post/BasicPostRequest.php:**

<?php

namespace App\Http\Requests\Admin\Post;

use Illuminate\Foundation\Http\FormRequest;

abstract class BasicPostRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*

\* @return bool

\*/

public function authorize()

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array<string, mixed>

\*/

public function rules()

{

return [

'title' => ['required', 'string', 'max:255'],

'text' => ['required', 'string', 'max:99999'],

'image' => $this->getImageRule(),

'active' => ['required', 'boolean'],

'slug' => $this->getSlugRule(),

];

}

public function attributes()

{

return [

'title' => \_\_('title'),

'slug' => \_\_('slug'),

'image' => \_\_('image'),

'text' => \_\_('text'),

];

}

/\*\*

\* @return string[]

\*/

abstract protected function getSlugRule(): array;

/\*\*

\* @return string[]

\*/

abstract protected function getImageRule(): array;

}

**app/Http/Requests/Admin/Post/BasicPostSaveRequest.php:**

<?php

namespace App\Http\Requests\Admin\Post;

use Illuminate\Foundation\Http\FormRequest;

abstract class BasicPostSaveRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*

\* @return bool

\*/

public function authorize()

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array<string, mixed>

\*/

public function rules()

{

return [

'title' => ['required', 'string', 'max:255'],

'text' => ['required', 'string', 'max:99999'],

'active' => ['required', 'boolean'],

'slug' => $this->getSlugRule()

];

}

public function attributes()

{

return [

'title' => \_\_('title'),

'slug' => \_\_('slug'),

'text' => \_\_('text'),

];

}

/\*\*

\* @return string[]

\*/

abstract protected function getSlugRule(): array;

}

**app/Http/Requests/Admin/Post/PostStoreRequest.php:**

<?php

namespace App\Http\Requests\Admin\Post;

class PostStoreRequest extends BasicPostRequest

{

/\*\*

\* @return string[]

\*/

public function getSlugRule(): array

{

return ['required', 'string', 'max:255', 'unique:posts', 'regex:/[a-z0-9\_-]+/'];

}

/\*\*

\* @return string[]

\*/

protected function getImageRule(): array

{

return ['required', 'image', 'mimes:jpg,png,jpeg|max:2048'];

}

}

**app/Http/Requests/Admin/Post/PostStoreSaveRequest.php:**

<?php

namespace App\Http\Requests\Admin\Post;

class PostStoreSaveRequest extends BasicPostSaveRequest

{

/\*\*

\* @return string[]

\*/

public function getSlugRule(): array

{

return ['required', 'string', 'max:255', 'unique:posts', 'regex:/[a-z0-9\_-]+/'];

}

}

**app/Http/Requests/Admin/Post/PostUpdateRequest.php:**

<?php

namespace App\Http\Requests\Admin\Post;

use App\Models\Post;

class PostUpdateRequest extends BasicPostRequest

{

/\*\*

\* @return string[]

\*/

public function getSlugRule(): array

{

$postId = isset($this->post) && $this->post instanceOf Post ? (int)$this->post->id : 0;

return ['required', 'string', 'max:255', 'unique:posts,slug,' . $postId, 'regex:/[a-z0-9\_-]+/'];

}

/\*\*

\* @return string[]

\*/

protected function getImageRule(): array

{

return ['nullable', 'image', 'mimes:jpg,png,jpeg|max:2048'];

}

}

**app/Http/Requests/Admin/Post/PostUpdateSaveRequest.php:**

<?php

namespace App\Http\Requests\Admin\Post;

use App\Models\Post;

class PostUpdateSaveRequest extends BasicPostSaveRequest

{

/\*\*

\* @return string[]

\*/

public function getSlugRule(): array

{

$postId = isset($this->post) && $this->post instanceOf Post ? (int)$this->post->id : 0;

return ['required', 'string', 'max:255', 'unique:posts,slug,' . $postId, 'regex:/[a-z0-9\_-]+/'];

}

}

**app/Http/Requests/Admin/Question/QuestionResponseRequest.php:**

<?php

namespace App\Http\Requests\Admin\Question;

use Illuminate\Foundation\Http\FormRequest;

class QuestionResponseRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*/

public function authorize(): bool

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array<string, \Illuminate\Contracts\Validation\ValidationRule|array<mixed>|string>

\*/

public function rules(): array

{

return [

"response" => "required|string|min:3|max:2000"

];

}

}

**app/Http/Requests/Admin/Question/QuestionUpdateRequest.php:**

<?php

namespace App\Http\Requests\Admin\Question;

use Illuminate\Foundation\Http\FormRequest;

class QuestionUpdateRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*

\* @return bool

\*/

public function authorize()

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array<string, mixed>

\*/

public function rules()

{

return ['verified' => ['required', 'boolean']];

}

/\*\*

\* @return array

\*/

public function attributes()

{

return ['verified' => \_\_('verified')];

}

}

**app/Http/Requests/Admin/User/BasicUserSaveRequest.php:**

<?php

namespace App\Http\Requests\Admin\User;

use Illuminate\Foundation\Http\FormRequest;

abstract class BasicUserSaveRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*

\* @return bool

\*/

public function authorize()

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array<string, mixed>

\*/

public function rules()

{

return [

'name' => ['required', 'string', 'max:255'],

'email' => $this->getEmailRule(),

'password' => $this->getPasswordRule(),

];

}

public function attributes()

{

return [

'name' => \_\_('name'),

'email' => \_\_('email'),

'password' => \_\_('password')

];

}

/\*\*

\* @return string[]

\*/

abstract protected function getEmailRule(): array;

/\*\*

\* @return string[]

\*/

abstract protected function getPasswordRule(): array;

}

**app/Http/Requests/Admin/User/UserStoreRequest.php:**

<?php

namespace App\Http\Requests\Admin\User;

class UserStoreRequest extends BasicUserSaveRequest

{

/\*\*

\* @return string[]

\*/

protected function getEmailRule(): array

{

return ['required', 'string', 'email', 'max:255', 'unique:users'];

}

/\*\*

\* @return string[]

\*/

protected function getPasswordRule(): array

{

return ['required', 'string', 'min:8', 'confirmed'];

}

}

**app/Http/Requests/Admin/User/UserUpdateRequest.php:**

<?php

namespace App\Http\Requests\Admin\User;

use App\Models\User;

class UserUpdateRequest extends BasicUserSaveRequest

{

/\*\*

\* @return string[]

\*/

protected function getEmailRule(): array

{

$userId = isset($this->user) && $this->user instanceOf User ? (int)$this->user->id : 0;

return ['required', 'string', 'email', 'max:255', 'unique:users,email,'. $userId];

}

/\*\*

\* @return string[]

\*/

protected function getPasswordRule(): array

{

return ['nullable', 'string', 'min:8', 'confirmed'];

}

}

**app/Http/Requests/API/InvoiceCreateRequest.php:**

<?php

namespace App\Http\Requests\API;

use App\Models\Invoice;

use Illuminate\Foundation\Http\FormRequest;

class InvoiceCreateRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*

\* @return bool

\*/

public function authorize()

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array<string, mixed>

\*/

public function rules()

{

$clientTypes = implode(',', array\_keys(Invoice::getClientTypes()));

$clientType = !empty($this->client\_type) && is\_numeric($this->client\_type) ? (int) $this->client\_type : 0;

$rules = [

'tax\_rate' => 'required|numeric|min:0|max:100',

'client\_type' => 'required|int|in:' . $clientTypes,

'client\_name' => 'required|string|max:255',

'client\_email' => 'required|string|email|max:255',

'client\_phone' => 'required|string|max:255',

'services' => 'required|array',

'services.\*.name' => 'required|string|max:255',

'services.\*.price' => 'required|numeric|min:0|max:9999999999',

];

if ($clientType === Invoice::CLIENT\_TYPE\_LEGAL\_ENTITY) {

$rules = array\_merge($rules, [

'client\_company' => 'required|string|max:255',

'client\_vat\_number' => 'required|string|max:255',

'client\_reg\_number' => 'required|string|max:255',

'client\_iban' => 'required|string|max:255',

'client\_swift' => 'required|string|max:255',

]);

}

return $rules;

}

public function attributes()

{

return [

'services' => \_\_('services'),

'services.\*.name' => \_\_('service name'),

'services.\*.price' => \_\_('service price'),

];

}

}

**app/Http/Requests/QuestionRequest.php:**

<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class QuestionRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*

\* @return bool

\*/

public function authorize()

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array<string, mixed>

\*/

public function rules()

{

return [

'email' => ['required', 'email', 'max:255'],

'question\_text' => ['required', 'string', 'max:9999'],

];

}

public function attributes()

{

return [

'email' => \_\_('email'),

'question\_text' => \_\_('question'),

];

}

}

**app/Http/Kernel.php:**

<?php

namespace App\Http;

use App\Http\Middleware\APIAuthorizationMiddleware;

use Illuminate\Foundation\Http\Kernel as HttpKernel;

class Kernel extends HttpKernel

{

/\*\*

\* The application's global HTTP middleware stack.

\*

\* These middleware are run during every request to your application.

\*

\* @var array<int, class-string|string>

\*/

protected $middleware = [

// \App\Http\Middleware\TrustHosts::class,

\App\Http\Middleware\TrustProxies::class,

\Illuminate\Http\Middleware\HandleCors::class,

\App\Http\Middleware\PreventRequestsDuringMaintenance::class,

\Illuminate\Foundation\Http\Middleware\ValidatePostSize::class,

\App\Http\Middleware\TrimStrings::class,

\Illuminate\Foundation\Http\Middleware\ConvertEmptyStringsToNull::class,

];

/\*\*

\* The application's route middleware groups.

\*

\* @var array<string, array<int, class-string|string>>

\*/

protected $middlewareGroups = [

'web' => [

\App\Http\Middleware\EncryptCookies::class,

\Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,

\Illuminate\Session\Middleware\StartSession::class,

\Illuminate\View\Middleware\ShareErrorsFromSession::class,

\App\Http\Middleware\VerifyCsrfToken::class,

\Illuminate\Routing\Middleware\SubstituteBindings::class,

],

'api' => [

// \Laravel\Sanctum\Http\Middleware\EnsureFrontendRequestsAreStateful::class,

\Illuminate\Routing\Middleware\ThrottleRequests::class.':api',

\Illuminate\Routing\Middleware\SubstituteBindings::class,

],

];

/\*\*

\* The application's middleware aliases.

\*

\* Aliases may be used instead of class names to conveniently assign middleware to routes and groups.

\*

\* @var array<string, class-string|string>

\*/

protected $middlewareAliases = [

'auth' => \App\Http\Middleware\Authenticate::class,

'company.auth' => \App\Http\Middleware\Company\Authenticate::class,

'api.auth' => APIAuthorizationMiddleware::class,

'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,

'auth.session' => \Illuminate\Session\Middleware\AuthenticateSession::class,

'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,

'can' => \Illuminate\Auth\Middleware\Authorize::class,

'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,

'company.guest' => \App\Http\Middleware\Company\RedirectIfAuthenticated::class,

'password.confirm' => \Illuminate\Auth\Middleware\RequirePassword::class,

'precognitive' => \Illuminate\Foundation\Http\Middleware\HandlePrecognitiveRequests::class,

'signed' => \App\Http\Middleware\ValidateSignature::class,

'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,

'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,

];

}

**app/Mail/QuestionResponseMail.php:**

<?php

namespace App\Mail;

use App\Models\Question;

use Illuminate\Bus\Queueable;

use Illuminate\Contracts\Queue\ShouldQueue;

use Illuminate\Mail\Mailable;

use Illuminate\Mail\Mailables\Content;

use Illuminate\Mail\Mailables\Envelope;

use Illuminate\Queue\SerializesModels;

class QuestionResponseMail extends Mailable

{

use Queueable, SerializesModels;

/\*\*

\* Create a new message instance.

\*/

public function \_\_construct(private Question $question, private string $response)

{

}

/\*\*

\* Get the message envelope.

\*/

public function envelope(): Envelope

{

return new Envelope(

subject: 'Re: ' . $this->question->question\_text,

);

}

/\*\*

\* Get the message content definition.

\*/

public function content(): Content

{

return new Content(

view: 'mail.reply',

with: ["question"=>$this->question, "response"=>$this->response]

);

}

/\*\*

\* Get the attachments for the message.

\*

\* @return array<int, \Illuminate\Mail\Mailables\Attachment>

\*/

public function attachments(): array

{

return [];

}

}

**app/Mail/UnpaidInvoiceEmail.php:**

<?php

namespace App\Mail;

use App\Models\Invoice;

use App\Services\InvoiceService;

use Illuminate\Bus\Queueable;

use Illuminate\Contracts\Queue\ShouldQueue;

use Illuminate\Mail\Attachment;

use Illuminate\Mail\Mailable;

use Illuminate\Mail\Mailables\Content;

use Illuminate\Mail\Mailables\Envelope;

use Illuminate\Queue\SerializesModels;

class UnpaidInvoiceEmail extends Mailable

{

use Queueable, SerializesModels;

/\*\*

\* Create a new message instance.

\*/

public function \_\_construct(private readonly Invoice $invoice, private readonly InvoiceService $invoiceService)

{

}

/\*\*

\* Get the message envelope.

\*/

public function envelope(): Envelope

{

return new Envelope(

subject: sprintf(\_\_('You have and unpaid invoice %s'), $this->invoice->reference),

);

}

/\*\*

\* Get the message content definition.

\*/

public function content(): Content

{

return new Content(

view: 'mail.unpaid-invoice',

with: ['invoice' => $this->invoice],

);

}

/\*\*

\* Get the attachments for the message.

\*

\* @return array<int, \Illuminate\Mail\Mailables\Attachment>

\*/

public function attachments(): array

{

return [Attachment::fromData(

fn() => $this->invoiceService->show($this->invoice),

sprintf('%s.pdf', $this->invoice->reference)

)];

}

}

**app/Models/Company.php:**

<?php

namespace App\Models;

use App\Notifications\CompanyResetPassword;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Foundation\Auth\User as Authenticatable;

use Illuminate\Notifications\Notifiable;

use Laravel\Sanctum\HasApiTokens;

class Company extends Authenticatable

{

use HasApiTokens, HasFactory, Notifiable;

protected $fillable = [

'name',

'reg\_number',

'vat\_number',

'address',

'description',

'email',

'phone',

'api\_code',

'password',

];

protected $hidden = ['password'];

/\*\*

\* The attributes that should be cast.

\*

\* @var array<string, string>

\*/

protected $casts = [

'password' => 'hashed',

];

/\*\*

\* Send the password reset notification.

\*

\* @param string $token

\* @return void

\*/

public function sendPasswordResetNotification($token): void

{

$this->notify(new CompanyResetPassword($token));

}

}

**app/Models/Invoice.php:**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

use Illuminate\Database\Eloquent\Relations\HasMany;

class Invoice extends Model

{

use HasFactory;

public const CLIENT\_TYPE\_INDIVIDUAL = 1;

public const CLIENT\_TYPE\_LEGAL\_ENTITY = 2;

public const PRICE\_ROUNDING = 2;

protected $fillable = [

'reference',

'company\_id',

'tax\_rate',

'provider\_name',

'provider\_email',

'provider\_phone',

'provider\_company',

'provider\_vat\_number',

'provider\_reg\_number',

'provider\_iban',

'provider\_swift',

'client\_type',

'client\_name',

'client\_email',

'client\_phone',

'client\_company',

'client\_vat\_number',

'client\_reg\_number',

'client\_iban',

'client\_swift',

'paid',

'transaction\_code'

];

/\*\*

\* @return HasMany

\*/

public function invoiceServices(): HasMany

{

return $this->hasMany(InvoiceService::class);

}

public static function getClientTypes(): array

{

return [

self::CLIENT\_TYPE\_INDIVIDUAL => \_\_('Individual'),

self::CLIENT\_TYPE\_LEGAL\_ENTITY => \_\_('Legal Entity'),

];

}

/\*\*

\* @return bool

\*/

public function isLegalEntityInvoice(): bool

{

return (int)$this->client\_type === self::CLIENT\_TYPE\_LEGAL\_ENTITY;

}

public function getTotalPriceWithoutTax(): float

{

$price = 0.0;

foreach ($this->invoiceServices as $service) {

$price += round($service->price, self::PRICE\_ROUNDING);

}

return round($price, self::PRICE\_ROUNDING);

}

public function getTotalPriceWithTax(): float

{

$price = 0.0;

foreach ($this->invoiceServices as $service) {

$price += round($service->price + ($service->price / 100 \* $this->tax\_rate), self::PRICE\_ROUNDING);

}

return round($price, self::PRICE\_ROUNDING);

}

}

**app/Models/InvoiceService.php:**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

use Illuminate\Database\Eloquent\Relations\BelongsTo;

class InvoiceService extends Model

{

use HasFactory;

/\*\*

\* Indicates if the IDs are auto-incrementing.

\*

\* @var bool

\*/

public $incrementing = false;

/\*\*

\* Indicates if the model should be timestamped.

\*

\* @var bool

\*/

public $timestamps = false;

/\*\*

\* The "type" of the primary key ID.

\*

\* @var string

\*/

protected $keyType = 'string';

protected $fillable = [

'id',

'invoice\_id',

'price',

'name'

];

/\*\*

\* @return BelongsTo

\*/

public function invoice(): BelongsTo

{

return $this->belongsTo(Invoice::class);

}

}

**app/Models/Post.php:**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

use Illuminate\Database\Eloquent\Relations\BelongsTo;

/\*\*

\* @property int $id

\* @property string $title

\* @property string $slug

\* @property string $text

\* @property string $image

\* @property ?int $user\_id

\* @property boolean $active

\*

\* @method static Post create(array $args = [])

\* @method static orderBy(...$args)

\*/

class Post extends Model

{

/\*\*

\* @var string[]

\*/

protected $fillable = [

'title',

'text',

'image',

'slug',

'user\_id',

'active',

];

/\*\*

\* @var string[]

\*/

protected $casts = [

'id' => 'integer',

'title' => 'string',

'slug' => 'string',

'text' => 'string',

'image' => 'string',

'active' => 'boolean',

];

/\*\*

\* @return string

\*/

public function getRouteKeyName(): string

{

return 'slug';

}

/\*\*

\* @return BelongsTo

\*/

public function user(): BelongsTo

{

return $this->belongsTo(User::class);

}

/\*\*

\* @return string

\*/

public function getImageFullUrl(): string

{

return sprintf('/storage/images/%s', $this->image);

}

/\*\*

\* @return string

\*/

public function getImageFullPath(): string

{

return storage\_path(sprintf('app/public/images/%s', $this->image));

}

}

**app/Models/Question.php:**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

/\*\*

\* @property int $id

\* @property string $email

\* @property string $question\_text

\* @property string $ip

\* @property boolean $verified

\*

\* @method static Question create(array $args = [])

\* @method static orderBy(...$args)

\* @method static where(...$args)

\*/

class Question extends Model

{

/\*\*

\* @var string[]

\*/

protected $fillable = [

'email',

'question\_text',

'ip',

'verified'

];

/\*\*

\* @var string[]

\*/

protected $casts = [

'id' => 'integer',

'email' => 'string',

'question\_text' => 'string',

'ip' => 'string',

'verified' => 'boolean'

];

}

**app/Models/User.php:**

<?php

namespace App\Models;

// use Illuminate\Contracts\Auth\MustVerifyEmail;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Foundation\Auth\User as Authenticatable;

use Illuminate\Notifications\Notifiable;

use Laravel\Sanctum\HasApiTokens;

class User extends Authenticatable

{

use HasApiTokens, HasFactory, Notifiable;

/\*\*

\* The attributes that are mass assignable.

\*

\* @var array<int, string>

\*/

protected $fillable = [

'name',

'email',

'password',

];

/\*\*

\* The attributes that should be hidden for serialization.

\*

\* @var array<int, string>

\*/

protected $hidden = [

'password',

'remember\_token',

];

/\*\*

\* The attributes that should be cast.

\*

\* @var array<string, string>

\*/

protected $casts = [

'email\_verified\_at' => 'datetime',

'password' => 'hashed',

];

}

**app/Notifications/CompanyResetPassword.php:**

<?php

namespace App\Notifications;

use Illuminate\Bus\Queueable;

use Illuminate\Contracts\Queue\ShouldQueue;

use Illuminate\Notifications\Messages\MailMessage;

use Illuminate\Notifications\Notification;

use Illuminate\Support\Facades\Lang;

class CompanyResetPassword extends Notification

{

use Queueable;

/\*\*

\* Create a new notification instance.

\*/

public function \_\_construct(private readonly string $token)

{

}

/\*\*

\* Get the notification's delivery channels.

\*

\* @return array<int, string>

\*/

public function via(object $notifiable): array

{

return ['mail'];

}

/\*\*

\* Get the mail representation of the notification.

\*/

public function toMail(object $notifiable): MailMessage

{

$url = route('companies.show\_password\_reset\_form', [

'token' => $this->token,

'email' => $notifiable->getEmailForPasswordReset(),

], true);

return (new MailMessage())

->subject(Lang::get('Reset Password Notification'))

->line(Lang::get('We received a password reset request for your account.'))

->action(Lang::get('Reset Password'), $url)

->line(Lang::get('This password reset link will expire in :count minutes.', [

'count' => config('auth.passwords.' . config('auth.defaults.passwords') . '.expire')]))

->line(Lang::get('If you did not request a password reset, no further action is required.'));

}

/\*\*

\* Get the array representation of the notification.

\*

\* @return array<string, mixed>

\*/

public function toArray(object $notifiable): array

{

return [

//

];

}

}

**app/Providers/AppServiceProvider.php:**

<?php

namespace App\Providers;

use Illuminate\Support\ServiceProvider;

class AppServiceProvider extends ServiceProvider

{

/\*\*

\* Register any application services.

\*/

public function register(): void

{

//

}

/\*\*

\* Bootstrap any application services.

\*/

public function boot(): void

{

//

}

}

**app/Providers/AuthServiceProvider.php:**

<?php

namespace App\Providers;

// use Illuminate\Support\Facades\Gate;

use Illuminate\Foundation\Support\Providers\AuthServiceProvider as ServiceProvider;

class AuthServiceProvider extends ServiceProvider

{

/\*\*

\* The model to policy mappings for the application.

\*

\* @var array<class-string, class-string>

\*/

protected $policies = [

//

];

/\*\*

\* Register any authentication / authorization services.

\*/

public function boot(): void

{

//

}

}

**app/Providers/BroadcastServiceProvider.php:**

<?php

namespace App\Providers;

use Illuminate\Support\Facades\Broadcast;

use Illuminate\Support\ServiceProvider;

class BroadcastServiceProvider extends ServiceProvider

{

/\*\*

\* Bootstrap any application services.

\*/

public function boot(): void

{

Broadcast::routes();

require base\_path('routes/channels.php');

}

}

**app/Providers/EventServiceProvider.php:**

<?php

namespace App\Providers;

use Illuminate\Auth\Events\Registered;

use Illuminate\Auth\Listeners\SendEmailVerificationNotification;

use Illuminate\Foundation\Support\Providers\EventServiceProvider as ServiceProvider;

use Illuminate\Support\Facades\Event;

class EventServiceProvider extends ServiceProvider

{

/\*\*

\* The event to listener mappings for the application.

\*

\* @var array<class-string, array<int, class-string>>

\*/

protected $listen = [

Registered::class => [

SendEmailVerificationNotification::class,

],

];

/\*\*

\* Register any events for your application.

\*/

public function boot(): void

{

//

}

/\*\*

\* Determine if events and listeners should be automatically discovered.

\*/

public function shouldDiscoverEvents(): bool

{

return false;

}

}

**app/Providers/RouteServiceProvider.php:**

<?php

namespace App\Providers;

use Illuminate\Cache\RateLimiting\Limit;

use Illuminate\Foundation\Support\Providers\RouteServiceProvider as ServiceProvider;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\RateLimiter;

use Illuminate\Support\Facades\Route;

class RouteServiceProvider extends ServiceProvider

{

/\*\*

\* The path to your application's "home" route.

\*

\* Typically, users are redirected here after authentication.

\*

\* @var string

\*/

public const HOME = '/home';

/\*\*

\* Define your route model bindings, pattern filters, and other route configuration.

\*/

public function boot(): void

{

RateLimiter::for('api', function (Request $request) {

return Limit::perMinute(60)->by($request->user()?->id ?: $request->ip());

});

$this->routes(function () {

Route::middleware('api')

->prefix('api')

->group(base\_path('routes/api.php'));

Route::middleware('web')

->group(base\_path('routes/web.php'));

});

}

}

**app/Services/CompanyService.php:**

<?php

namespace App\Services;

use App\Models\Company;

class CompanyService

{

public function create(array $data): Company

{

return Company::create([

'name' => $data['name'],

'reg\_number' => $data['reg\_number'],

'vat\_number' => $data['vat\_number'],

'address' => $data['address'],

'email' => $data['email'],

'phone' => $data['phone'],

'description' => $data['description'] ?? null,

'api\_code' => $data['api\_code'] ?? null,

'password' => $data['password'] ?? ''

]);

}

public function update(Company $company, array $data): void

{

$dataToUpdate = [

'name' => $data['name'],

'reg\_number' => $data['reg\_number'],

'vat\_number' => $data['vat\_number'],

'address' => $data['address'],

'email' => $data['email'],

'phone' => $data['phone'],

'description' => $data['description'] ?? null,

'api\_code' => $data['api\_code'] ?? null

];

if (isset($data['password']) && $data['password'] !== '') {

$dataToUpdate['password'] = $data['password'];

}

$company->update($dataToUpdate);

}

public function destroy(Company $company): void

{

$company->delete();

}

}

**app/Services/InvoiceService.php:**

<?php

namespace App\Services;

use App\Models\Company;

use App\Models\Invoice;

use App\Models\InvoiceService as Service;

use Dompdf\Dompdf;

use Dompdf\Options;

use Illuminate\Support\Facades\DB;

use Illuminate\Support\Str;

class InvoiceService

{

/\*\*

\* @param array $data

\* @return Invoice

\*/

public function store(array $data): Invoice

{

return DB::transaction(static function () use ($data) : Invoice {

$company = Company::find($data['company\_id']);

$invoice = Invoice::create([

'reference' => Str::uuid(),

'company\_id' => $data['company\_id'],

'tax\_rate' => $data['tax\_rate'],

'provider\_name' => $company->name,

'provider\_email' => $company->email,

'provider\_phone' => $company->phone,

'provider\_company' => $company->name,

'provider\_vat\_number' => $company->vat\_number,

'provider\_reg\_number' => $company->reg\_number,

'provider\_iban' => '',

'provider\_swift' => '',

'client\_type' => $data['client\_type'],

'client\_name' => $data['client\_name'],

'client\_email' => $data['client\_email'],

'client\_phone' => $data['client\_phone'],

'client\_company' => $data['client\_company'] ?? null,

'client\_vat\_number' => $data['client\_vat\_number'] ?? null,

'client\_reg\_number' => $data['client\_reg\_number'] ?? null,

'client\_iban' => $data['client\_iban'] ?? null,

'client\_swift' => $data['client\_swift'] ?? null

]);

foreach ($data['services'] as $serviceData) {

Service::create([

'id' => Str::uuid(),

'invoice\_id' => $invoice->id,

'name' => $serviceData['name'],

'price' => $serviceData['price'],

]);

}

return $invoice;

});

}

/\*\*

\* @param Invoice $invoice

\* @param array $data

\*/

public function update(Invoice $invoice, array $data): void

{

DB::transaction(static function () use ($invoice, $data) : void {

$company = Company::find($data['company\_id']);

$invoice->update([

'company\_id' => $data['company\_id'],

'tax\_rate' => $data['tax\_rate'],

'provider\_name' => $company->name,

'provider\_email' => $company->email,

'provider\_phone' => $company->phone,

'provider\_company' => $company->name,

'provider\_vat\_number' => $company->vat\_number,

'provider\_reg\_number' => $company->reg\_number,

'provider\_iban' => '',

'provider\_swift' => '',

'client\_type' => $data['client\_type'],

'client\_name' => $data['client\_name'],

'client\_email' => $data['client\_email'],

'client\_phone' => $data['client\_phone'],

'client\_company' => $data['client\_company'] ?? null,

'client\_vat\_number' => $data['client\_vat\_number'] ?? null,

'client\_reg\_number' => $data['client\_reg\_number'] ?? null,

'client\_iban' => $data['client\_iban'] ?? null,

'client\_swift' => $data['client\_swift'] ?? null

]);

// delete old non-provided services

$invoice->invoiceServices()->delete();

// create new services

if (!empty($data['services'])) {

foreach ($data['services'] as $serviceData) {

Service::create([

'id' => Str::uuid(),

'invoice\_id' => $invoice->id,

'name' => $serviceData['name'],

'price' => $serviceData['price'],

]);

}

}

});

}

/\*\*

\* @param Invoice $invoice

\* @return void

\*/

public function destroy(Invoice $invoice): void

{

$invoice->delete();

}

/\*\*

\* @param Invoice $invoice

\* @return string|null

\*/

public function show(Invoice $invoice): ?string

{

$options = new Options();

$options->set('defaultFont', 'DejaVu Sans');

$dompdf = new Dompdf($options);

$dompdf->loadHtml(view('admin.invoice.pdf', compact('invoice'))->render());

$dompdf->render();

return $dompdf->output();

}

public function setPaid(Invoice $invoice): void

{

$invoice->update(['paid' => true]);

}

public function setPaidWithTransaction(Invoice $invoice, string $transaction): void

{

$invoice->update(['paid' => true]);

}

}

**app/Services/MailSendingService.php:**

<?php

namespace App\Services;

use App\Mail\UnpaidInvoiceEmail;

use App\Models\Invoice;

use DomainException;

use Illuminate\Support\Facades\Mail;

class MailSendingService

{

/\*\*

\* @var InvoiceService

\*/

private InvoiceService $invoiceService;

/\*\*

\* @param InvoiceService $invoiceService

\*/

public function \_\_construct(InvoiceService $invoiceService)

{

$this->invoiceService = $invoiceService;

}

public function sendUnpaidInvoiceMessage(Invoice $invoice): void

{

if ($invoice->paid) {

throw new DomainException(\_\_('This invoice has already been paid.'));

}

Mail::to($invoice->client\_email)->send(new UnpaidInvoiceEmail($invoice, $this->invoiceService));

}

}

**app/Services/PostService.php:**

<?php

namespace App\Services;

use App\Models\Post;

use App\Models\User;

use Illuminate\Http\UploadedFile;

use Illuminate\Support\Str;

class PostService

{

/\*\*

\* @param array $data ['title' => 'string', 'slug' => 'string', 'text' => 'string', 'active' => bool, 'image' => UploadedImage]

\* @param User $user

\* @return Post

\*/

public function store(array $data, User $user): Post

{

return Post::create([

'user\_id' => $user->id,

'title' => $data['title'],

'slug' => $data['slug'],

'text' => $data['text'],

'active' => $data['active'],

'image' => $this->uploadImage($data['image'])

]);

}

/\*\*

\* @param Post $post

\* @param array $data ['title' => 'string', 'slug' => 'string', 'text' => 'string', 'active' => bool, 'image' => ?UploadedImage]

\* @return void

\*/

public function update(Post $post, array $data): void

{

// update post

$post->update([

'title' => $data['title'],

'slug' => $data['slug'],

'text' => $data['text'],

'active' => $data['active'],

'image' => isset($data['image']) ? $this->uploadImage($data['image'], $post) : $post->image,

]);

}

/\*\*

\* @param Post $post

\* @return void

\*/

public function destroy(Post $post): void

{

// always remove file before post destroying

if (is\_file($post->getImageFullPath())) {

unlink($post->getImageFullPath());

}

$post->delete();

}

/\*\*

\* @param UploadedFile $file

\* @param Post|null $post

\* @return string

\*/

private function uploadImage(UploadedFile $file, ?Post $post = null): string

{

// at first delete old image

if ($post !== null && is\_file($post->getImageFullPath())) {

unlink($post->getImageFullPath());

}

$extension = $file->extension();

$filename = $extension === '' ? Str::uuid() : sprintf('%s.%s', Str::uuid(), $extension);

// file uploading

$file->storeAs('images', $filename, 'public');

return $filename;

}

}

**app/Services/QuestionService.php:**

<?php

namespace App\Services;

use App\Models\Question;

use Carbon\Carbon;

use DomainException;

class QuestionService

{

public const SECONDS\_TO\_WAIT\_FOR\_NEXT\_MESSAGE\_WITH\_THE\_SAME\_IP = 180;

/\*\*

\* @param array $data ['email' => 'string', 'question\_text' => 'string']

\* @param string $ip

\* @param bool $verified

\* @return Question

\*/

public function create(array $data, string $ip, bool $verified): Question

{

$timeBefore = Carbon::now()->subSeconds(self::SECONDS\_TO\_WAIT\_FOR\_NEXT\_MESSAGE\_WITH\_THE\_SAME\_IP);

$questionsQty = Question::where('ip', $ip)->where('created\_at', '>=', $timeBefore)->count();

if ($questionsQty > 0) {

$message = sprintf(

\_\_('You are allowed to ask question only 1 time per %d seconds.'),

self::SECONDS\_TO\_WAIT\_FOR\_NEXT\_MESSAGE\_WITH\_THE\_SAME\_IP

);

throw new DomainException($message);

}

return Question::create([

'email' => $data['email'],

'question\_text' => $data['question\_text'],

'ip' => $ip,

'verified' => $verified

]);

}

/\*\*

\* @param Question $question

\* @param array $data ['verified' => bool]

\* @return void

\*/

public function update(Question $question, array $data): void

{

$question->update(['verified' => $data['verified']]);

}

/\*\*

\* @param Question $question

\* @return void

\*/

public function destroy(Question $question): void

{

$question->delete();

}

}

**app/Services/StripeApiService.php:**

<?php

namespace App\Services;

use Stripe\Checkout\Session as StripeSession;

use Stripe\Exception\ApiErrorException;

use Stripe\PaymentIntent;

use Stripe\StripeClient;

class StripeApiService

{

/\*\*

\* @throws ApiErrorException

\*/

public function generateProductId(StripeClient $stripeClient, array $data): string

{

return $stripeClient->products->create($data)->id;

}

/\*\*

\* @throws ApiErrorException

\*/

public function generatePriceId(StripeClient $stripeClient, array $data): string

{

return $stripeClient->prices->create($data)->id;

}

/\*\*

\* @throws ApiErrorException

\*/

public function generateSession(array $data): StripeSession

{

return StripeSession::create($data);

}

/\*\*

\* @throws ApiErrorException

\*/

public function retrievePaymentIntent(string $paymentIntentCode): PaymentIntent

{

return PaymentIntent::retrieve($paymentIntentCode);

}

/\*\*

\* @throws ApiErrorException

\*/

public function retrieveStripeSession(string $sessionId): StripeSession

{

return StripeSession::retrieve($sessionId);

}

}

**app/Services/StripeService.php:**

<?php

namespace App\Services;

use App\Models\Invoice;

use DomainException;

use Illuminate\Support\Facades\DB;

use Illuminate\Support\Facades\Log;

use Stripe\Checkout\Session as StripeSession;

use Stripe\Exception\ApiErrorException;

use Stripe\Stripe;

use Stripe\StripeClient;

class StripeService

{

/\*\*

\* @var InvoiceService

\*/

private InvoiceService $invoiceService;

/\*\*

\* @var StripeApiService

\*/

private StripeApiService $stripeApiService;

/\*\*

\* @param InvoiceService $invoiceService

\* @param StripeApiService $stripeApiService

\*/

public function \_\_construct(InvoiceService $invoiceService, StripeApiService $stripeApiService)

{

$this->invoiceService = $invoiceService;

$this->stripeApiService = $stripeApiService;

}

public function getRedirectUrl(Invoice $invoice): string

{

if ($invoice->getTotalPriceWithTax() <= 0) {

throw new DomainException(\_\_('Stripe payment can not bee free.'));

}

$stripeClient = $this->getStripeClient();

$priceId = $this->getPriceId($invoice, $stripeClient);

$stripeSession = $this->createStripeSession($invoice, $priceId);

$this->saveStripeSessionId($invoice, $stripeSession->id);

return $stripeSession->url;

}

private function getStripeClient(): StripeClient

{

return new StripeClient($this->getStripeSecretKey());

}

private function getStripeSecretKey(): string

{

return env('STRIPE\_SECRET\_KEY');

}

private function getPriceId(Invoice $invoice, StripeClient $stripeClient): string

{

try {

$productId = $this->stripeApiService

->generateProductId($stripeClient, ['name' => $invoice->reference]);

$priceId = $this->stripeApiService

->generatePriceId($stripeClient, [

'product' => $productId,

'unit\_amount' => $this->getSumInBaseIntegers($invoice->getTotalPriceWithTax(), 2),

'currency' => 'EUR',

]);

} catch (ApiErrorException $e) {

Log::error($e->getMessage(), debug\_backtrace(DEBUG\_BACKTRACE\_IGNORE\_ARGS));

throw new DomainException(\_\_('Impossible to obtain Stripe price.'));

}

return $priceId;

}

private function createStripeSession(Invoice $invoice, string $priceId): StripeSession

{

Stripe::setApiKey($this->getStripeSecretKey());

try {

$session = $this->stripeApiService->generateSession([

'line\_items' => [

[

'price' => $priceId,

'quantity' => 1,

],

],

'mode' => 'payment',

'success\_url' => route('payment.finish', $invoice),

'cancel\_url' => route('payment.cancel', $invoice),

'locale' => $this->getStripeSessionLocale(),

]);

} catch (ApiErrorException $e) {

Log::error($e->getMessage(), debug\_backtrace(DEBUG\_BACKTRACE\_IGNORE\_ARGS));

throw new DomainException(\_\_('Impossible to obtain Stripe session.'));

}

return $session;

}

private function getStripeSessionLocale(): string

{

return 'lv';

}

/\*\*

\* @param Invoice $invoice

\* @param string $sessionId

\* @return void

\*/

private function saveStripeSessionId(Invoice $invoice, string $sessionId): void

{

DB::table('stripe\_tokens')->where(['invoice\_id' => $invoice->id])->delete();

DB::table('stripe\_tokens')->insert([

'invoice\_id' => $invoice->id,

'token' => $sessionId,

]);

}

public function finalizeStripePayment(Invoice $invoice): void

{

if ($invoice->paid) {

return;

}

$sessionId = $this->getStripeSessionId($invoice);

Stripe::setApiKey($this->getStripeSecretKey());

try {

$session = $this->stripeApiService->retrieveStripeSession($sessionId);

$intent = $this->stripeApiService->retrievePaymentIntent($session->payment\_intent ?? '');

} catch (ApiErrorException $e) {

Log::error($e->getMessage(), debug\_backtrace(DEBUG\_BACKTRACE\_IGNORE\_ARGS));

throw new DomainException(\_\_('Impossible to obtain Stripe payment code.'));

}

if ($intent->status === 'succeeded' || $intent->status === 'pending') {

$transactionId = $intent->latest\_charge;

$this->invoiceService->setPaidWithTransaction($invoice, $transactionId);

}

}

/\*\*

\* @param Invoice $invoice

\* @return string

\*/

public function getStripeSessionId(Invoice $invoice): string

{

return (string)DB::table('stripe\_tokens')->where('invoice\_id', $invoice->id)

->value('token');

}

/\*\*

\* @param float $sum

\* @param int $decimalPlaces

\* @return int

\*/

private function getSumInBaseIntegers(float $sum, int $decimalPlaces): int

{

return (int)round($sum \* (10 \*\* $decimalPlaces));

}

}

**app/Services/UserService.php:**

<?php

namespace App\Services;

use App\Models\User;

use Illuminate\Support\Facades\Hash;

class UserService

{

/\*\*

\* @param array $data ['name' => 'string', 'email' => 'string', 'password' => 'string']

\* @return User

\*/

public function store(array $data): User

{

return User::create([

'name' => $data['name'],

'email' => $data['email'],

'password' => Hash::make($data['password'])

]);

}

/\*\*

\* @param User $user

\* @param array $data ['name' => 'string', 'email' => 'string', 'password' => 'string|null']

\* @return void

\*/

public function update(User $user, array $data): void

{

$values = [

'name' => $data['name'],

'email' => $data['email'],

];

// update password only password is provided

if (isset($data['password']) && $data['password'] !== '') {

$values['password'] = Hash::make($data['password']);

}

$user->update($values);

}

/\*\*

\* @param User $user

\* @return void

\*/

public function destroy(User $user): void

{

$user->delete();

}

/\*\*

\* @return User|null

\*/

public function getCurrentUser(): ?User

{

/\*\* @noinspection PhpIncompatibleReturnTypeInspection \*/

return auth()->guard('web')->user();

}

}

**app/bootstrap/app.php:**

<?php

/\*

|--------------------------------------------------------------------------

| Create The Application

|--------------------------------------------------------------------------

|

| The first thing we will do is create a new Laravel application instance

| which serves as the "glue" for all the components of Laravel, and is

| the IoC container for the system binding all of the various parts.

|

\*/

$app = new Illuminate\Foundation\Application(

$\_ENV['APP\_BASE\_PATH'] ?? dirname(\_\_DIR\_\_)

);

/\*

|--------------------------------------------------------------------------

| Bind Important Interfaces

|--------------------------------------------------------------------------

|

| Next, we need to bind some important interfaces into the container so

| we will be able to resolve them when needed. The kernels serve the

| incoming requests to this application from both the web and CLI.

|

\*/

$app->singleton(

Illuminate\Contracts\Http\Kernel::class,

App\Http\Kernel::class

);

$app->singleton(

Illuminate\Contracts\Console\Kernel::class,

App\Console\Kernel::class

);

$app->singleton(

Illuminate\Contracts\Debug\ExceptionHandler::class,

App\Exceptions\Handler::class

);

/\*

|--------------------------------------------------------------------------

| Return The Application

|--------------------------------------------------------------------------

|

| This script returns the application instance. The instance is given to

| the calling script so we can separate the building of the instances

| from the actual running of the application and sending responses.

|

\*/

return $app;

**app/config/…:**

<?php

use Illuminate\Support\Facades\Facade;

use Illuminate\Support\ServiceProvider;

return [

/\*

|--------------------------------------------------------------------------

| Application Name

|--------------------------------------------------------------------------

|

| This value is the name of your application. This value is used when the

| framework needs to place the application's name in a notification or

| any other location as required by the application or its packages.

|

\*/

'name' => env('APP\_NAME', 'Laravel'),

/\*

|--------------------------------------------------------------------------

| Application Environment

|--------------------------------------------------------------------------

|

| This value determines the "environment" your application is currently

| running in. This may determine how you prefer to configure various

| services the application utilizes. Set this in your ".env" file.

|

\*/

'env' => env('APP\_ENV', 'production'),

/\*

|--------------------------------------------------------------------------

| Application Debug Mode

|--------------------------------------------------------------------------

|

| When your application is in debug mode, detailed error messages with

| stack traces will be shown on every error that occurs within your

| application. If disabled, a simple generic error page is shown.

|

\*/

'debug' => (bool) env('APP\_DEBUG', false),

/\*

|--------------------------------------------------------------------------

| Application URL

|--------------------------------------------------------------------------

|

| This URL is used by the console to properly generate URLs when using

| the Artisan command line tool. You should set this to the root of

| your application so that it is used when running Artisan tasks.

|

\*/

'url' => env('APP\_URL', 'http://localhost'),

'asset\_url' => env('ASSET\_URL'),

/\*

|--------------------------------------------------------------------------

| Application Timezone

|--------------------------------------------------------------------------

|

| Here you may specify the default timezone for your application, which

| will be used by the PHP date and date-time functions. We have gone

| ahead and set this to a sensible default for you out of the box.

|

\*/

'timezone' => 'UTC',

/\*

|--------------------------------------------------------------------------

| Application Locale Configuration

|--------------------------------------------------------------------------

|

| The application locale determines the default locale that will be used

| by the translation service provider. You are free to set this value

| to any of the locales which will be supported by the application.

|

\*/

'locale' => 'en',

/\*

|--------------------------------------------------------------------------

| Application Fallback Locale

|--------------------------------------------------------------------------

|

| The fallback locale determines the locale to use when the current one

| is not available. You may change the value to correspond to any of

| the language folders that are provided through your application.

|

\*/

'fallback\_locale' => 'en',

/\*

|--------------------------------------------------------------------------

| Faker Locale

|--------------------------------------------------------------------------

|

| This locale will be used by the Faker PHP library when generating fake

| data for your database seeds. For example, this will be used to get

| localized telephone numbers, street address information and more.

|

\*/

'faker\_locale' => 'en\_US',

/\*

|--------------------------------------------------------------------------

| Encryption Key

|--------------------------------------------------------------------------

|

| This key is used by the Illuminate encrypter service and should be set

| to a random, 32 character string, otherwise these encrypted strings

| will not be safe. Please do this before deploying an application!

|

\*/

'key' => env('APP\_KEY'),

'cipher' => 'AES-256-CBC',

/\*

|--------------------------------------------------------------------------

| Maintenance Mode Driver

|--------------------------------------------------------------------------

|

| These configuration options determine the driver used to determine and

| manage Laravel's "maintenance mode" status. The "cache" driver will

| allow maintenance mode to be controlled across multiple machines.

|

| Supported drivers: "file", "cache"

|

\*/

'maintenance' => [

'driver' => 'file',

// 'store' => 'redis',

],

/\*

|--------------------------------------------------------------------------

| Autoloaded Service Providers

|--------------------------------------------------------------------------

|

| The service providers listed here will be automatically loaded on the

| request to your application. Feel free to add your own services to

| this array to grant expanded functionality to your applications.

|

\*/

'providers' => ServiceProvider::defaultProviders()->merge([

/\*

\* Package Service Providers...

\*/

/\*

\* Application Service Providers...

\*/

App\Providers\AppServiceProvider::class,

App\Providers\AuthServiceProvider::class,

// App\Providers\BroadcastServiceProvider::class,

App\Providers\EventServiceProvider::class,

App\Providers\RouteServiceProvider::class,

])->toArray(),

/\*

|--------------------------------------------------------------------------

| Class Aliases

|--------------------------------------------------------------------------

|

| This array of class aliases will be registered when this application

| is started. However, feel free to register as many as you wish as

| the aliases are "lazy" loaded so they don't hinder performance.

|

\*/

'aliases' => Facade::defaultAliases()->merge([

// 'Example' => App\Facades\Example::class,

])->toArray(),

];

<?php

use App\Models\Company;

return [

/\*

|--------------------------------------------------------------------------

| Authentication Defaults

|--------------------------------------------------------------------------

|

| This option controls the default authentication "guard" and password

| reset options for your application. You may change these defaults

| as required, but they're a perfect start for most applications.

|

\*/

'defaults' => [

'guard' => 'web',

'passwords' => 'users',

],

/\*

|--------------------------------------------------------------------------

| Authentication Guards

|--------------------------------------------------------------------------

|

| Next, you may define every authentication guard for your application.

| Of course, a great default configuration has been defined for you

| here which uses session storage and the Eloquent user provider.

|

| All authentication drivers have a user provider. This defines how the

| users are actually retrieved out of your database or other storage

| mechanisms used by this application to persist your user's data.

|

| Supported: "session"

|

\*/

'guards' => [

'web' => [

'driver' => 'session',

'provider' => 'users',

],

'company' => [

'driver' => 'session',

'provider' => 'companies',

],

],

/\*

|--------------------------------------------------------------------------

| User Providers

|--------------------------------------------------------------------------

|

| All authentication drivers have a user provider. This defines how the

| users are actually retrieved out of your database or other storage

| mechanisms used by this application to persist your user's data.

|

| If you have multiple user tables or models you may configure multiple

| sources which represent each model / table. These sources may then

| be assigned to any extra authentication guards you have defined.

|

| Supported: "database", "eloquent"

|

\*/

'providers' => [

'users' => [

'driver' => 'eloquent',

'model' => App\Models\User::class,

],

'companies' => [

'driver' => 'eloquent',

'model' => Company::class,

],

// 'users' => [

// 'driver' => 'database',

// 'table' => 'users',

// ],

],

/\*

|--------------------------------------------------------------------------

| Resetting Passwords

|--------------------------------------------------------------------------

|

| You may specify multiple password reset configurations if you have more

| than one user table or model in the application and you want to have

| separate password reset settings based on the specific user types.

|

| The expiry time is the number of minutes that each reset token will be

| considered valid. This security feature keeps tokens short-lived so

| they have less time to be guessed. You may change this as needed.

|

| The throttle setting is the number of seconds a user must wait before

| generating more password reset tokens. This prevents the user from

| quickly generating a very large amount of password reset tokens.

|

\*/

'passwords' => [

'users' => [

'provider' => 'users',

'table' => 'password\_reset\_tokens',

'expire' => 60,

'throttle' => 60,

],

'companies' => [

'provider' => 'companies',

'table' => 'companies\_password\_reset\_tokens',

'expire' => 60,

'throttle' => 60,

],

],

/\*

|--------------------------------------------------------------------------

| Password Confirmation Timeout

|--------------------------------------------------------------------------

|

| Here you may define the amount of seconds before a password confirmation

| times out and the user is prompted to re-enter their password via the

| confirmation screen. By default, the timeout lasts for three hours.

|

\*/

'password\_timeout' => 10800,

];

<?php

return [

/\*

|--------------------------------------------------------------------------

| Default Broadcaster

|--------------------------------------------------------------------------

|

| This option controls the default broadcaster that will be used by the

| framework when an event needs to be broadcast. You may set this to

| any of the connections defined in the "connections" array below.

|

| Supported: "pusher", "ably", "redis", "log", "null"

|

\*/

'default' => env('BROADCAST\_DRIVER', 'null'),

/\*

|--------------------------------------------------------------------------

| Broadcast Connections

|--------------------------------------------------------------------------

|

| Here you may define all of the broadcast connections that will be used

| to broadcast events to other systems or over websockets. Samples of

| each available type of connection are provided inside this array.

|

\*/

'connections' => [

'pusher' => [

'driver' => 'pusher',

'key' => env('PUSHER\_APP\_KEY'),

'secret' => env('PUSHER\_APP\_SECRET'),

'app\_id' => env('PUSHER\_APP\_ID'),

'options' => [

'cluster' => env('PUSHER\_APP\_CLUSTER'),

'host' => env('PUSHER\_HOST') ?: 'api-'.env('PUSHER\_APP\_CLUSTER', 'mt1').'.pusher.com',

'port' => env('PUSHER\_PORT', 443),

'scheme' => env('PUSHER\_SCHEME', 'https'),

'encrypted' => true,

'useTLS' => env('PUSHER\_SCHEME', 'https') === 'https',

],

'client\_options' => [

// Guzzle client options: https://docs.guzzlephp.org/en/stable/request-options.html

],

],

'ably' => [

'driver' => 'ably',

'key' => env('ABLY\_KEY'),

],

'redis' => [

'driver' => 'redis',

'connection' => 'default',

],

'log' => [

'driver' => 'log',

],

'null' => [

'driver' => 'null',

],

],

];

<?php

use Illuminate\Support\Str;

return [

/\*

|--------------------------------------------------------------------------

| Default Cache Store

|--------------------------------------------------------------------------

|

| This option controls the default cache connection that gets used while

| using this caching library. This connection is used when another is

| not explicitly specified when executing a given caching function.

|

\*/

'default' => env('CACHE\_DRIVER', 'file'),

/\*

|--------------------------------------------------------------------------

| Cache Stores

|--------------------------------------------------------------------------

|

| Here you may define all of the cache "stores" for your application as

| well as their drivers. You may even define multiple stores for the

| same cache driver to group types of items stored in your caches.

|

| Supported drivers: "apc", "array", "database", "file",

| "memcached", "redis", "dynamodb", "octane", "null"

|

\*/

'stores' => [

'apc' => [

'driver' => 'apc',

],

'array' => [

'driver' => 'array',

'serialize' => false,

],

'database' => [

'driver' => 'database',

'table' => 'cache',

'connection' => null,

'lock\_connection' => null,

],

'file' => [

'driver' => 'file',

'path' => storage\_path('framework/cache/data'),

'lock\_path' => storage\_path('framework/cache/data'),

],

'memcached' => [

'driver' => 'memcached',

'persistent\_id' => env('MEMCACHED\_PERSISTENT\_ID'),

'sasl' => [

env('MEMCACHED\_USERNAME'),

env('MEMCACHED\_PASSWORD'),

],

'options' => [

// Memcached::OPT\_CONNECT\_TIMEOUT => 2000,

],

'servers' => [

[

'host' => env('MEMCACHED\_HOST', '127.0.0.1'),

'port' => env('MEMCACHED\_PORT', 11211),

'weight' => 100,

],

],

],

'redis' => [

'driver' => 'redis',

'connection' => 'cache',

'lock\_connection' => 'default',

],

'dynamodb' => [

'driver' => 'dynamodb',

'key' => env('AWS\_ACCESS\_KEY\_ID'),

'secret' => env('AWS\_SECRET\_ACCESS\_KEY'),

'region' => env('AWS\_DEFAULT\_REGION', 'us-east-1'),

'table' => env('DYNAMODB\_CACHE\_TABLE', 'cache'),

'endpoint' => env('DYNAMODB\_ENDPOINT'),

],

'octane' => [

'driver' => 'octane',

],

],

/\*

|--------------------------------------------------------------------------

| Cache Key Prefix

|--------------------------------------------------------------------------

|

| When utilizing the APC, database, memcached, Redis, or DynamoDB cache

| stores there might be other applications using the same cache. For

| that reason, you may prefix every cache key to avoid collisions.

|

\*/

'prefix' => env('CACHE\_PREFIX', Str::slug(env('APP\_NAME', 'laravel'), '\_').'\_cache\_'),

];

<?php

return [

/\*

|--------------------------------------------------------------------------

| Cross-Origin Resource Sharing (CORS) Configuration

|--------------------------------------------------------------------------

|

| Here you may configure your settings for cross-origin resource sharing

| or "CORS". This determines what cross-origin operations may execute

| in web browsers. You are free to adjust these settings as needed.

|

| To learn more: https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS

|

\*/

'paths' => ['api/\*', 'sanctum/csrf-cookie'],

'allowed\_methods' => ['\*'],

'allowed\_origins' => ['\*'],

'allowed\_origins\_patterns' => [],

'allowed\_headers' => ['\*'],

'exposed\_headers' => [],

'max\_age' => 0,

'supports\_credentials' => false,

];

<?php

use Illuminate\Support\Str;

return [

/\*

|--------------------------------------------------------------------------

| Default Database Connection Name

|--------------------------------------------------------------------------

|

| Here you may specify which of the database connections below you wish

| to use as your default connection for all database work. Of course

| you may use many connections at once using the Database library.

|

\*/

'default' => env('DB\_CONNECTION', 'mysql'),

/\*

|--------------------------------------------------------------------------

| Database Connections

|--------------------------------------------------------------------------

|

| Here are each of the database connections setup for your application.

| Of course, examples of configuring each database platform that is

| supported by Laravel is shown below to make development simple.

|

|

| All database work in Laravel is done through the PHP PDO facilities

| so make sure you have the driver for your particular database of

| choice installed on your machine before you begin development.

|

\*/

'connections' => [

'sqlite' => [

'driver' => 'sqlite',

'url' => env('DATABASE\_URL'),

'database' => env('DB\_DATABASE', database\_path('database.sqlite')),

'prefix' => '',

'foreign\_key\_constraints' => env('DB\_FOREIGN\_KEYS', true),

],

'mysql' => [

'driver' => 'mysql',

'url' => env('DATABASE\_URL'),

'host' => env('DB\_HOST', '127.0.0.1'),

'port' => env('DB\_PORT', '3306'),

'database' => env('DB\_DATABASE', 'forge'),

'username' => env('DB\_USERNAME', 'forge'),

'password' => env('DB\_PASSWORD', ''),

'unix\_socket' => env('DB\_SOCKET', ''),

'charset' => 'utf8mb4',

'collation' => 'utf8mb4\_unicode\_ci',

'prefix' => '',

'prefix\_indexes' => true,

'strict' => true,

'engine' => null,

'options' => extension\_loaded('pdo\_mysql') ? array\_filter([

PDO::MYSQL\_ATTR\_SSL\_CA => env('MYSQL\_ATTR\_SSL\_CA'),

]) : [],

],

'pgsql' => [

'driver' => 'pgsql',

'url' => env('DATABASE\_URL'),

'host' => env('DB\_HOST', '127.0.0.1'),

'port' => env('DB\_PORT', '5432'),

'database' => env('DB\_DATABASE', 'forge'),

'username' => env('DB\_USERNAME', 'forge'),

'password' => env('DB\_PASSWORD', ''),

'charset' => 'utf8',

'prefix' => '',

'prefix\_indexes' => true,

'search\_path' => 'public',

'sslmode' => 'prefer',

],

'sqlsrv' => [

'driver' => 'sqlsrv',

'url' => env('DATABASE\_URL'),

'host' => env('DB\_HOST', 'localhost'),

'port' => env('DB\_PORT', '1433'),

'database' => env('DB\_DATABASE', 'forge'),

'username' => env('DB\_USERNAME', 'forge'),

'password' => env('DB\_PASSWORD', ''),

'charset' => 'utf8',

'prefix' => '',

'prefix\_indexes' => true,

// 'encrypt' => env('DB\_ENCRYPT', 'yes'),

// 'trust\_server\_certificate' => env('DB\_TRUST\_SERVER\_CERTIFICATE', 'false'),

],

],

/\*

|--------------------------------------------------------------------------

| Migration Repository Table

|--------------------------------------------------------------------------

|

| This table keeps track of all the migrations that have already run for

| your application. Using this information, we can determine which of

| the migrations on disk haven't actually been run in the database.

|

\*/

'migrations' => 'migrations',

/\*

|--------------------------------------------------------------------------

| Redis Databases

|--------------------------------------------------------------------------

|

| Redis is an open source, fast, and advanced key-value store that also

| provides a richer body of commands than a typical key-value system

| such as APC or Memcached. Laravel makes it easy to dig right in.

|

\*/

'redis' => [

'client' => env('REDIS\_CLIENT', 'phpredis'),

'options' => [

'cluster' => env('REDIS\_CLUSTER', 'redis'),

'prefix' => env('REDIS\_PREFIX', Str::slug(env('APP\_NAME', 'laravel'), '\_').'\_database\_'),

],

'default' => [

'url' => env('REDIS\_URL'),

'host' => env('REDIS\_HOST', '127.0.0.1'),

'username' => env('REDIS\_USERNAME'),

'password' => env('REDIS\_PASSWORD'),

'port' => env('REDIS\_PORT', '6379'),

'database' => env('REDIS\_DB', '0'),

],

'cache' => [

'url' => env('REDIS\_URL'),

'host' => env('REDIS\_HOST', '127.0.0.1'),

'username' => env('REDIS\_USERNAME'),

'password' => env('REDIS\_PASSWORD'),

'port' => env('REDIS\_PORT', '6379'),

'database' => env('REDIS\_CACHE\_DB', '1'),

],

],

];

<?php

return [

/\*

|--------------------------------------------------------------------------

| Default Filesystem Disk

|--------------------------------------------------------------------------

|

| Here you may specify the default filesystem disk that should be used

| by the framework. The "local" disk, as well as a variety of cloud

| based disks are available to your application. Just store away!

|

\*/

'default' => env('FILESYSTEM\_DISK', 'local'),

/\*

|--------------------------------------------------------------------------

| Filesystem Disks

|--------------------------------------------------------------------------

|

| Here you may configure as many filesystem "disks" as you wish, and you

| may even configure multiple disks of the same driver. Defaults have

| been set up for each driver as an example of the required values.

|

| Supported Drivers: "local", "ftp", "sftp", "s3"

|

\*/

'disks' => [

'local' => [

'driver' => 'local',

'root' => storage\_path('app'),

'throw' => false,

],

'public' => [

'driver' => 'local',

'root' => storage\_path('app/public'),

'url' => env('APP\_URL').'/storage',

'visibility' => 'public',

'throw' => false,

],

's3' => [

'driver' => 's3',

'key' => env('AWS\_ACCESS\_KEY\_ID'),

'secret' => env('AWS\_SECRET\_ACCESS\_KEY'),

'region' => env('AWS\_DEFAULT\_REGION'),

'bucket' => env('AWS\_BUCKET'),

'url' => env('AWS\_URL'),

'endpoint' => env('AWS\_ENDPOINT'),

'use\_path\_style\_endpoint' => env('AWS\_USE\_PATH\_STYLE\_ENDPOINT', false),

'throw' => false,

],

],

/\*

|--------------------------------------------------------------------------

| Symbolic Links

|--------------------------------------------------------------------------

|

| Here you may configure the symbolic links that will be created when the

| `storage:link` Artisan command is executed. The array keys should be

| the locations of the links and the values should be their targets.

|

\*/

'links' => [

public\_path('storage') => storage\_path('app/public'),

],

];

<?php

return [

/\*

|--------------------------------------------------------------------------

| Default Hash Driver

|--------------------------------------------------------------------------

|

| This option controls the default hash driver that will be used to hash

| passwords for your application. By default, the bcrypt algorithm is

| used; however, you remain free to modify this option if you wish.

|

| Supported: "bcrypt", "argon", "argon2id"

|

\*/

'driver' => 'bcrypt',

/\*

|--------------------------------------------------------------------------

| Bcrypt Options

|--------------------------------------------------------------------------

|

| Here you may specify the configuration options that should be used when

| passwords are hashed using the Bcrypt algorithm. This will allow you

| to control the amount of time it takes to hash the given password.

|

\*/

'bcrypt' => [

'rounds' => env('BCRYPT\_ROUNDS', 10),

],

/\*

|--------------------------------------------------------------------------

| Argon Options

|--------------------------------------------------------------------------

|

| Here you may specify the configuration options that should be used when

| passwords are hashed using the Argon algorithm. These will allow you

| to control the amount of time it takes to hash the given password.

|

\*/

'argon' => [

'memory' => 65536,

'threads' => 1,

'time' => 4,

],

];

<?php

use Monolog\Handler\NullHandler;

use Monolog\Handler\StreamHandler;

use Monolog\Handler\SyslogUdpHandler;

use Monolog\Processor\PsrLogMessageProcessor;

return [

/\*

|--------------------------------------------------------------------------

| Default Log Channel

|--------------------------------------------------------------------------

|

| This option defines the default log channel that gets used when writing

| messages to the logs. The name specified in this option should match

| one of the channels defined in the "channels" configuration array.

|

\*/

'default' => env('LOG\_CHANNEL', 'stack'),

/\*

|--------------------------------------------------------------------------

| Deprecations Log Channel

|--------------------------------------------------------------------------

|

| This option controls the log channel that should be used to log warnings

| regarding deprecated PHP and library features. This allows you to get

| your application ready for upcoming major versions of dependencies.

|

\*/

'deprecations' => [

'channel' => env('LOG\_DEPRECATIONS\_CHANNEL', 'null'),

'trace' => false,

],

/\*

|--------------------------------------------------------------------------

| Log Channels

|--------------------------------------------------------------------------

|

| Here you may configure the log channels for your application. Out of

| the box, Laravel uses the Monolog PHP logging library. This gives

| you a variety of powerful log handlers / formatters to utilize.

|

| Available Drivers: "single", "daily", "slack", "syslog",

| "errorlog", "monolog",

| "custom", "stack"

|

\*/

'channels' => [

'stack' => [

'driver' => 'stack',

'channels' => ['single'],

'ignore\_exceptions' => false,

],

'single' => [

'driver' => 'single',

'path' => storage\_path('logs/laravel.log'),

'level' => env('LOG\_LEVEL', 'debug'),

'replace\_placeholders' => true,

],

'daily' => [

'driver' => 'daily',

'path' => storage\_path('logs/laravel.log'),

'level' => env('LOG\_LEVEL', 'debug'),

'days' => 14,

'replace\_placeholders' => true,

],

'slack' => [

'driver' => 'slack',

'url' => env('LOG\_SLACK\_WEBHOOK\_URL'),

'username' => 'Laravel Log',

'emoji' => ':boom:',

'level' => env('LOG\_LEVEL', 'critical'),

'replace\_placeholders' => true,

],

'papertrail' => [

'driver' => 'monolog',

'level' => env('LOG\_LEVEL', 'debug'),

'handler' => env('LOG\_PAPERTRAIL\_HANDLER', SyslogUdpHandler::class),

'handler\_with' => [

'host' => env('PAPERTRAIL\_URL'),

'port' => env('PAPERTRAIL\_PORT'),

'connectionString' => 'tls://'.env('PAPERTRAIL\_URL').':'.env('PAPERTRAIL\_PORT'),

],

'processors' => [PsrLogMessageProcessor::class],

],

'stderr' => [

'driver' => 'monolog',

'level' => env('LOG\_LEVEL', 'debug'),

'handler' => StreamHandler::class,

'formatter' => env('LOG\_STDERR\_FORMATTER'),

'with' => [

'stream' => 'php://stderr',

],

'processors' => [PsrLogMessageProcessor::class],

],

'syslog' => [

'driver' => 'syslog',

'level' => env('LOG\_LEVEL', 'debug'),

'facility' => LOG\_USER,

'replace\_placeholders' => true,

],

'errorlog' => [

'driver' => 'errorlog',

'level' => env('LOG\_LEVEL', 'debug'),

'replace\_placeholders' => true,

],

'null' => [

'driver' => 'monolog',

'handler' => NullHandler::class,

],

'emergency' => [

'path' => storage\_path('logs/laravel.log'),

],

],

];

<?php

return [

/\*

|--------------------------------------------------------------------------

| Default Mailer

|--------------------------------------------------------------------------

|

| This option controls the default mailer that is used to send any email

| messages sent by your application. Alternative mailers may be setup

| and used as needed; however, this mailer will be used by default.

|

\*/

'default' => env('MAIL\_MAILER', 'smtp'),

/\*

|--------------------------------------------------------------------------

| Mailer Configurations

|--------------------------------------------------------------------------

|

| Here you may configure all of the mailers used by your application plus

| their respective settings. Several examples have been configured for

| you and you are free to add your own as your application requires.

|

| Laravel supports a variety of mail "transport" drivers to be used while

| sending an e-mail. You will specify which one you are using for your

| mailers below. You are free to add additional mailers as required.

|

| Supported: "smtp", "sendmail", "mailgun", "ses", "ses-v2",

| "postmark", "log", "array", "failover"

|

\*/

'mailers' => [

'smtp' => [

'transport' => 'smtp',

'url' => env('MAIL\_URL'),

'host' => env('MAIL\_HOST', 'smtp.mailgun.org'),

'port' => env('MAIL\_PORT', 587),

'encryption' => env('MAIL\_ENCRYPTION', 'tls'),

'username' => env('MAIL\_USERNAME'),

'password' => env('MAIL\_PASSWORD'),

'timeout' => null,

'local\_domain' => env('MAIL\_EHLO\_DOMAIN'),

],

'ses' => [

'transport' => 'ses',

],

'mailgun' => [

'transport' => 'mailgun',

// 'client' => [

// 'timeout' => 5,

// ],

],

'postmark' => [

'transport' => 'postmark',

// 'client' => [

// 'timeout' => 5,

// ],

],

'sendmail' => [

'transport' => 'sendmail',

'path' => env('MAIL\_SENDMAIL\_PATH', '/usr/sbin/sendmail -bs -i'),

],

'log' => [

'transport' => 'log',

'channel' => env('MAIL\_LOG\_CHANNEL'),

],

'array' => [

'transport' => 'array',

],

'failover' => [

'transport' => 'failover',

'mailers' => [

'smtp',

'log',

],

],

],

/\*

|--------------------------------------------------------------------------

| Global "From" Address

|--------------------------------------------------------------------------

|

| You may wish for all e-mails sent by your application to be sent from

| the same address. Here, you may specify a name and address that is

| used globally for all e-mails that are sent by your application.

|

\*/

'from' => [

'address' => env('MAIL\_FROM\_ADDRESS', 'hello@example.com'),

'name' => env('MAIL\_FROM\_NAME', 'Example'),

],

/\*

|--------------------------------------------------------------------------

| Markdown Mail Settings

|--------------------------------------------------------------------------

|

| If you are using Markdown based email rendering, you may configure your

| theme and component paths here, allowing you to customize the design

| of the emails. Or, you may simply stick with the Laravel defaults!

|

\*/

'markdown' => [

'theme' => 'default',

'paths' => [

resource\_path('views/vendor/mail'),

],

],

];

<?php

return [

/\*

|--------------------------------------------------------------------------

| Default Queue Connection Name

|--------------------------------------------------------------------------

|

| Laravel's queue API supports an assortment of back-ends via a single

| API, giving you convenient access to each back-end using the same

| syntax for every one. Here you may define a default connection.

|

\*/

'default' => env('QUEUE\_CONNECTION', 'sync'),

/\*

|--------------------------------------------------------------------------

| Queue Connections

|--------------------------------------------------------------------------

|

| Here you may configure the connection information for each server that

| is used by your application. A default configuration has been added

| for each back-end shipped with Laravel. You are free to add more.

|

| Drivers: "sync", "database", "beanstalkd", "sqs", "redis", "null"

|

\*/

'connections' => [

'sync' => [

'driver' => 'sync',

],

'database' => [

'driver' => 'database',

'table' => 'jobs',

'queue' => 'default',

'retry\_after' => 90,

'after\_commit' => false,

],

'beanstalkd' => [

'driver' => 'beanstalkd',

'host' => 'localhost',

'queue' => 'default',

'retry\_after' => 90,

'block\_for' => 0,

'after\_commit' => false,

],

'sqs' => [

'driver' => 'sqs',

'key' => env('AWS\_ACCESS\_KEY\_ID'),

'secret' => env('AWS\_SECRET\_ACCESS\_KEY'),

'prefix' => env('SQS\_PREFIX', 'https://sqs.us-east-1.amazonaws.com/your-account-id'),

'queue' => env('SQS\_QUEUE', 'default'),

'suffix' => env('SQS\_SUFFIX'),

'region' => env('AWS\_DEFAULT\_REGION', 'us-east-1'),

'after\_commit' => false,

],

'redis' => [

'driver' => 'redis',

'connection' => 'default',

'queue' => env('REDIS\_QUEUE', 'default'),

'retry\_after' => 90,

'block\_for' => null,

'after\_commit' => false,

],

],

/\*

|--------------------------------------------------------------------------

| Job Batching

|--------------------------------------------------------------------------

|

| The following options configure the database and table that store job

| batching information. These options can be updated to any database

| connection and table which has been defined by your application.

|

\*/

'batching' => [

'database' => env('DB\_CONNECTION', 'mysql'),

'table' => 'job\_batches',

],

/\*

|--------------------------------------------------------------------------

| Failed Queue Jobs

|--------------------------------------------------------------------------

|

| These options configure the behavior of failed queue job logging so you

| can control which database and table are used to store the jobs that

| have failed. You may change them to any database / table you wish.

|

\*/

'failed' => [

'driver' => env('QUEUE\_FAILED\_DRIVER', 'database-uuids'),

'database' => env('DB\_CONNECTION', 'mysql'),

'table' => 'failed\_jobs',

],

];

<?php

use Laravel\Sanctum\Sanctum;

return [

/\*

|--------------------------------------------------------------------------

| Stateful Domains

|--------------------------------------------------------------------------

|

| Requests from the following domains / hosts will receive stateful API

| authentication cookies. Typically, these should include your local

| and production domains which access your API via a frontend SPA.

|

\*/

'stateful' => explode(',', env('SANCTUM\_STATEFUL\_DOMAINS', sprintf(

'%s%s',

'localhost,localhost:3000,127.0.0.1,127.0.0.1:8000,::1',

Sanctum::currentApplicationUrlWithPort()

))),

/\*

|--------------------------------------------------------------------------

| Sanctum Guards

|--------------------------------------------------------------------------

|

| This array contains the authentication guards that will be checked when

| Sanctum is trying to authenticate a request. If none of these guards

| are able to authenticate the request, Sanctum will use the bearer

| token that's present on an incoming request for authentication.

|

\*/

'guard' => ['web'],

/\*

|--------------------------------------------------------------------------

| Expiration Minutes

|--------------------------------------------------------------------------

|

| This value controls the number of minutes until an issued token will be

| considered expired. If this value is null, personal access tokens do

| not expire. This won't tweak the lifetime of first-party sessions.

|

\*/

'expiration' => null,

/\*

|--------------------------------------------------------------------------

| Sanctum Middleware

|--------------------------------------------------------------------------

|

| When authenticating your first-party SPA with Sanctum you may need to

| customize some of the middleware Sanctum uses while processing the

| request. You may change the middleware listed below as required.

|

\*/

'middleware' => [

'verify\_csrf\_token' => App\Http\Middleware\VerifyCsrfToken::class,

'encrypt\_cookies' => App\Http\Middleware\EncryptCookies::class,

],

];

<?php

return [

/\*

|--------------------------------------------------------------------------

| Third Party Services

|--------------------------------------------------------------------------

|

| This file is for storing the credentials for third party services such

| as Mailgun, Postmark, AWS and more. This file provides the de facto

| location for this type of information, allowing packages to have

| a conventional file to locate the various service credentials.

|

\*/

'mailgun' => [

'domain' => env('MAILGUN\_DOMAIN'),

'secret' => env('MAILGUN\_SECRET'),

'endpoint' => env('MAILGUN\_ENDPOINT', 'api.mailgun.net'),

'scheme' => 'https',

],

'postmark' => [

'token' => env('POSTMARK\_TOKEN'),

],

'ses' => [

'key' => env('AWS\_ACCESS\_KEY\_ID'),

'secret' => env('AWS\_SECRET\_ACCESS\_KEY'),

'region' => env('AWS\_DEFAULT\_REGION', 'us-east-1'),

],

];

<?php

use Illuminate\Support\Str;

return [

/\*

|--------------------------------------------------------------------------

| Default Session Driver

|--------------------------------------------------------------------------

|

| This option controls the default session "driver" that will be used on

| requests. By default, we will use the lightweight native driver but

| you may specify any of the other wonderful drivers provided here.

|

| Supported: "file", "cookie", "database", "apc",

| "memcached", "redis", "dynamodb", "array"

|

\*/

'driver' => env('SESSION\_DRIVER', 'file'),

/\*

|--------------------------------------------------------------------------

| Session Lifetime

|--------------------------------------------------------------------------

|

| Here you may specify the number of minutes that you wish the session

| to be allowed to remain idle before it expires. If you want them

| to immediately expire on the browser closing, set that option.

|

\*/

'lifetime' => env('SESSION\_LIFETIME', 120),

'expire\_on\_close' => false,

/\*

|--------------------------------------------------------------------------

| Session Encryption

|--------------------------------------------------------------------------

|

| This option allows you to easily specify that all of your session data

| should be encrypted before it is stored. All encryption will be run

| automatically by Laravel and you can use the Session like normal.

|

\*/

'encrypt' => false,

/\*

|--------------------------------------------------------------------------

| Session File Location

|--------------------------------------------------------------------------

|

| When using the native session driver, we need a location where session

| files may be stored. A default has been set for you but a different

| location may be specified. This is only needed for file sessions.

|

\*/

'files' => storage\_path('framework/sessions'),

/\*

|--------------------------------------------------------------------------

| Session Database Connection

|--------------------------------------------------------------------------

|

| When using the "database" or "redis" session drivers, you may specify a

| connection that should be used to manage these sessions. This should

| correspond to a connection in your database configuration options.

|

\*/

'connection' => env('SESSION\_CONNECTION'),

/\*

|--------------------------------------------------------------------------

| Session Database Table

|--------------------------------------------------------------------------

|

| When using the "database" session driver, you may specify the table we

| should use to manage the sessions. Of course, a sensible default is

| provided for you; however, you are free to change this as needed.

|

\*/

'table' => 'sessions',

/\*

|--------------------------------------------------------------------------

| Session Cache Store

|--------------------------------------------------------------------------

|

| While using one of the framework's cache driven session backends you may

| list a cache store that should be used for these sessions. This value

| must match with one of the application's configured cache "stores".

|

| Affects: "apc", "dynamodb", "memcached", "redis"

|

\*/

'store' => env('SESSION\_STORE'),

/\*

|--------------------------------------------------------------------------

| Session Sweeping Lottery

|--------------------------------------------------------------------------

|

| Some session drivers must manually sweep their storage location to get

| rid of old sessions from storage. Here are the chances that it will

| happen on a given request. By default, the odds are 2 out of 100.

|

\*/

'lottery' => [2, 100],

/\*

|--------------------------------------------------------------------------

| Session Cookie Name

|--------------------------------------------------------------------------

|

| Here you may change the name of the cookie used to identify a session

| instance by ID. The name specified here will get used every time a

| new session cookie is created by the framework for every driver.

|

\*/

'cookie' => env(

'SESSION\_COOKIE',

Str::slug(env('APP\_NAME', 'laravel'), '\_').'\_session'

),

/\*

|--------------------------------------------------------------------------

| Session Cookie Path

|--------------------------------------------------------------------------

|

| The session cookie path determines the path for which the cookie will

| be regarded as available. Typically, this will be the root path of

| your application but you are free to change this when necessary.

|

\*/

'path' => '/',

/\*

|--------------------------------------------------------------------------

| Session Cookie Domain

|--------------------------------------------------------------------------

|

| Here you may change the domain of the cookie used to identify a session

| in your application. This will determine which domains the cookie is

| available to in your application. A sensible default has been set.

|

\*/

'domain' => env('SESSION\_DOMAIN'),

/\*

|--------------------------------------------------------------------------

| HTTPS Only Cookies

|--------------------------------------------------------------------------

|

| By setting this option to true, session cookies will only be sent back

| to the server if the browser has a HTTPS connection. This will keep

| the cookie from being sent to you when it can't be done securely.

|

\*/

'secure' => env('SESSION\_SECURE\_COOKIE'),

/\*

|--------------------------------------------------------------------------

| HTTP Access Only

|--------------------------------------------------------------------------

|

| Setting this value to true will prevent JavaScript from accessing the

| value of the cookie and the cookie will only be accessible through

| the HTTP protocol. You are free to modify this option if needed.

|

\*/

'http\_only' => true,

/\*

|--------------------------------------------------------------------------

| Same-Site Cookies

|--------------------------------------------------------------------------

|

| This option determines how your cookies behave when cross-site requests

| take place, and can be used to mitigate CSRF attacks. By default, we

| will set this value to "lax" since this is a secure default value.

|

| Supported: "lax", "strict", "none", null

|

\*/

'same\_site' => 'lax',

];

<?php

return [

/\*

|--------------------------------------------------------------------------

| View Storage Paths

|--------------------------------------------------------------------------

|

| Most templating systems load templates from disk. Here you may specify

| an array of paths that should be checked for your views. Of course

| the usual Laravel view path has already been registered for you.

|

\*/

'paths' => [

resource\_path('views'),

],

/\*

|--------------------------------------------------------------------------

| Compiled View Path

|--------------------------------------------------------------------------

|

| This option determines where all the compiled Blade templates will be

| stored for your application. Typically, this is within the storage

| directory. However, as usual, you are free to change this value.

|

\*/

'compiled' => env(

'VIEW\_COMPILED\_PATH',

realpath(storage\_path('framework/views'))

),

];

**app/database/migrations:**

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*/

public function up(): void

{

Schema::create('users', function (Blueprint $table) {

$table->id();

$table->string('name');

$table->string('email')->unique();

$table->timestamp('email\_verified\_at')->nullable();

$table->string('password');

$table->rememberToken();

$table->timestamps();

});

}

/\*\*

\* Reverse the migrations.

\*/

public function down(): void

{

Schema::dropIfExists('users');

}

};

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*/

public function up(): void

{

Schema::create('password\_reset\_tokens', function (Blueprint $table) {

$table->string('email')->primary();

$table->string('token');

$table->timestamp('created\_at')->nullable();

});

}

/\*\*

\* Reverse the migrations.

\*/

public function down(): void

{

Schema::dropIfExists('password\_reset\_tokens');

}

};

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*

\* @return void

\*/

public function up()

{

Schema::create('password\_resets', function (Blueprint $table) {

$table->string('email')->index();

$table->string('token');

$table->timestamp('created\_at')->nullable();

});

}

/\*\*

\* Reverse the migrations.

\*

\* @return void

\*/

public function down()

{

Schema::dropIfExists('password\_resets');

}

};

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*/

public function up(): void

{

Schema::create('failed\_jobs', function (Blueprint $table) {

$table->id();

$table->string('uuid')->unique();

$table->text('connection');

$table->text('queue');

$table->longText('payload');

$table->longText('exception');

$table->timestamp('failed\_at')->useCurrent();

});

}

/\*\*

\* Reverse the migrations.

\*/

public function down(): void

{

Schema::dropIfExists('failed\_jobs');

}

};

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*/

public function up(): void

{

Schema::create('personal\_access\_tokens', function (Blueprint $table) {

$table->id();

$table->morphs('tokenable');

$table->string('name');

$table->string('token', 64)->unique();

$table->text('abilities')->nullable();

$table->timestamp('last\_used\_at')->nullable();

$table->timestamp('expires\_at')->nullable();

$table->timestamps();

});

}

/\*\*

\* Reverse the migrations.

\*/

public function down(): void

{

Schema::dropIfExists('personal\_access\_tokens');

}

};

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*

\* @return void

\*/

public function up()

{

Schema::create('posts', function (Blueprint $table) {

$table->id();

$table->string('title', 255);

$table->string('slug', 255)->unique();

$table->string('image', 255);

$table->longText('text');

$table->boolean('active');

$table->bigInteger('user\_id')->unsigned()->nullable()->index();

$table->timestamps();

$table->foreign('user\_id')->on('users')->references('id')->onDelete('set null');

});

}

/\*\*

\* Reverse the migrations.

\*

\* @return void

\*/

public function down()

{

Schema::dropIfExists('posts');

}

};

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*/

public function up(): void

{

Schema::create('companies', function (Blueprint $table) {

$table->id();

$table->string('name', 255);

$table->string('reg\_number', 255)->unique();

$table->string('vat\_number', 255)->unique();

$table->text('address');

$table->text('description')->nullable();

$table->string('email', 255)->unique();

$table->string('phone', 255)->unique();

$table->string('api\_code')->nullable();

$table->string('password', 255);

$table->rememberToken();

$table->timestamps();

});

}

/\*\*

\* Reverse the migrations.

\*/

public function down(): void

{

Schema::dropIfExists('companies');

}

};

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*

\* @return void

\*/

public function up()

{

Schema::create('questions', function (Blueprint $table) {

$table->id();

$table->string('email', 255);

$table->text('question\_text');

$table->string('ip', 255)->index();

$table->boolean('verified');

$table->timestamps();

});

}

/\*\*

\* Reverse the migrations.

\*

\* @return void

\*/

public function down()

{

Schema::dropIfExists('questions');

}

};

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*/

public function up(): void

{

Schema::create('invoices', function (Blueprint $table) {

$table->id();

$table->string('reference')->unique();

$table->unsignedBigInteger('company\_id')->index();

$table->decimal('tax\_rate', 20, 2);

$table->string('provider\_name')->nullable();

$table->string('provider\_email')->nullable();

$table->string('provider\_phone')->nullable();

$table->string('provider\_company')->nullable();

$table->string('provider\_vat\_number')->nullable();

$table->string('provider\_reg\_number')->nullable();

$table->string('provider\_iban')->nullable();

$table->string('provider\_swift')->nullable();

$table->integer('client\_type')->unsigned();

$table->string('client\_name');

$table->string('client\_email');

$table->string('client\_phone');

$table->string('client\_company')->nullable();

$table->string('client\_vat\_number')->nullable();

$table->string('client\_reg\_number')->nullable();

$table->string('client\_iban')->nullable();

$table->string('client\_swift')->nullable();

$table->boolean('paid')->default(0);

$table->string('transaction\_code')->nullable();

$table->timestamps();

$table->foreign('company\_id')->on('companies')->references('id')->onDelete('cascade');

});

}

/\*\*

\* Reverse the migrations.

\*/

public function down(): void

{

Schema::dropIfExists('invoices');

}

};

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*/

public function up(): void

{

Schema::create('invoice\_services', function (Blueprint $table) {

$table->uuid('id')->unique();

$table->unsignedBigInteger('invoice\_id')->index();

$table->decimal('price', 20, 2);

$table->string('name');

$table->foreign('invoice\_id')->on('invoices')->references('id')->onDelete('cascade');

});

}

/\*\*

\* Reverse the migrations.

\*/

public function down(): void

{

Schema::dropIfExists('invoice\_services');

}

};

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*/

public function up(): void

{

Schema::create('stripe\_tokens', function (Blueprint $table) {

$table->unsignedBigInteger('invoice\_id')->index();

$table->string('token', 500);

$table->foreign('invoice\_id')->on('invoices')->references('id')->onDelete('cascade');

});

}

/\*\*

\* Reverse the migrations.

\*/

public function down(): void

{

Schema::dropIfExists('stripe\_tokens');

}

};

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*/

public function up(): void

{

Schema::create('companies\_password\_reset\_tokens', function (Blueprint $table) {

$table->string('email')->primary();

$table->string('token');

$table->timestamp('created\_at')->nullable();

});

}

/\*\*

\* Reverse the migrations.

\*/

public function down(): void

{

Schema::dropIfExists('companies\_password\_reset\_tokens');

}

};

**app/public:**

/\*-----------------------------------\*\

#CUSTOM PROPERTY

\\*-----------------------------------\*/

:root {

/\*\*

\* colors

\*/

--dark-cornflower-blue\_a7: hsla(214, 88%, 27%, 0.07);

--white: hsla(0, 0%, 100%, 1);

--white\_a3: hsla(0, 0%, 100%, 0.03);

--white\_a8: hsla(0, 0%, 100%, 0.08);

--white\_a12: hsla(0, 0%, 100%, 0.12);

--white\_a70: hsla(0, 0%, 100%, 0.7);

--cultured: hsla(220, 20%, 97%, 1);

--lavender-web: hsla(233, 52%, 94%, 1);

--cadet-blue-crayola: hsla(220, 12%, 70%, 1);

--cadet-blue-crayola\_a20: hsla(222, 23%, 71%, 0.2);

--charcoal: hsla(218, 22%, 26%, 1);

--raisin-black: hsla(216, 14%, 14%, 1);

--light-gray: hsla(0, 0%, 79%, 1);

--blue-crayola: hsla(219, 72%, 56%, 1);

--black-coral: hsla(220, 12%, 43%, 1);

/\*\*

\* typography

\*/

--ff-manrope: 'Manrope', sans-serif;

--fs-1: calc(2.7rem + 1.38vw);

--fs-2: calc(2.6rem + .66vw);

--fs-3: 2.2rem;

--fs-4: 1.9rem;

--fs-5: 1.8rem;

--fs-6: 1.7rem;

--fs-7: 1.5rem;

--fs-8: 1.4rem;

--fw-700: 700;

/\*\*

\* spacing

\*/

--section-padding: 90px;

/\*\*

\* box shadow

\*/

--shadow-1: 0 0 20px hsla(216, 14%, 14%, 0.05);

--shadow-2: 0 0 0 0.05rem hsla(214, 88%, 27%, 0.08), 0 0 1.25rem hsla(216, 14%, 14%, 0.06);

--shadow-3: 0 0 1.25rem hsla(216, 14%, 14%, 0.04);

/\*\*

\* border radius

\*/

--radius-circle: 50%;

--radius-pill: 100px;

--radius-10: 10px;

--radius-8: 8px;

--radius-6: 6px;

/\*\*

\* transition

\*/

--transition-1: 0.25s ease;

--transition-2: 0.5s ease;

--transition-3: 0.3s ease-in-out;

}

/\*-----------------------------------\*\

#RESET

\\*-----------------------------------\*/

\*,

\*::before,

\*::after {

margin: 0;

padding: 0;

box-sizing: border-box;

}

li { list-style: none; }

a,

img,

span,

input,

button,

ion-icon { display: block; }

a {

color: inherit;

text-decoration: none;

}

img { height: auto; }

input,

button {

background: none;

border: none;

font: inherit;

}

input { width: 100%; }

button { cursor: pointer; }

ion-icon { pointer-events: none; }

address { font-style: normal; }

html {

font-family: var(--ff-manrope);

font-size: 10px;

scroll-behavior: smooth;

}

body {

background-color: var(--white);

color: var(--black-coral);

font-size: 1.6rem;

line-height: 1.7;

}

body.nav-active { overflow: hidden; }

::-webkit-scrollbar { width: 10px; }

::-webkit-scrollbar-track { background-color: hsl(0, 0%, 98%); }

::-webkit-scrollbar-thumb { background-color: hsl(0, 0%, 80%); }

::-webkit-scrollbar-thumb:hover { background-color: hsl(0, 0%, 70%); }

/\*-----------------------------------\*\

#REUSED STYLE

\\*-----------------------------------\*/

.container { padding-inline: 16px; }

.social-list {

display: flex;

align-items: center;

gap: 12px;

}

.social-link {

font-size: 2rem;

transition: var(--transition-1);

}

.social-link:is(:hover, :focus-visible) { transform: translateY(-3px); }

.section { padding-block: var(--section-padding); }

.has-bg-image {

background-repeat: no-repeat;

background-position: center;

background-size: cover;

}

.h1,

.h2,

.h3,

.h4,

.h5 {

color: var(--charcoal);

font-weight: var(--fw-700);

line-height: 1.3;

}

.h1 { font-size: var(--fs-1); }

.h2 { font-size: var(--fs-2); }

.h3 { font-size: var(--fs-3); }

.h4 { font-size: var(--fs-4); }

.h5 { font-size: var(--fs-6); }

.btn {

color: var(--white);

font-size: var(--fs-6);

font-weight: var(--fw-700);

border: 2px solid var(--white);

max-width: max-content;

padding: 12px 28px;

border-radius: var(--radius-pill);

transition: var(--transition-1);

will-change: transform;

}

.btn:is(:hover, :focus-visible) { transform: translateY(-4px); }

.btn-primary,

.btn-outline:is(:hover, :focus-visible) {

background-color: var(--white);

color: var(--charcoal);

}

.img-holder {

aspect-ratio: var(--width) / var(--height);

background-color: var(--light-gray);

overflow: hidden;

}

.img-cover {

width: 100%;

height: 100%;

object-fit: cover;

}

.section-subtitle {

font-size: var(--fs-7);

text-transform: uppercase;

color: var(--violet-blue-crayola);

font-weight: var(--fw-700);

margin-block-end: 16px;

}

.grid-list {

display: grid;

gap: 25px;

}

.w-100 { width: 100%; }

/\*-----------------------------------\*\

#HEADER

\\*-----------------------------------\*/

.header .btn { display: none; }

.header {

position: absolute;

top: 0;

left: 0;

width: 100%;

padding-block: 20px;

box-shadow: var(--shadow-1);

z-index: 4;

}

.header .container {

display: flex;

justify-content: space-between;

align-items: center;

gap: 16px;

}

.nav-open-btn {

font-size: 3.5rem;

color: var(--white);

}

.navbar {

position: fixed;

top: 0;

left: -300px;

background-color: var(--raisin-black);

color: var(--white);

max-width: 300px;

width: 100%;

height: 100vh;

padding: 30px;

padding-block-end: 40px;

display: flex;

flex-direction: column;

gap: 30px;

z-index: 2;

visibility: hidden;

transition: var(--transition-3);

}

.navbar.active {

visibility: visible;

transform: translateX(300px);

}

.navbar-top {

display: flex;

justify-content: space-between;

align-items: center;

}

.nav-close-btn {

background-color: var(--white\_a8);

color: var(--white);

font-size: 2rem;

padding: 6px;

border-radius: var(--radius-circle);

transition: var(--transition-1);

}

.navbar-list { margin-block-end: auto; }

.navbar-link {

font-weight: var(--fw-700);

padding-block: 6px;

}

/\*-----------------------------------\*\

#HERO

\\*-----------------------------------\*/

.hero {

padding-block-start: calc(var(--section-padding) + 70px);

text-align: center;

}

.hero .container {

display: grid;

gap: 70px;

}

.hero-title { color: var(--white); }

.hero-text {

font-size: var(--fs-5);

color: var(--white);

margin-block: 24px 36px;

}

.btn-wrapper {

display: flex;

flex-wrap: wrap;

justify-content: center;

align-items: center;

gap: 12px;

}

.hero-slider,

.hero-card { position: relative; }

.hero-card .play-btn {

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

background-color: var(--white);

color: var(--charcoal);

font-size: 2rem;

padding: 25px;

border-radius: var(--radius-circle);

transition: var(--transition-1);

animation: pulse 2s ease infinite;

}

.hero-card .play-btn:is(:hover, :focus-visible) { color: var(--blue-crayola); }

@keyframes pulse {

0% { box-shadow: 0 0 0 0 var(--white\_a70); }

75% { box-shadow: 0 0 0 20px transparent; }

}

.hero .slider-inner {

border-radius: var(--radius-10);

overflow: hidden;

}

.hero .slider-container {

position: relative;

display: flex;

align-items: flex-start;

gap: 8px;

transition: var(--transition-2);

}

.hero .slider-item {

min-width: 100%;

width: 100%;

border-radius: var(--radius-10);

overflow: hidden;

}

.hero .slider-btn {

position: absolute;

top: 50%;

transform: translateY(-50%);

background-color: var(--white\_a70);

color: var(--charcoal);

font-size: 2rem;

padding: 12px;

border-radius: var(--radius-circle);

transition: var(--transition-1);

}

.hero .slider-btn:is(:hover, :focus-visible) { background-color: var(--white); }

.hero .slider-btn.prev { left: 20px; }

.hero .slider-btn.next { right: 20px; }

/\*-----------------------------------\*\

#SERVICE

\\*-----------------------------------\*/

.service { text-align: center; }

.service .section-title { margin-block-end: 50px; }

.service-card {

padding: 40px;

box-shadow: var(--shadow-2);

border-radius: var(--radius-6);

}

.service-card .card-icon {

width: 60px;

height: 60px;

background-color: var(--blue-crayola);

display: grid;

place-items: center;

color: var(--white);

font-size: 2.5rem;

border-radius: var(--radius-circle);

margin-inline: auto;

}

.service-card .card-icon ion-icon { --ionicon-stroke-width: 50px; }

.service-card .card-title { margin-block: 16px 10px; }

.service-card .btn-text {

display: flex;

justify-content: center;

align-items: center;

gap: 4px;

margin-block-start: 10px;

color: var(--blue-crayola);

font-weight: var(--fw-700);

transition: var(--transition-1);

}

.service-card .btn-text:is(:hover, :focus-visible) { opacity: 0.9; }

/\*-----------------------------------\*\

#ABOUT

\\*-----------------------------------\*/

.about .container {

display: grid;

gap: 50px;

}

.about .section-title { margin-block-end: 35px; }

.accordion-card .card-title { padding-block-end: 20px; }

.accordion-btn {

display: flex;

align-items: center;

gap: 10px;

}

.accordion-btn ion-icon {

font-size: 1.5rem;

color: var(--blue-crayola);

transition: var(--transition-1);

}

.accordion-card.expanded .accordion-btn ion-icon {

transform: rotate(0.5turn);

}

.accordion-btn .span { transition: var(--transition-1); }

.accordion-btn:is(:hover, :focus-visible) .span,

.accordion-card.expanded .accordion-btn .span { color: var(--violet-blue-crayola); }

.accordion-content {

padding-inline-start: 24px;

max-height: 0;

overflow: hidden;

}

.accordion-card.expanded .accordion-content {

max-height: max-content;

padding-block-end: 20px;

}

/\*-----------------------------------\*\

#FEATURES

\\*-----------------------------------\*/

.feature .container {

display: grid;

gap: 50px;

}

.feature .section-text { margin-block: 25px 30px; }

.feature-list {

display: grid;

gap: 15px;

}

.feature-card {

display: flex;

align-items: center;

gap: 10px;

}

.feature-card .card-icon {

background-color: var(--lavender-web);

font-size: 1.4rem;

padding: 4px;

border-radius: var(--radius-circle);

}

.feature-card .card-icon ion-icon { --ionicon-stroke-width: 40px; }

/\*-----------------------------------\*\

#STATS

\\*-----------------------------------\*/

.stats {

background-image: linear-gradient(to bottom, var(--white) 50%, var(--cultured) 50%);

display: flex;

justify-content: center;

align-items: center;

}

.stats-card {

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

text-align: center;

gap: 36px 24px;

padding: 45px 30px;

border-radius: var(--radius-8);

}

.stats-card .card-text > \* {

color: var(--white);

}

/\*-----------------------------------\*\

#PROJECT

\\*-----------------------------------\*/

.project { background-color: var(--cultured); }

.project :is(.section-subtitle, .section-title) { text-align: center; }

.project .section-title { margin-block-end: 50px; }

.project-card {

background-color: var(--white);

border-radius: var(--radius-8);

overflow: hidden;

box-shadow: var(--shadow-2);

height: 100%;

}

.project-card .card-content { padding: 30px; }

.project-card .card-title { transition: var(--transition-1); }

.project-card .card-title:is(:hover, :focus-visible) { color: var(--violet-blue-crayola); }

.project-card .card-text { margin-block: 16px 20px; }

.project-card .card-meta-list {

display: flex;

flex-wrap: wrap;

gap: 12px;

}

.project-card .card-meta-item {

display: flex;

align-items: center;

gap: 4px;

color: var(--cadet-blue-crayola);

font-size: var(--fs-8);

}

/\*-----------------------------------\*\

#CTA

\\*-----------------------------------\*/

.cta { background-color: var(--charcoal); }

.cta .container {

padding-block: 100px 60px;

border-block-end: 1px solid var(--cadet-blue-crayola\_a20);

}

.cta .section-title {

color: var(--white);

margin-block-end: 30px;

}

/\*-----------------------------------\*\

#FOOTER

\\*-----------------------------------\*/

.footer {

background-color: var(--charcoal);

padding-block: 60px 100px;

color: var(--light-gray);

}

.footer-brand .footer-text { margin-block: 20px; }

/\*-----------------------------------\*\

#MEDIA QUERIES

\\*-----------------------------------\*/

/\*\*

\* responsive for large than 575px screen

\*/

@media (min-width: 575px) {

/\*\*

\* REUSED STYLE

\*/

.container {

max-width: 540px;

width: 100%;

margin-inline: auto;

}

/\*\*

\* PROJECTS

\*/

.project-card .card-content { padding: 40px; }

}

/\*\*

\* responsive for large than 768px screen

\*/

@media (min-width: 768px) {

/\*\*

\* CUSTOM PROPERTY

\*/

:root {

/\*\*

\* spacing

\*/

--section-padding: 120px;

}

/\*\*

\* REUSED STYLE

\*/

.container { max-width: 720px; }

.grid-list { grid-template-columns: 1fr 1fr; }

/\*\*

\* HEADER

\*/

.header .btn {

display: block;

margin-inline-start: auto;

padding: 8px 20px;

}

.header.active .btn {

background-color: var(--violet-blue-crayola);

border-color: var(--violet-blue-crayola);

color: var(--white);

}

/\*\*

\* HERO

\*/

.hero-content {

max-width: 85%;

margin-inline: auto;

}

.hero-text { --fs-5: 2rem; }

/\*\*

\* SERVICE

\*/

.service .section-title {

max-width: 30ch;

margin-inline: auto;

}

/\*\*

\* ABOUT

\*/

.about .container {

grid-template-columns: 1fr 0.7fr;

align-items: center;

}

/\*\*

\* FEATURE

\*/

.feature .container {

grid-template-columns: 0.7fr 1fr;

align-items: center;

}

.feature-banner { order: 1; }

}

/\*\*

\* responsive for large than 992px screen

\*/

@media (min-width: 992px) {

/\*\*

\* REUSED STYLE

\*/

.container { max-width: 960px; }

/\*\*

\* HEADER

\*/

.nav-open-btn,

.navbar > \*:not(.navbar-list),

.overlay { display: none; }

.navbar,

.navbar.active {

all: unset;

display: block;

margin-inline: auto 24px;

}

.navbar-list {

display: flex;

gap: 30px;

}

.navbar-link {

color: var(--white);

transition: var(--transition-1);

}

.navbar-link:is(:hover, :focus-visible) { opacity: 0.7; }

.header.active .navbar-link { color: var(--charcoal); }

.header.active .navbar-link:is(:hover, :focus-visible) {

opacity: 1;

color: var(--violet-blue-crayola);

}

.header .btn { margin-inline-start: 0; }

/\*\*

\* HERO

\*/

.hero {

padding-block-start: calc(var(--section-padding) + 50px);

text-align: left;

}

.hero .container {

grid-template-columns: 1fr 0.8fr;

align-items: center;

}

.hero-content {

max-width: unset;

margin-inline: 0;

}

.btn-wrapper { justify-content: flex-start; }

/\*\*

\* STATS

\*/

.stats-card { grid-template-columns: repeat(4, 1fr); }

/\*\*

\* PROJECT

\*/

.project .section-title {

max-width: 32ch;

margin-inline: auto;

}

/\*\*

\* FOOTER

\*/

.footer .grid-list { grid-template-columns: repeat(4, 1fr); }

}

/\*\*

\* responsive for large than 1200px screen

\*/

@media (min-width: 1200px) {

/\*\*

\* REUSED STYLE

\*/

.container { max-width: 1140px; }

/\*\*

\* HERO

\*/

.hero-text { padding-inline-end: 100px; }

/\*\*

\* SERVICE

\*/

.service .grid-list { grid-template-columns: repeat(4, 1fr); }

/\*\*

\* STATS

\*/

.stats .container { max-width: 70%; }

.stats-card { padding: 60px; }

/\*\*

\* PROJECT

\*/

.project .grid-list { grid-template-columns: repeat(3, 1fr); }

/\*\*

\* FOOTER

\*/

address.footer-text {

padding-inline-end: 100px;

margin-block-end: 16px;

}

}

/\*\*

\* responsive for large than 1400px screen

\*/

@media (min-width: 1400px) {

/\*\*

\* REUSED STYLE

\*/

.container { max-width: 1320px; }

/\*\*

\* FEATURE

\*/

.feature-list { grid-template-columns: 1fr 1fr; }

}

'use strict';

/\*\*

\* add event listener on multiple elements

\*/

const addEventOnElements = function (elements, eventType, callback) {

for (let i = 0, len = elements.length; i < len; i++) {

elements[i].addEventListener(eventType, callback);

}

}

/\*\*

\* NAVBAR TOGGLE FOR MOBILE

\*/

const navbar = document.querySelector("[data-navbar]");

const navTogglers = document.querySelectorAll("[data-nav-toggler]");

const overlay = document.querySelector("[data-overlay]");

const toggleNavbar = function () {

navbar.classList.toggle("active");

overlay.classList.toggle("active");

document.body.classList.toggle("nav-active");

}

addEventOnElements(navTogglers, "click", toggleNavbar);

/\*\*

\* SLIDER

\*/

const sliders = document.querySelectorAll("[data-slider]");

const initSlider = function(currentSlider) {

const sldierContainer = currentSlider.querySelector("[data-slider-container]");

const sliderPrevBtn = currentSlider.querySelector("[data-slider-prev]");

const sliderNextBtn = currentSlider.querySelector("[data-slider-next]");

let currentSlidePos = 0;

const moveSliderItem = function () {

sldierContainer.style.transform = `translateX(-${sldierContainer.children[currentSlidePos].offsetLeft}px)`;

}

/\*\*

\* NEXT SLIDE

\*/

const slideNext = function () {

const slideEnd = currentSlidePos >= sldierContainer.childElementCount - 1;

if (slideEnd) {

currentSlidePos = 0;

} else {

currentSlidePos++;

}

moveSliderItem();

}

sliderNextBtn.addEventListener("click", slideNext);

/\*\*

\* PREVIOUS SLIDE

\*/

const slidePrev = function () {

if (currentSlidePos <= 0) {

currentSlidePos = sldierContainer.childElementCount - 1;

} else {

currentSlidePos--;

}

moveSliderItem();

}

sliderPrevBtn.addEventListener("click", slidePrev);

const dontHaveExtraItem = sldierContainer.childElementCount <= 1;

if (dontHaveExtraItem) {

sliderNextBtn.style.display = "none";

sliderPrevBtn.style.display = "none";

}

}

for (let i = 0, len = sliders.length; i < len; i++) { initSlider(sliders[i]); }

/\*\*

\* ACCORDION

\*/

const accordions = document.querySelectorAll("[data-accordion]");

let lastActiveAccordion = accordions[0];

const initAccordion = function (currentAccordion) {

const accordionBtn = currentAccordion.querySelector("[data-accordion-btn]");

const expandAccordion = function () {

if (lastActiveAccordion && lastActiveAccordion !== currentAccordion) {

lastActiveAccordion.classList.remove("expanded");

}

currentAccordion.classList.toggle("expanded");

lastActiveAccordion = currentAccordion;

}

accordionBtn.addEventListener("click", expandAccordion);

}

for (let i = 0, len = accordions.length; i < len; i++) { initAccordion(accordions[i]); }

\*{

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: "Poppins", sans-serif;

}

body{

display: flex;

justify-content: center;

align-items: center;

min-height: 100vh;

background: url(bg.jpg) no-repeat;

background-size: cover;

background-position: center;

}

.wrapper{

width: 420px;

background: transparent;

border: 2px solid rgba(255, 255, 255, .2);

backdrop-filter: blur(9px);

color: #fff;

border-radius: 12px;

padding: 30px 40px;

}

.wrapper h1{

font-size: 36px;

text-align: center;

}

.wrapper .input-box{

position: relative;

width: 100%;

margin: 30px 0;

}

.input-box input, .input-box textarea {

width: 100%;

background: transparent;

border: none;

outline: none;

border: 2px solid rgba(255, 255, 255, .2);

border-radius: 40px;

font-size: 16px;

color: #fff;

padding: 20px 45px 20px 20px;

}

.input-box input::placeholder, .input-box textarea::placeholder{

color: #fff;

}

.input-box i {

position: absolute;

right: 20px;

top: 30%;

transform: translate(-50%);

font-size: 20px;

}

.wrapper .remember-forgot{

display: flex;

justify-content: space-between;

font-size: 14.5px;

margin: -15px 0 15px;

}

.remember-forgot label input{

accent-color: #fff;

margin-right: 3px;

}

.remember-forgot a{

color: #fff;

text-decoration: none;

}

.remember-forgot a:hover{

text-decoration: underline;

}

.wrapper .btn{

width: 100%;

height: 45px;

background: #fff;

border: none;

outline: none;

border-radius: 40px;

box-shadow: 0 0 10px rgba(0, 0, 0, .1);

cursor: pointer;

font-size: 16px;

color: #333;

font-weight: 600;

}

.wrapper .main-link{

font-size: 14.5px;

text-align: center;

margin: 20px 0 15px;

}

.main-link p a{

color: #fff;

text-decoration: none;

font-weight: 600;

}

.main-link p a:hover{

text-decoration: underline;

}

textarea {

resize: none;

min-height: 140px;

}

**App/resources/views:**

@extends('admin-layout')

@section('main\_content')

<form action="{{ route('admin.companies.store') }}" enctype="multipart/form-data" method="POST">

@csrf

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Create Company') }}</strong>

</div>

<div class="col-lg-6 text-end">

<a href="{{ route('admin.companies.index') }}" class="btn btn-primary">

{{ \_\_('Return to Companies') }}

</a>

</div>

</div>

</div>

<div class="card-body">

<div class="row mb-3">

<label for="name" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Name') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="name" type="text" class="form-control @error('name') is-invalid @enderror" name="name" value="{{ old('name') }}" autocomplete="off">

@error('name')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="reg\_number" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Reg. Number') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="reg\_number" type="text" class="form-control @error('reg\_number') is-invalid @enderror" name="reg\_number" value="{{ old('reg\_number') }}" autocomplete="off">

@error('reg\_number')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="vat\_number" class="col-lg-3 col-form-label text-md-end">{{ \_\_('VAT Number') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="vat\_number" type="text" class="form-control @error('vat\_number') is-invalid @enderror" name="vat\_number" value="{{ old('vat\_number') }}" autocomplete="off">

@error('vat\_number')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="email" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Email') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="email" type="text" class="form-control @error('email') is-invalid @enderror" name="email" value="{{ old('email') }}" autocomplete="off">

@error('email')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="password" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Password') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="password" type="password" class="form-control @error('password') is-invalid @enderror" name="password" value="" autocomplete="off">

@error('password')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="phone" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Phone') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="phone" type="text" class="form-control @error('phone') is-invalid @enderror" name="phone" value="{{ old('phone') }}" autocomplete="off">

@error('phone')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="address" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Address') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<textarea id="address" class="form-control @error('address') is-invalid @enderror" name="address" autocomplete="off">{{ old('address') }}</textarea>

@error('address')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="api\_code" class="col-lg-3 col-form-label text-md-end">{{ \_\_('API Code') }}</label>

<div class="col-lg-6">

<input id="api\_code" type="text" class="form-control @error('api\_code') is-invalid @enderror" name="api\_code" value="{{ old('api\_code') }}" autocomplete="off">

@error('api\_code')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="description" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Description') }}</label>

<div class="col-lg-6">

<textarea id="description" class="form-control @error('description') is-invalid @enderror" name="description" autocomplete="off">{{ old('description') }}</textarea>

@error('description')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

</div>

<div class="card-footer text-end">

<button type="submit" class="btn btn-primary text-uppercase">

{{ \_\_('Create') }}

</button>

</div>

</div>

</div>

</div>

</div>

</form>

@endsection

@extends('admin-layout')

@section('main\_content')

<form action="{{ route('admin.companies.update', $company) }}" enctype="multipart/form-data" method="POST">

@method('PATCH')

@csrf

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Update Company') }}</strong>

</div>

<div class="col-lg-6 text-end">

<a href="{{ route('admin.companies.index') }}" class="btn btn-primary">

{{ \_\_('Return to Companies') }}

</a>

</div>

</div>

</div>

<div class="card-body">

<div class="row mb-3">

<label for="name" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Name') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="name" type="text" class="form-control @error('name') is-invalid @enderror" name="name" value="{{ old('name', $company->name) }}" autocomplete="off">

@error('name')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="reg\_number" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Reg. Number') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="reg\_number" type="text" class="form-control @error('reg\_number') is-invalid @enderror" name="reg\_number" value="{{ old('reg\_number', $company->reg\_number) }}" autocomplete="off">

@error('reg\_number')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="vat\_number" class="col-lg-3 col-form-label text-md-end">{{ \_\_('VAT Number') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="vat\_number" type="text" class="form-control @error('vat\_number') is-invalid @enderror" name="vat\_number" value="{{ old('vat\_number', $company->vat\_number) }}" autocomplete="off">

@error('vat\_number')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="email" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Email') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="email" type="text" class="form-control @error('email') is-invalid @enderror" name="email" value="{{ old('email', $company->email) }}" autocomplete="off">

@error('email')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="password" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Password') }}</label>

<div class="col-lg-6">

<input id="password" type="password" class="form-control @error('password') is-invalid @enderror" name="password" value="" autocomplete="off">

@error('password')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="phone" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Phone') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="phone" type="text" class="form-control @error('phone') is-invalid @enderror" name="phone" value="{{ old('phone', $company->phone) }}" autocomplete="off">

@error('phone')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="address" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Address') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<textarea id="address" class="form-control @error('address') is-invalid @enderror" name="address" autocomplete="off">{{ old('address', $company->address) }}</textarea>

@error('address')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="api\_code" class="col-lg-3 col-form-label text-md-end">{{ \_\_('API Code') }}</label>

<div class="col-lg-6">

<input id="api\_code" type="text" class="form-control @error('api\_code') is-invalid @enderror" name="api\_code" value="{{ old('api\_code', $company->api\_code) }}" autocomplete="off">

@error('api\_code')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="description" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Description') }}</label>

<div class="col-lg-6">

<textarea id="description" class="form-control @error('description') is-invalid @enderror" name="description" autocomplete="off">{{ old('description', $company->description) }}</textarea>

@error('description')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

</div>

<div class="card-footer text-end">

<button type="submit" class="btn btn-primary text-uppercase">

{{ \_\_('Update') }}

</button>

</div>

</div>

</div>

</div>

</div>

</form>

@endsection

@extends('admin-layout')

@section('main\_content')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Companies') }}</strong>

</div>

<div class="col-lg-6 text-end">

<a href="{{ route('admin.companies.create') }}" class="btn btn-primary">

{{ \_\_('Create') }}

</a>

</div>

</div>

</div>

<div class="card-body">

@if ($companies->count() > 0)

<table class="table table-bordered table-striped mb-0">

<thead>

<tr>

<th>{{ \_\_('Name') }}</th>

<th>{{ \_\_('Email') }}</th>

<th>{{ \_\_('Phone') }}</th>

<th>{{ \_\_('Address') }}</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach($companies as $company)

<tr>

<td class="align-middle">{{ $company->name }}</td>

<td class="align-middle">{{ $company->email }}</td>

<td class="align-middle">{{ $company->phone }}</td>

<td class="align-middle">{{ $company->address }}</td>

<td class="text-center">

<div class="btn-group">

<a class="btn btn-primary btn-sm" href="{{ route('admin.companies.edit', $company) }}">

{{ \_\_('Edit') }}

</a>

<button type="button" class="btn btn-primary btn-sm dropdown-toggle dropdown-toggle-split" data-bs-toggle="dropdown" aria-expanded="false" data-bs-reference="parent">

<span class="visually-hidden">Toggle</span>

</button>

<ul class="dropdown-menu">

<li>

<form action="{{ route('admin.companies.destroy', $company) }}" method="POST">

@csrf

@method('DELETE')

<button class="dropdown-item">{{ \_\_('Delete') }}</button>

</form>

</li>

</ul>

</div>

</td>

</tr>

@endforeach

</tbody>

</table>

@if ($companies->lastPage() > 1)

<hr>

<div class="mt-3">

{!! $companies->links('vendor.pagination.bootstrap-5') !!}

</div>

@endif

@else

<div class="text-center text-uppercase">{{ \_\_('Currently there are no companies.') }}</div>

@endif

</div>

</div>

</div>

</div>

</div>

@endsection

@extends('admin-layout')

@section('main\_content')

{{ \_\_('Hello, Administrator') }}

@endsection

@extends('admin-layout')

@section('main\_content')

<form action="{{ route('admin.invoices.store') }}" method="POST">

@csrf

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Create Invoice') }}</strong>

</div>

<div class="col-lg-6 text-end">

<a href="{{ route('admin.invoices.index') }}" class="btn btn-primary">

{{ \_\_('Return to Invoices') }}

</a>

</div>

</div>

</div>

<div class="card-body">

<div class="row mb-3">

<label for="company\_id" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Company') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<select id="company\_id" class="form-control form-select @error('company\_id') is-invalid @enderror" name="company\_id">

@foreach($companies as $company)

<option {{ old('company\_id') == $company->id ? 'selected' : '' }} value="{{ $company->id }}">{{ $company->name }}</option>

@endforeach

</select>

@error('company\_id')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="tax\_rate" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Tax Rate') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="tax\_rate" type="text" class="form-control @error('tax\_rate') is-invalid @enderror" name="tax\_rate" value="{{ old('tax\_rate') }}" autocomplete="off">

@error('tax\_rate')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="card">

<div class="card-header">{{ \_\_('Client') }}</div>

<div class="card-body">

<div class="row mb-3">

<label for="client\_type" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Type') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<select id="client\_type" class="form-control form-select @error('client\_type') is-invalid @enderror" name="client\_type">

@foreach($clientTypes as $clientTypeCode => $clientTypeTitle)

<option {{ old('client\_type') == $clientTypeCode ? 'selected' : '' }} value="{{ $clientTypeCode }}">{{ $clientTypeTitle }}</option>

@endforeach

</select>

@error('client\_type')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_name" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Name') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_name" type="text" class="form-control @error('tax\_rate') is-invalid @enderror" name="client\_name" value="{{ old('client\_name') }}" autocomplete="off">

@error('client\_name')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_email" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Email') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_email" type="text" class="form-control @error('client\_email') is-invalid @enderror" name="client\_email" value="{{ old('client\_email') }}" autocomplete="off">

@error('client\_email')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_phone" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Phone') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_phone" type="text" class="form-control @error('client\_phone') is-invalid @enderror" name="client\_phone" value="{{ old('client\_phone') }}" autocomplete="off">

@error('client\_phone')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div id="legal-entity-container" @if (old('client\_type') != 2) class="d-none" @endif>

<div class="row mb-3">

<div class="col-12">

<hr>

</div>

</div>

<div class="row mb-3">

<label for="client\_company" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Company') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_company" type="text" class="form-control @error('client\_company') is-invalid @enderror" name="client\_company" value="{{ old('client\_company') }}" autocomplete="off">

@error('client\_company')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_vat\_number" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client VAT Number') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_vat\_number" type="text" class="form-control @error('client\_vat\_number') is-invalid @enderror" name="client\_vat\_number" value="{{ old('client\_vat\_number') }}" autocomplete="off">

@error('client\_vat\_number')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_reg\_number" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Reg. Number') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_reg\_number" type="text" class="form-control @error('client\_reg\_number') is-invalid @enderror" name="client\_reg\_number" value="{{ old('client\_reg\_number') }}" autocomplete="off">

@error('client\_reg\_number')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_iban" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client IBAN') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_iban" type="text" class="form-control @error('client\_iban') is-invalid @enderror" name="client\_iban" value="{{ old('client\_iban') }}" autocomplete="off">

@error('client\_iban')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_swift" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client SWIFT') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_swift" type="text" class="form-control @error('client\_swift') is-invalid @enderror" name="client\_swift" value="{{ old('client\_swift') }}" autocomplete="off">

@error('client\_swift')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

</div>

</div>

</div>

@error('services')

<div class="alert alert-danger mt-3">

{{ $message }}

</div>

@enderror

<div class="card mt-3">

<div class="card-header">

<div class="row d-flex justify-content-center align-items-center">

<div class="col-lg-6">

{{ \_\_('Services') }}

</div>

<div class="col-lg-6 text-end">

<div class="btn btn-sm btn-success" id="add-service">

{{ \_\_('Add Service') }}

</div>

</div>

</div>

</div>

<div class="card-body" id="services-container">

@if (is\_array(old('services')) && !empty(old('services')))

@foreach(old('services') as $serviceCode => $service)

@include('admin.invoice.service', [

'identifier' => 'services',

'code' => $serviceCode,

'name' => isset($service['name']) && is\_scalar($service['name']) ? $service['name'] : '',

'price' => isset($service['price']) && is\_scalar($service['price']) ? $service['price'] : '',

])

@endforeach

@endif

</div>

</div>

</div>

<div class="card-footer text-end">

<button type="submit" class="btn btn-primary text-uppercase">

{{ \_\_('Create') }}

</button>

</div>

</div>

</div>

</div>

</div>

</form>

<template id="service-template">

@include('admin.invoice.service', ['identifier' => 'services', 'code' => '@', 'name' => '', 'price' => ''])

</template>

<script>

document.addEventListener("DOMContentLoaded", () => {

const legalEntityCode = 2;

document.addEventListener('click', (event) => {

if (event.target.classList.contains('remove-service')) {

event.target.closest('.service-container').remove();

}

});

document.querySelector('#add-service')?.addEventListener('click', () => {

const service = document.querySelector('#service-template').content.cloneNode(true);

const code = uuidv4();

service.querySelectorAll('.generic-code').forEach((elem) => {

elem.setAttribute('name', elem.getAttribute('name').replace('@', code));

});

document.querySelector('#services-container').append(service);

});

document.querySelector('#client\_type')?.addEventListener('change', (event) => {

const selectedCode = parseInt(event.target.value);

const legalEntityContainer = document.querySelector('#legal-entity-container');

if (selectedCode === legalEntityCode) {

legalEntityContainer.classList.remove('d-none');

} else {

legalEntityContainer.classList.add('d-none');

}

});

function uuidv4() {

return 'xxxxxxxx-xxxx-4xxx-yxxx-xxxxxxxxxxxx'

.replace(/[xy]/g, function (c) {

const r = Math.random() \* 16 | 0,

v = c == 'x' ? r : (r & 0x3 | 0x8);

return v.toString(16);

});

}

});

</script>

@endsection

@extends('admin-layout')

@section('main\_content')

<form action="{{ route('admin.invoices.update', $invoice) }}" method="POST">

@csrf

@method('PATCH')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Update Invoice') }}</strong>

</div>

<div class="col-lg-6 text-end">

<a href="{{ route('admin.invoices.index') }}" class="btn btn-primary">

{{ \_\_('Return to Invoices') }}

</a>

</div>

</div>

</div>

<div class="card-body">

<div class="row mb-3">

<label for="company\_id" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Company') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<select id="company\_id" class="form-control form-select @error('company\_id') is-invalid @enderror" name="company\_id">

@foreach($companies as $company)

<option {{ old('company\_id', $company->company\_id) == $company->id ? 'selected' : '' }} value="{{ $company->id }}">{{ $company->name }}</option>

@endforeach

</select>

@error('company\_id')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="tax\_rate" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Tax Rate') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="tax\_rate" type="text" class="form-control @error('tax\_rate') is-invalid @enderror" name="tax\_rate" value="{{ old('tax\_rate', $invoice->tax\_rate) }}" autocomplete="off">

@error('tax\_rate')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="card">

<div class="card-header">{{ \_\_('Client') }}</div>

<div class="card-body">

<div class="row mb-3">

<label for="client\_type" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Type') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<select id="client\_type" class="form-control form-select @error('client\_type') is-invalid @enderror" name="client\_type">

@foreach($clientTypes as $clientTypeCode => $clientTypeTitle)

<option {{ old('client\_type', $invoice->client\_type) == $clientTypeCode ? 'selected' : '' }} value="{{ $clientTypeCode }}">{{ $clientTypeTitle }}</option>

@endforeach

</select>

@error('client\_type')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_name" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Name') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_name" type="text" class="form-control @error('tax\_rate') is-invalid @enderror" name="client\_name" value="{{ old('client\_name', $invoice->client\_name) }}" autocomplete="off">

@error('client\_name')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_email" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Email') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_email" type="text" class="form-control @error('client\_email') is-invalid @enderror" name="client\_email" value="{{ old('client\_email', $invoice->client\_email) }}" autocomplete="off">

@error('client\_email')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_phone" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Phone') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_phone" type="text" class="form-control @error('client\_phone') is-invalid @enderror" name="client\_phone" value="{{ old('client\_phone', $invoice->client\_phone) }}" autocomplete="off">

@error('client\_phone')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div id="legal-entity-container" @if (old('client\_type', $invoice->client\_type) != 2) class="d-none" @endif>

<div class="row mb-3">

<div class="col-12">

<hr>

</div>

</div>

<div class="row mb-3">

<label for="client\_company" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Company') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_company" type="text" class="form-control @error('client\_company') is-invalid @enderror" name="client\_company" value="{{ old('client\_company', $invoice->client\_company) }}" autocomplete="off">

@error('client\_company')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_vat\_number" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client VAT Number') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_vat\_number" type="text" class="form-control @error('client\_vat\_number') is-invalid @enderror" name="client\_vat\_number" value="{{ old('client\_vat\_number', $invoice->client\_vat\_number) }}" autocomplete="off">

@error('client\_vat\_number')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_reg\_number" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Reg. Number') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_reg\_number" type="text" class="form-control @error('client\_reg\_number') is-invalid @enderror" name="client\_reg\_number" value="{{ old('client\_reg\_number', $invoice->client\_reg\_number) }}" autocomplete="off">

@error('client\_reg\_number')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_iban" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client IBAN') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_iban" type="text" class="form-control @error('client\_iban') is-invalid @enderror" name="client\_iban" value="{{ old('client\_iban', $invoice->client\_iban) }}" autocomplete="off">

@error('client\_iban')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="client\_swift" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client SWIFT') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="client\_swift" type="text" class="form-control @error('client\_swift') is-invalid @enderror" name="client\_swift" value="{{ old('client\_swift', $invoice->client\_swift) }}" autocomplete="off">

@error('client\_swift')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

</div>

</div>

</div>

@error('services')

<div class="alert alert-danger mt-3">

{{ $message }}

</div>

@enderror

<div class="card mt-3">

<div class="card-header">

<div class="row d-flex justify-content-center align-items-center">

<div class="col-lg-6">

{{ \_\_('Services') }}

</div>

<div class="col-lg-6 text-end">

<div class="btn btn-sm btn-success" id="add-service">

{{ \_\_('Add Service') }}

</div>

</div>

</div>

</div>

<div class="card-body" id="services-container">

@if (old('services'))

@if (is\_array(old('services')) && !empty(old('services')))

@foreach(old('services') as $serviceCode => $service)

@include('admin.invoice.service', [

'identifier' => 'services',

'code' => $serviceCode,

'name' => isset($service['name']) && is\_scalar($service['name']) ? $service['name'] : '',

'price' => isset($service['price']) && is\_scalar($service['price']) ? $service['price'] : '',

])

@endforeach

@endif

@else

@foreach($invoice->invoiceServices->reverse() as $service)

@include('admin.invoice.service', [

'identifier' => 'services',

'code' => $service->id,

'name' => $service->name,

'price' => $service->price,

])

@endforeach

@endif

</div>

</div>

</div>

<div class="card-footer text-end">

<button type="submit" class="btn btn-primary text-uppercase">

{{ \_\_('Update') }}

</button>

</div>

</div>

</div>

</div>

</div>

</form>

<template id="service-template">

@include('admin.invoice.service', ['identifier' => 'services', 'code' => '@', 'name' => '', 'price' => ''])

</template>

<script>

document.addEventListener("DOMContentLoaded", () => {

const legalEntityCode = 2;

document.addEventListener('click', (event) => {

if (event.target.classList.contains('remove-service')) {

event.target.closest('.service-container').remove();

}

});

document.querySelector('#add-service')?.addEventListener('click', () => {

const service = document.querySelector('#service-template').content.cloneNode(true);

const code = uuidv4();

service.querySelectorAll('.generic-code').forEach((elem) => {

elem.setAttribute('name', elem.getAttribute('name').replace('@', code));

});

document.querySelector('#services-container').append(service);

});

document.querySelector('#client\_type')?.addEventListener('change', (event) => {

const selectedCode = parseInt(event.target.value);

const legalEntityContainer = document.querySelector('#legal-entity-container');

if (selectedCode === legalEntityCode) {

legalEntityContainer.classList.remove('d-none');

} else {

legalEntityContainer.classList.add('d-none');

}

});

function uuidv4() {

return 'xxxxxxxx-xxxx-4xxx-yxxx-xxxxxxxxxxxx'

.replace(/[xy]/g, function (c) {

const r = Math.random() \* 16 | 0,

v = c == 'x' ? r : (r & 0x3 | 0x8);

return v.toString(16);

});

}

});

</script>

@endsection

@extends('admin-layout')

@section('main\_content')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Invoices') }}</strong>

</div>

<div class="col-lg-6 text-end">

<a href="{{ route('admin.invoices.create') }}" class="btn btn-primary">

{{ \_\_('Create') }}

</a>

</div>

</div>

</div>

<div class="card-body">

@if ($invoices->count() > 0)

<table class="table table-bordered table-striped mb-0">

<thead>

<tr>

<th>{{ \_\_('Reference') }}</th>

<th>{{ \_\_('Provider Name') }}</th>

<th>{{ \_\_('Client Name') }}</th>

<th>{{ \_\_('Paid') }}</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach($invoices as $invoice)

<tr>

<td class="align-middle">{{ $invoice->reference }}</td>

<td class="align-middle">{{ $invoice->provider\_name }}</td>

<td class="align-middle">{{ $invoice->client\_name }}</td>

<td class="align-middle text-center">{{ $invoice->paid ? '+' : '-' }}</td>

<td class="text-center">

<div class="btn-group">

<a class="btn btn-primary btn-sm" href="{{ route('admin.invoices.edit', $invoice) }}">

{{ \_\_('Edit') }}

</a>

<button type="button" class="btn btn-primary btn-sm dropdown-toggle dropdown-toggle-split" data-bs-toggle="dropdown" aria-expanded="false" data-bs-reference="parent">

<span class="visually-hidden">Toggle</span>

</button>

<ul class="dropdown-menu">

<li>

<a class="dropdown-item" target="\_blank" href="{{ route('admin.invoices.show', $invoice) }}">{{ \_\_('Show') }}</a>

</li>

<li>

<form action="{{ route('admin.invoices.destroy', $invoice) }}" method="POST">

@csrf

@method('DELETE')

<button class="dropdown-item">{{ \_\_('Delete') }}</button>

</form>

</li>

@if (!$invoice->paid)

<li>

<a class="dropdown-item" target="\_blank" href="{{ route('payment.show-form', $invoice) }}">{{ \_\_('Payment Form') }}</a>

</li>

<li>

<form action="{{ route('admin.invoices.send-email', $invoice) }}" method="POST">

@csrf

@method('POST')

<button class="dropdown-item">{{ \_\_('Send Email') }}</button>

</form>

</li>

@endif

</ul>

</div>

</td>

</tr>

@endforeach

</tbody>

</table>

@if ($invoices->lastPage() > 1)

<hr>

<div class="mt-3">

{!! $invoices->links('vendor.pagination.bootstrap-5') !!}

</div>

@endif

@else

<div class="text-center text-uppercase">{{ \_\_('Currently there are no invoices.') }}</div>

@endif

</div>

</div>

</div>

</div>

</div>

@endsection

<html>

<head>

<style>

body {

font-size: 10px;

}

.invoice-table {

border-collapse: collapse;

width: 100%;

}

.bordered-table td {

border-bottom: 1px solid #afafaf;

}

.invoice-table td, .invoice-table th {

padding: 4px 3px 4px;

}

#total-table {

margin-top: 30px;

}

.text-right {

text-align: right !important;

}

.text-left {

text-align: left !important;

}

.text-center {

text-align: center !important;

}

.w-100 {

width: 100% !important;

}

.w-65 {

width: 65% !important;

}

.w-35 {

width: 35% !important;

}

.w-50 {

width: 50% !important;

}

.w-33 {

width: 33.3% !important;

}

.w-30 {

width: 30% !important;

}

.pt-10 {

margin-top: 30px;

}

.row-title {

font-weight: bold;

}

.row-value {

color: #444;

}

.underline {

text-decoration: underline;

}

.vertical-align-top {

vertical-align: top;

}

.header-text-td {

text-align: right;

border-bottom: 1px solid #2798d2;

font-size: 14pt; color: #494949;

}

.header-image-td {

border-bottom: 1px solid #2798d2;

}

.table-top-title {

background-color: #777;

color: white;

font-weight: bold;

}

.order-divider {

background-color: #777;

padding: 0;

margin: 0;

}

.text-uppercase {

text-transform: uppercase;

}

.text-danger {

color: red;

}

</style>

</head>

<body>

<table class="invoice-table">

<tbody>

<tr>

<td class="w-100 header-text-td">

{{ $invoice->created\_at }}

<br/>

{{ \_\_('Reference') }}: <strong>{{ $invoice->reference }}</strong>

</td>

</tr>

</tbody>

</table>

<table class="w-100">

<tbody>

<tr>

<td class="w-33 vertical-align-top">

<h3 class="text-center">{{ \_\_('General Info') }}</h3>

<table id="invoice-info-table" class="invoice-table">

<tbody>

<tr>

<td class="text-right w-50 row-title">{{ \_\_('Invoice Date') }}:</td>

<td class="text-left w-50 row-value">{{ $invoice->created\_at }}</td>

</tr>

<tr style="border-left: 1px solid #afafaf; border-top: 1px solid #afafaf; border-right: 1px solid #afafaf;">

<td class="text-right w-50 row-title">{{ \_\_('Total') }}:</td>

<td class="text-left w-50 row-value">{{ number\_format($invoice->getTotalPriceWithTax(), $invoice::PRICE\_ROUNDING, '.', '') }}</td>

</tr>

<tr style="border-left: 1px solid #afafaf; border-bottom: 1px solid #afafaf; border-right: 1px solid #afafaf;">

<td class="text-right pt-10 w-50 row-title">{{ \_\_('Payment Goal') }}:</td>

<td class="text-left w-50 row-value">{{ $invoice->reference }}</td>

</tr>

</tbody>

</table>

</td>

<td class="w-33 vertical-align-top">

<h3 class="text-center">PIEGĀDĀTĀJS</h3>

<table id="company-table" class="invoice-table">

<tbody>

<tr>

<td class="text-right w-50 row-title">Uzņēmums:</td>

<td class="text-left w-50 row-value">{{ $invoice->provider\_name }}</td>

</tr>

<tr>

<td class="text-right w-50 row-title">Reģ. Nr.:</td>

<td class="text-left w-50 row-value">{{ $invoice->provider\_reg\_number }}</td>

</tr>

<tr>

<td class="text-right w-50 row-title">PVN Nr.:</td>

<td class="text-left w-50 row-value">{{ $invoice->provider\_vat\_number }}</td>

</tr>

<tr>

<td class="text-right w-50 row-title">SWIFT:</td>

<td class="text-left w-50 row-value"></td>

</tr>

<tr>

<td class="text-right w-50 row-title">IBAN:</td>

<td class="text-left w-50 row-value"></td>

</tr>

</tbody>

</table>

</td>

<td class="w-33 vertical-align-top">

<h3 class="text-center">SAŅĒMĒJS</h3>

<table id="company-table" class="invoice-table">

<tbody>

<tr>

<td class="text-right row-title">Vārds, uzvārds:</td>

<td class="text-left row-value">{{ $invoice->client\_name }}</td>

</tr>

<tr>

<td class="text-right row-title">E-pasts:</td>

<td class="text-left row-value">{{ $invoice->client\_email }}</td>

</tr>

@if ($invoice->isLegalEntityInvoice())

<tr>

<td class="text-right row-title">Kompānija:</td>

<td class="text-left row-value">{{ $invoice->client\_company }}</td>

</tr>

<tr>

<td class="text-right row-title">Reģistrācijas numurs:</td>

<td class="text-left row-value">{{ $invoice->client\_reg\_number }}</td>

</tr>

<tr>

<td class="text-right row-title">PVN numurs:</td>

<td class="text-left row-value">{{ $invoice->client\_vat\_number }}</td>

</tr>

<tr>

<td class="text-right row-title">SWIFT:</td>

<td class="text-left row-value">{{ $invoice->client\_swift }}</td>

</tr>

<tr>

<td class="text-right row-title">IBAN:</td>

<td class="text-left row-value">{{ $invoice->client\_iban }}</td>

</tr>

@endif

</tbody>

</table>

</td>

<tr>

</tbody>

</table>

<table id="common-table" class="invoice-table bordered-table">

<thead>

<tr>

<th class="table-top-title w-30">{{ \_\_('Name') }}</th>

<th class="table-top-title">{{ \_\_('Price Tax Excl.') }}</th>

<th class="table-top-title">{{ \_\_('Tax Rate') }}</th>

</tr>

</thead>

<tbody>

@foreach ($invoice->invoiceServices as $service)

<tr>

<td class="text-center">{{ $service->name }}</td>

<td class="text-center">{{ number\_format($service->price, $invoice::PRICE\_ROUNDING, '.', '') }}</td>

<td class="text-center">{{ $invoice->tax\_rate }} %</td>

</tr>

@endforeach

</tbody>

</table>

<table class="w-100">

<tbody>

<tr>

<td class="w-65"></td>

<td class="w-35">

<table id="total-table" class="invoice-table bordered-table">

<tbody>

<tr>

<td class="w-50 text-left">Kopā bez PVN</td>

<td class="w-50 text-right">{{ number\_format($invoice->getTotalPriceWithoutTax(), $invoice::PRICE\_ROUNDING, '.', '') }}</td>

</tr>

<tr>

<td class="w-50 text-left">Kopā ar PVN</td>

<td class="w-50 text-right" style="background-color: #dddddd;"><strong>{{ number\_format($invoice->getTotalPriceWithTax(), $invoice::PRICE\_ROUNDING, '.', '') }}</strong></td>

</tr>

</tbody>

</table>

</td>

</tr>

</tbody>

</table>

</body>

</html>

<div class="card mb-2 service-container">

<div class="card-body">

<div class="row d-flex justify-content-center align-items-center">

<div class="@if (empty($hideDelete)) col-md-5 @else col-md-6 @endif">

<div class="form-floating">

<input type="text" class="form-control form-control-sm generic-code @error(sprintf('%s.%s.name', $identifier, $code)) is-invalid @enderror" name="{{ $identifier }}[{{ $code }}][name]" value="{{ $name }}">

@error(sprintf('%s.%s.name', $identifier, $code))

<span class="invalid-feedback" role="alert">

{{ $message }}

</span>

@enderror

<label>{{ \_\_('Name') }}</label>

</div>

</div>

<div class="@if (empty($hideDelete)) col-md-5 @else col-md-6 @endif">

<div class="form-floating">

<input type="text" class="form-control form-control-sm generic-code @error(sprintf('%s.%s.price', $identifier, $code)) is-invalid @enderror" name="{{ $identifier }}[{{ $code }}][price]" value="{{ $price }}">

@error(sprintf('%s.%s.price', $identifier, $code))

<span class="invalid-feedback" role="alert">

{{ $message }}

</span>

@enderror

<label>{{ \_\_('Price') }}</label>

</div>

</div>

<div class="col-md-2">

@if (empty($hideDelete))

<div class="btn btn-sm btn-danger w-100 remove-service">

{{ \_\_('Delete') }}

</div>

@endif

</div>

</div>

</div>

</div>

@extends('admin-layout')

@section('main\_content')

<form action="{{ route('admin.posts.store') }}" enctype="multipart/form-data" method="POST">

@csrf

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Create Post') }}</strong>

</div>

<div class="col-lg-6 text-end">

<a href="{{ route('admin.posts.index') }}" class="btn btn-primary">

{{ \_\_('Return to Posts') }}

</a>

</div>

</div>

</div>

<div class="card-body">

<div class="row mb-3">

<label for="title" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Title') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="title" type="text" class="form-control @error('title') is-invalid @enderror" name="title" value="{{ old('title') }}" autocomplete="off">

@error('title')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="slug" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Slug') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="slug" type="text" class="form-control @error('slug') is-invalid @enderror" name="slug" value="{{ old('slug') }}" autocomplete="off">

@error('slug')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="image" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Image') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="image" type="file" class="form-control @error('image') is-invalid @enderror" name="image" autocomplete="off">

@error('image')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="text" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Text') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<textarea id="text" class="form-control @error('text') is-invalid @enderror" name="text" autocomplete="off">{{ old('title') }}</textarea>

@error('text')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="active" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Active') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<select class="form-select @error('active') is-invalid @enderror" autocomplete="off" name="active">

<option value="0" {{ !old('active') ? 'selected' : '' }}>{{ \_\_('No') }}</option>

<option value="1" {{ old('active') ? 'selected' : '' }}>{{ \_\_('Yes') }}</option>

</select>

@error('active')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

</div>

<div class="card-footer text-end">

<button type="submit" class="btn btn-primary text-uppercase">

{{ \_\_('Create') }}

</button>

</div>

</div>

</div>

</div>

</div>

</form>

@endsection

@extends('admin-layout')

@section('main\_content')

<form action="{{ route('admin.posts.update', $post) }}" enctype="multipart/form-data" method="POST">

@csrf

@method('PATCH')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Update Post') }}</strong>

</div>

<div class="col-lg-6 text-end">

<a href="{{ route('admin.posts.index') }}" class="btn btn-primary">

{{ \_\_('Return to Posts') }}

</a>

</div>

</div>

</div>

<div class="card-body">

<div class="row mb-3">

<label for="title" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Title') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="title" type="text" class="form-control @error('title') is-invalid @enderror" name="title" value="{{ old('title', $post->title) }}" autocomplete="off">

@error('title')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="slug" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Slug') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<input id="slug" type="text" class="form-control @error('slug') is-invalid @enderror" name="slug" value="{{ old('slug', $post->slug) }}" autocomplete="off">

@error('slug')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="image" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Image') }}</label>

<div class="col-lg-6 mb-2">

<input id="image" type="file" class="form-control @error('image') is-invalid @enderror" name="image" autocomplete="off">

@error('image')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

<div class="col-lg-6 offset-lg-3">

<img src="{{ $post->getImageFullUrl() }}" class="img-fluid img-thumbnail rounded" alt="image">

</div>

</div>

<div class="row mb-3">

<label for="text" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Text') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<textarea id="text" class="form-control @error('text') is-invalid @enderror" name="text" autocomplete="off">{{ old('title', $post->text) }}</textarea>

@error('text')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="active" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Active') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<select class="form-select @error('active') is-invalid @enderror" autocomplete="off" name="active">

<option value="0" {{ !old('active', !$post->active) ? 'selected' : '' }}>{{ \_\_('No') }}</option>

<option value="1" {{ old('active', $post->active) ? 'selected' : '' }}>{{ \_\_('Yes') }}</option>

</select>

@error('active')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

</div>

<div class="card-footer text-end">

<button type="submit" class="btn btn-primary text-uppercase">

{{ \_\_('Update') }}

</button>

</div>

</div>

</div>

</div>

</div>

</form>

@endsection

@extends('admin-layout')

@section('main\_content')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Posts') }}</strong>

</div>

<div class="col-lg-6 text-end">

<a href="{{ route('admin.posts.create') }}" class="btn btn-primary">

{{ \_\_('Create') }}

</a>

</div>

</div>

</div>

<div class="card-body">

@if ($posts->count() > 0)

<table class="table table-bordered table-striped mb-0">

<thead>

<tr>

<th>{{ \_\_('Title') }}</th>

<th>{{ \_\_('Slug') }}</th>

<th>{{ \_\_('Author') }}</th>

<th class="text-center">{{ \_\_('Active') }}</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach($posts as $post)

<tr>

<td class="align-middle">{{ $post->title }}</td>

<td class="align-middle">{{ $post->slug }}</td>

<td class="align-middle">{{ $post->user?->name ?? '...' }}</td>

<td class="align-middle text-center">

@if ($post->active)

<span class="badge bg-success text-uppercase">{{ \_\_('active') }}</span>

@else

<span class="badge bg-danger text-uppercase">{{ \_\_('disabled') }}</span>

@endif

</td>

<td class="text-center">

<div class="btn-group">

<a class="btn btn-primary btn-sm" href="{{ route('admin.posts.edit', $post) }}">

{{ \_\_('Edit') }}

</a>

<button type="button" class="btn btn-primary btn-sm dropdown-toggle dropdown-toggle-split" data-bs-toggle="dropdown" aria-expanded="false" data-bs-reference="parent">

<span class="visually-hidden">Toggle</span>

</button>

<ul class="dropdown-menu">

<li>

<form action="{{ route('admin.posts.destroy', $post) }}" method="POST">

@csrf

@method('DELETE')

<button class="dropdown-item">{{ \_\_('Delete') }}</button>

</form>

</li>

</ul>

</div>

</td>

</tr>

@endforeach

</tbody>

</table>

@if ($posts->lastPage() > 1)

<hr>

<div class="mt-3">

{!! $posts->links('vendor.pagination.bootstrap-5') !!}

</div>

@endif

@else

<div class="text-center text-uppercase">{{ \_\_('Currently there are no posts.') }}</div>

@endif

</div>

</div>

</div>

</div>

</div>

@endsection

@extends('admin-layout')

@section('main\_content')

<form action="{{ route('admin.questions.update', $question) }}" enctype="multipart/form-data" method="POST">

@csrf

@method('PATCH')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Update Question') }}</strong>

</div>

<div class="col-lg-6 text-end">

<a href="{{ route('admin.questions.index') }}" class="btn btn-primary">

{{ \_\_('Return to Questions') }}

</a>

</div>

</div>

</div>

<div class="card-body">

<div class="row mb-3">

<label for="email" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Email') }}</label>

<div class="col-lg-6">

<input id="email" type="text" class="form-control" name="email" value="{{ $question->email }}" disabled>

</div>

</div>

<div class="row mb-3">

<label for="ip" class="col-lg-3 col-form-label text-md-end">{{ \_\_('IP') }}</label>

<div class="col-lg-6">

<input id="ip" type="text" class="form-control" name="ip" value="{{ $question->ip }}" disabled>

</div>

</div>

<div class="row mb-3">

<label for="question\_text" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Question') }}</label>

<div class="col-lg-6">

<textarea id="question\_text" class="form-control" name="question\_text" disabled>{{ $question->question\_text }}</textarea>

</div>

</div>

<div class="row mb-3">

<label for="verified" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Verified') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<select class="form-select @error('verified') is-invalid @enderror" autocomplete="off" name="verified">

<option value="0" {{ !old('verified', !$question->verified) ? 'selected' : '' }}>{{ \_\_('No') }}</option>

<option value="1" {{ old('verified', $question->verified) ? 'selected' : '' }}>{{ \_\_('Yes') }}</option>

</select>

@error('verified')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

</div>

<div class="card-footer text-end">

<button type="submit" class="btn btn-primary text-uppercase">

{{ \_\_('Update') }}

</button>

</div>

</div>

</div>

</div>

</div>

</form>

<hr>

<form action="{{ route('admin.questions.response', $question) }}" enctype="multipart/form-data" method="POST">

@csrf

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Send response') }}</strong>

</div>

</div>

</div>

<div class="card-body">

<div class="row mb-3">

<label for="response" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Response') }}<span class="text-danger">\*</span></label>

<div class="col-lg-6">

<textarea id="response" class="form-control @error('response') is-invalid @enderror" name="response">{{ old("response") }}</textarea>

@error('response')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

</div>

<div class="card-footer text-end">

<button type="submit" class="btn btn-primary text-uppercase">

{{ \_\_('Send') }}

</button>

</div>

</div>

</div>

</div>

</div>

</form>

@endsection

@extends('admin-layout')

@section('main\_content')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<strong>{{ \_\_('Questions') }}</strong>

</div>

<div class="card-body">

@if ($questions->count() > 0)

<table class="table table-bordered table-striped mb-0">

<thead>

<tr>

<th>{{ \_\_('Email') }}</th>

<th>{{ \_\_('IP') }}</th>

<th>{{ \_\_('Question') }}</th>

<th class="text-center">{{ \_\_('Verified') }}</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach($questions as $question)

<tr>

<td class="align-middle">{{ $question->email }}</td>

<td class="align-middle">{{ $question->ip }}</td>

<td class="align-middle">{{ Str::limit($question->question\_text, 10) }}</td>

<td class="align-middle text-center">

@if ($question->verified)

<span class="badge bg-success text-uppercase">{{ \_\_('verified') }}</span>

@else

<span class="badge bg-danger text-uppercase">{{ \_\_('not verified') }}</span>

@endif

</td>

<td class="text-center">

<div class="btn-group">

<a class="btn btn-primary btn-sm" href="{{ route('admin.questions.edit', $question) }}">

{{ \_\_('Edit') }}

</a>

<button type="button" class="btn btn-primary btn-sm dropdown-toggle dropdown-toggle-split" data-bs-toggle="dropdown" aria-expanded="false" data-bs-reference="parent">

<span class="visually-hidden">Toggle</span>

</button>

<ul class="dropdown-menu">

<li>

<form action="{{ route('admin.questions.destroy', $question) }}" method="POST">

@csrf

@method('DELETE')

<button class="dropdown-item">{{ \_\_('Delete') }}</button>

</form>

</li>

</ul>

</div>

</td>

</tr>

@endforeach

</tbody>

</table>

@if ($questions->lastPage() > 1)

<hr>

<div class="mt-3">

{!! $questions->links('vendor.pagination.bootstrap-5') !!}

</div>

@endif

@else

<div class="text-center text-uppercase">{{ \_\_('Currently there are no questions.') }}</div>

@endif

</div>

</div>

</div>

</div>

</div>

@endsection

@extends("admin-layout")

@section('title') Account management @endsection

@section("main\_content")

<h1>PRIVET</h1>

@endsection

@extends('layouts.app')

@section('content')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header">{{ \_\_('Confirm Password') }}</div>

<div class="card-body">

{{ \_\_('Please confirm your password before continuing.') }}

<form method="POST" action="{{ route('password.confirm') }}">

@csrf

<div class="row mb-3">

<label for="password" class="col-md-4 col-form-label text-md-end">{{ \_\_('Password') }}</label>

<div class="col-md-6">

<input id="password" type="password" class="form-control @error('password') is-invalid @enderror" name="password" required autocomplete="current-password">

@error('password')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-0">

<div class="col-md-8 offset-md-4">

<button type="submit" class="btn btn-primary">

{{ \_\_('Confirm Password') }}

</button>

@if (Route::has('password.request'))

<a class="btn btn-link" href="{{ route('password.request') }}">

{{ \_\_('Forgot Your Password?') }}

</a>

@endif

</div>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

@endsection

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Reset password</title>

<link rel="stylesheet" href="{{asset("login-assets/styles.css")}}">

<link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>

</head>

<body>

<div class="wrapper">

<form method="POST" action="{{ route('password.email') }}">

@csrf

<h1>Reset password</h1>

<div class="input-box">

<input type="text" name="email" placeholder="Email" required>

<i class='bx bxs-user'></i>

@error('email')

<strong>{{ $message }}</strong>

@enderror

</div>

<button type="submit" class="btn">Send Password Reset Link</button>

<div class="main-link">

<p>Return to <a href="{{route("main")}}">main page</a></p>

</div>

</form>

</div>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Reset password</title>

<link rel="stylesheet" href="{{asset("login-assets/styles.css")}}">

<link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>

</head>

<body>

<div class="wrapper">

<form method="POST" action="{{ route('password.update') }}">

@csrf

<input type="hidden" name="token" value="{{ $token }}">

<h1>Reset password</h1>

<div class="input-box">

<input id="email" type="email" class="form-control @error('email') is-invalid @enderror" name="email" value="{{ $email ?? old('email') }}" required autocomplete="email" autofocus>

<i class='bx bxs-user'></i>

@error('email')

<strong>{{ $message }}</strong>

@enderror

</div>

<div class="input-box">

<input id="password" type="password" class="form-control @error('password') is-invalid @enderror" name="password" required autocomplete="new-password" placeholder="password">

<i class='bx bxs-lock-alt' ></i>

@error('password')

<strong>{{ $message }}</strong>

@enderror

</div>

<div class="input-box">

<input id="password-confirm" type="password" class="form-control" name="password\_confirmation" required autocomplete="new-password" placeholder="password confirmation">

<i class='bx bxs-lock-alt' ></i>

</div>

<button type="submit" class="btn">Reset Password</button>

<div class="main-link">

<p>Return to <a href="{{route("main")}}">main page</a></p>

</div>

</form>

</div>

</body>

</html>

@extends('layout')

@section('content')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header">{{ \_\_('Reset Password') }}</div>

<div class="card-body">

<form method="POST" action="{{ route('password.update') }}">

@csrf

<input type="hidden" name="token" value="{{ $token }}">

<div class="row mb-3">

<label for="email"

class="col-md-4 col-form-label text-md-end">{{ \_\_('Email Address') }}</label>

<div class="col-md-6">

<input id="email" type="email"

class="form-control @error('email') is-invalid @enderror" name="email"

value="{{ $email ?? old('email') }}" required autocomplete="email" autofocus>

@error('email')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="password"

class="col-md-4 col-form-label text-md-end">{{ \_\_('Password') }}</label>

<div class="col-md-6">

<input id="password" type="password"

class="form-control @error('password') is-invalid @enderror" name="password"

required autocomplete="new-password">

@error('password')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="password-confirm"

class="col-md-4 col-form-label text-md-end">{{ \_\_('Confirm Password') }}</label>

<div class="col-md-6">

<input id="password-confirm" type="password" class="form-control"

name="password\_confirmation" required autocomplete="new-password">

</div>

</div>

<div class="row mb-0">

<div class="col-md-6 offset-md-4">

<button type="submit" class="btn btn-primary">

{{ \_\_('Reset Password') }}

</button>

</div>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

@endsection

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Login</title>

<link rel="stylesheet" href="{{asset("login-assets/styles.css")}}">

<link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>

</head>

<body>

<div class="wrapper">

<form method="POST" action="{{ route('login') }}">

@csrf

<h1>Login</h1>

<div class="input-box">

<input placeholder="Email" type="email" name="email" value="{{ old('email') }}" required>

<i class='bx bxs-user'></i>

@error('email')

<hr>

<span>{{ $message }}</span>

@enderror

</div>

<div class="input-box">

<input type="password" name="password" placeholder="Password" required>

<i class='bx bxs-lock-alt' ></i>

@error('password')

<hr>

<span>{{ $message }}</span>

@enderror

</div>

<div class="remember-forgot">

<label><input id="remember\_me" name="remember\_me" type="checkbox">Remember Me</label>

<a href="{{ route('password.request') }}">Forgot Password</a>

</div>

<button type="submit" class="btn">Login</button>

<div class="main-link">

<p>Return to <a href="{{route("main")}}">main page</a></p>

</div>

</form>

</div>

</body>

</html>

@extends('layouts.app')

@section('content')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header">{{ \_\_('Register') }}</div>

<div class="card-body">

<form method="POST" action="{{ route('register') }}">

@csrf

<div class="row mb-3">

<label for="name" class="col-md-4 col-form-label text-md-end">{{ \_\_('Name') }}</label>

<div class="col-md-6">

<input id="name" type="text" class="form-control @error('name') is-invalid @enderror" name="name" value="{{ old('name') }}" required autocomplete="name" autofocus>

@error('name')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="email" class="col-md-4 col-form-label text-md-end">{{ \_\_('Email Address') }}</label>

<div class="col-md-6">

<input id="email" type="email" class="form-control @error('email') is-invalid @enderror" name="email" value="{{ old('email') }}" required autocomplete="email">

@error('email')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="password" class="col-md-4 col-form-label text-md-end">{{ \_\_('Password') }}</label>

<div class="col-md-6">

<input id="password" type="password" class="form-control @error('password') is-invalid @enderror" name="password" required autocomplete="new-password">

@error('password')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="password-confirm" class="col-md-4 col-form-label text-md-end">{{ \_\_('Confirm Password') }}</label>

<div class="col-md-6">

<input id="password-confirm" type="password" class="form-control" name="password\_confirmation" required autocomplete="new-password">

</div>

</div>

<div class="row mb-0">

<div class="col-md-6 offset-md-4">

<button type="submit" class="btn btn-primary">

{{ \_\_('Register') }}

</button>

</div>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

@endsection

@extends('layouts.app')

@section('content')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header">{{ \_\_('Verify Your Email Address') }}</div>

<div class="card-body">

@if (session('resent'))

<div class="alert alert-success" role="alert">

{{ \_\_('A fresh verification link has been sent to your email address.') }}

</div>

@endif

{{ \_\_('Before proceeding, please check your email for a verification link.') }}

{{ \_\_('If you did not receive the email') }},

<form class="d-inline" method="POST" action="{{ route('verification.resend') }}">

@csrf

<button type="submit" class="btn btn-link p-0 m-0 align-baseline">{{ \_\_('click here to request another') }}</button>.

</form>

</div>

</div>

</div>

</div>

</div>

@endsection

@extends('layout')

@section('content')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="row">

@foreach($posts as $post)

<div class="col-lg-4 mt-4">

<div class="card text-center text-bg-light h-100">

<img class="card-img-top" src="{{ $post->getImageFullUrl() }}" alt="image">

<div class="card-body d-flex align-items-center flex-column">

<h5 class="card-title">{{ $post->title }}</h5>

<p class="card-text mb-auto text-start mb-2">{{ Str::limit(strip\_tags($post->text), 100) }}</p>

<a href="{{ route('blog.show', $post) }}"

class="btn btn-primary btn-sm">{{ \_\_('Read More') }}</a>

</div>

<div class="card-footer text-muted">

{{ $post->created\_at }}

</div>

</div>

</div>

@endforeach

</div>

</div>

@if ($posts->lastPage() > 1)

<div class="col-md-8">

<hr>

{!! $posts->links('vendor.pagination.bootstrap-5') !!}

</div>

@endif

</div>

</div>

@endsection

@extends('layout')

@section('content')

<br>

<br>

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<h2 class="display-4 fw-normal text-center mb-4">{{ $post->title }}</h2>

<div class="card mt-3">

<div class="card-header text-center">

{{ $post->created\_at }}

</div>

<div class="card-body p-5 lead">

<img class="float-start img-fluid pe-3" src="{{ $post->getImageFullUrl() }}" alt="image">

{!! nl2br($post->text) !!}

</div>

</div>

</div>

</div>

</div>

<br>

<br>

<br>

@endsection

@extends('admin-company-layout')

@section('main\_content')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Invoices') }}</strong>

</div>

</div>

</div>

<div class="card-body">

@if ($invoices->count() > 0)

<table class="table table-bordered table-striped mb-0">

<thead>

<tr>

<th>{{ \_\_('Reference') }}</th>

<th>{{ \_\_('Provider Name') }}</th>

<th>{{ \_\_('Client Name') }}</th>

<th>{{ \_\_('Paid') }}</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach($invoices as $invoice)

<tr>

<td class="align-middle">{{ $invoice->reference }}</td>

<td class="align-middle">{{ $invoice->provider\_name }}</td>

<td class="align-middle">{{ $invoice->client\_name }}</td>

<td class="align-middle text-center">{{ $invoice->paid ? '+' : '-' }}</td>

<td class="text-center">

<div class="btn-group">

<a class="btn btn-primary btn-sm" href="{{ route('companies-admin.show', $invoice) }}">

{{ \_\_('Show') }}

</a>

<button type="button" class="btn btn-primary btn-sm dropdown-toggle dropdown-toggle-split" data-bs-toggle="dropdown" aria-expanded="false" data-bs-reference="parent">

<span class="visually-hidden">Toggle</span>

</button>

<ul class="dropdown-menu">

<li>

<a class="dropdown-item" target="\_blank" href="{{ route('companies-admin.show-pdf', $invoice) }}">{{ \_\_('PDF') }}</a>

</li>

</ul>

</div>

</td>

</tr>

@endforeach

</tbody>

</table>

@if ($invoices->lastPage() > 1)

<hr>

<div class="mt-3">

{!! $invoices->links('vendor.pagination.bootstrap-5') !!}

</div>

@endif

@else

<div class="text-center text-uppercase">{{ \_\_('Currently there are no invoices.') }}</div>

@endif

</div>

</div>

</div>

</div>

</div>

@endsection

@extends('admin-company-layout')

@section('main\_content')

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<div class="row">

<div class="col-lg-6 d-flex align-items-center">

<strong>{{ \_\_('Show Invoice') }}</strong>

</div>

<div class="col-lg-6 text-end">

<a href="{{ route('companies-admin.index') }}" class="btn btn-primary">

{{ \_\_('Return to Invoices') }}

</a>

</div>

</div>

</div>

<div class="card-body">

<form>

<fieldset disabled>

<div class="row mb-3">

<label for="company\_id" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Company') }}</label>

<div class="col-lg-6">

<select id="company\_id" class="form-control form-select" name="company\_id">

@foreach($companies as $company)

<option {{ $company->company\_id == $company->id ? 'selected' : '' }} value="{{ $company->id }}">{{ $company->name }}</option>

@endforeach

</select>

</div>

</div>

<div class="row mb-3">

<label for="tax\_rate" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Tax Rate') }}</label>

<div class="col-lg-6">

<input id="tax\_rate" type="text" class="form-control" name="tax\_rate" value="{{ $invoice->tax\_rate }}" autocomplete="off">

</div>

</div>

<div class="card">

<div class="card-header">{{ \_\_('Client') }}</div>

<div class="card-body">

<div class="row mb-3">

<label for="client\_type" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Type') }}</label>

<div class="col-lg-6">

<select id="client\_type" class="form-control form-select" name="client\_type">

@foreach($clientTypes as $clientTypeCode => $clientTypeTitle)

<option {{ $invoice->client\_type == $clientTypeCode ? 'selected' : '' }} value="{{ $clientTypeCode }}">{{ $clientTypeTitle }}</option>

@endforeach

</select>

</div>

</div>

<div class="row mb-3">

<label for="client\_name" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Name') }}</label>

<div class="col-lg-6">

<input id="client\_name" type="text" class="form-control" name="client\_name" value="{{ $invoice->client\_name }}" autocomplete="off">

</div>

</div>

<div class="row mb-3">

<label for="client\_email" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Email') }}</label>

<div class="col-lg-6">

<input id="client\_email" type="text" class="form-control" name="client\_email" value="{{ $invoice->client\_email }}" autocomplete="off">

</div>

</div>

<div class="row mb-3">

<label for="client\_phone" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Phone') }}</label>

<div class="col-lg-6">

<input id="client\_phone" type="text" class="form-control" name="client\_phone" value="{{ $invoice->client\_phone }}" autocomplete="off">

</div>

</div>

<div id="legal-entity-container" @if ($invoice->client\_type != 2) class="d-none" @endif>

<div class="row mb-3">

<div class="col-12">

<hr>

</div>

</div>

<div class="row mb-3">

<label for="client\_company" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Company') }}</label>

<div class="col-lg-6">

<input id="client\_company" type="text" class="form-control" name="client\_company" value="{{ $invoice->client\_company }}" autocomplete="off">

</div>

</div>

<div class="row mb-3">

<label for="client\_vat\_number" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client VAT Number') }}</label>

<div class="col-lg-6">

<input id="client\_vat\_number" type="text" class="form-control" name="client\_vat\_number" value="{{ $invoice->client\_vat\_number }}" autocomplete="off">

</div>

</div>

<div class="row mb-3">

<label for="client\_reg\_number" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client Reg. Number') }}</label>

<div class="col-lg-6">

<input id="client\_reg\_number" type="text" class="form-control" name="client\_reg\_number" value="{{ $invoice->client\_reg\_number }}" autocomplete="off">

</div>

</div>

<div class="row mb-3">

<label for="client\_iban" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client IBAN') }}</label>

<div class="col-lg-6">

<input id="client\_iban" type="text" class="form-control" name="client\_iban" value="{{ $invoice->client\_iban }}" autocomplete="off">

</div>

</div>

<div class="row mb-3">

<label for="client\_swift" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Client SWIFT') }}</label>

<div class="col-lg-6">

<input id="client\_swift" type="text" class="form-control" name="client\_swift" value="{{ $invoice->client\_swift }}" autocomplete="off">

@error('client\_swift')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

</div>

</div>

</div>

<div class="card mt-3">

<div class="card-header">

<div class="row d-flex align-items-center">

<div class="col-lg-6">

{{ \_\_('Services') }}

</div>

</div>

</div>

<div class="card-body" id="services-container">

@foreach($invoice->invoiceServices->reverse() as $service)

@include('admin.invoice.service', [

'identifier' => 'services',

'code' => $service->id,

'name' => $service->name,

'price' => $service->price,

'hideDelete' => true

])

@endforeach

</div>

</div>

</fieldset>

</form>

</div>

</div>

</div>

</div>

</div>

@endsection

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Reset password</title>

<link rel="stylesheet" href="{{asset("login-assets/styles.css")}}">

<link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>

</head>

<body>

<div class="wrapper">

<form method="POST" action="{{ route('companies.send\_password\_reset\_link\_email') }}">

@csrf

<h1>Reset password</h1>

<div class="input-box">

<input type="text" name="email" placeholder="Email" required>

<i class='bx bxs-user'></i>

@error('email')

<strong>{{ $message }}</strong>

@enderror

</div>

<button type="submit" class="btn">Send Password Reset Link</button>

<div class="main-link">

<p>Return to <a href="{{route("main")}}">main page</a></p>

</div>

</form>

</div>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Reset password</title>

<link rel="stylesheet" href="{{asset("login-assets/styles.css")}}">

<link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>

</head>

<body>

<div class="wrapper">

<form method="POST" action="{{ route('companies.reset\_password') }}">

@csrf

<input type="hidden" name="token" value="{{ $token }}">

<h1>Reset password</h1>

<div class="input-box">

<input id="email" type="email" class="form-control @error('email') is-invalid @enderror" name="email" value="{{ $email ?? old('email') }}" required autocomplete="email" autofocus>

<i class='bx bxs-user'></i>

@error('email')

<strong>{{ $message }}</strong>

@enderror

</div>

<div class="input-box">

<input id="password" type="password" class="form-control @error('password') is-invalid @enderror" name="password" required autocomplete="new-password" placeholder="password">

<i class='bx bxs-lock-alt' ></i>

@error('password')

<strong>{{ $message }}</strong>

@enderror

</div>

<div class="input-box">

<input id="password-confirm" type="password" class="form-control" name="password\_confirmation" required autocomplete="new-password" placeholder="password confirmation">

<i class='bx bxs-lock-alt' ></i>

</div>

<button type="submit" class="btn">Reset Password</button>

<div class="main-link">

<p>Return to <a href="{{route("main")}}">main page</a></p>

</div>

</form>

</div>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Login</title>

<link rel="stylesheet" href="{{asset("login-assets/styles.css")}}">

<link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>

</head>

<body>

<div class="wrapper">

<form method="POST" action="{{ route('companies.login') }}">

@csrf

<h1>Login</h1>

<div class="input-box">

<input placeholder="Email" type="email" name="email" value="{{ old('email') }}" required>

<i class='bx bxs-user'></i>

@error('email')

<hr>

<span>{{ $message }}</span>

@enderror

</div>

<div class="input-box">

<input type="password" name="password" placeholder="Password" required>

<i class='bx bxs-lock-alt' ></i>

@error('password')

<hr>

<span>{{ $message }}</span>

@enderror

</div>

<div class="remember-forgot">

<label><input id="remember\_me" name="remember\_me" type="checkbox">Remember Me</label>

<a href="{{ route('companies.show\_password\_link\_request\_form') }}">Forgot Password</a>

</div>

<button type="submit" class="btn">Login</button>

<div class="main-link">

<p>Return to <a href="{{route("main")}}">main page</a></p>

</div>

</form>

</div>

</body>

</html>

{{-- @extends('layout')

@section('content')

<form action="{{ route('contacts.submit') }}" method="POST">

@csrf

@method('POST')

<div class="container">

<div class="row justify-content-center">

@if (session()->has('error'))

<div class="col-md-8">

<div class="alert alert-danger">

{{ session('error') }}

</div>

</div>

@endif

@if (session()->has('success'))

<div class="col-md-8">

<div class="alert alert-success">

{{ session('success') }}

</div>

</div>

@endif

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<strong>{{ \_\_('Have any questions ?') }}</strong>

</div>

<div class="card-body">

<div class="row mb-3">

<label for="email" class="col-lg-3 col-form-label text-md-end">{{ \_\_('Email') }}<span

class="text-danger"></span></label>

<div class="col-lg-6">

<input id="email" type="text"

class="form-control @error('email') is-invalid @enderror" name="email"

value="{{ old('email') }}" autocomplete="off">

@error('email')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

<div class="row mb-3">

<label for="question\_text"

class="col-lg-3 col-form-label text-md-end">{{ \_\_('Question') }}<span

class="text-danger"></span></label>

<div class="col-lg-6">

<textarea id="question\_text"

class="form-control @error('question\_text') is-invalid @enderror"

name="question\_text">{{ old('question\_text') }}</textarea>

@error('question\_text')

<span class="invalid-feedback" role="alert">

<strong>{{ $message }}</strong>

</span>

@enderror

</div>

</div>

</div>

<div class="card-footer text-end">

<button type="submit" class="btn btn-primary text-uppercase">

{{ \_\_('Submit') }}

</button>

</div>

</div>

</div>

</div>

</div>

<style>

.container {

max-width: 960px;

margin: 0 auto;

padding: 15px;

}

.card {

box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2);

transition: 0.3s;

border-radius: 5px;

}

.card:hover {

box-shadow: 0 8px 16px 0 rgba(0, 0, 0, 0.2);

}

.card-header {

background-color: #f1f1f1;

padding: 10px;

text-align: center;

font-size: 18px;

font-weight: bold;

}

.card-body {

padding: 10px;

}

.btn-primary {

background-color: #9d44e5;

border: none;

color: white;

padding: 15px 32px;

text-align: center;

text-decoration: none;

display: inline-block;

font-size: 16px;

margin: 6px 3px;

cursor: pointer;

}

</style>

</form>

@endsection --}}

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Questions</title>

<link rel="stylesheet" href="{{asset("login-assets/styles.css")}}">

<link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>

</head>

<body>

<div class="wrapper">

<form action="{{ route('contacts.submit') }}" method="POST">

@csrf

@method('POST')

@if (session()->has('error'))

{{ session('error') }}

@endif

@if (session()->has('success'))

{{ session('success') }}

@endif

<h1>Have any questions ?</h1>

<div class="input-box">

<input name="email" type="text" placeholder="Email" required>

<i class='bx bxs-user'></i>

@error('email')

<strong>{{ $message }}</strong>

@enderror

</div>

<div class="input-box">

<textarea name="question\_text" placeholder="Question" required></textarea>

@error('question\_text')

<strong>{{ $message }}</strong>

@enderror

</div>

<div style="clear:both"></div>

<button type="submit" class="btn">Send</button>

</form>

<div class="main-link">

<p>Return to <a href="{{route("main")}}">main page</a></p>

</div>

</div>

</body>

</html>

<!doctype html>

<html lang="{{ str\_replace('\_', '-', app()->getLocale()) }}">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<!-- CSRF Token -->

<meta name="csrf-token" content="{{ csrf\_token() }}">

<title>{{ config('app.name', 'Laravel') }}</title>

<!-- Fonts -->

<link rel="dns-prefetch" href="//fonts.bunny.net">

<link href="https://fonts.bunny.net/css?family=Nunito" rel="stylesheet">

<!-- Scripts -->

@vite(['resources/sass/app.scss', 'resources/js/app.js'])

</head>

<body>

<div id="app">

<nav class="navbar navbar-expand-md navbar-light bg-white shadow-sm">

<div class="container">

<a class="navbar-brand" href="{{ url('/') }}">

{{ config('app.name', 'Laravel') }}

</a>

<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="{{ \_\_('Toggle navigation') }}">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">

<!-- Left Side Of Navbar -->

<ul class="navbar-nav me-auto">

</ul>

<!-- Right Side Of Navbar -->

<ul class="navbar-nav ms-auto">

<!-- Authentication Links -->

@guest

@if (Route::has('login'))

<li class="nav-item">

<a class="nav-link" href="{{ route('login') }}">{{ \_\_('Login') }}</a>

</li>

@endif

@if (Route::has('register'))

<li class="nav-item">

<a class="nav-link" href="{{ route('register') }}">{{ \_\_('Register') }}</a>

</li>

@endif

@else

<li class="nav-item dropdown">

<a id="navbarDropdown" class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-haspopup="true" aria-expanded="false" v-pre>

{{ Auth::user()->name }}

</a>

<div class="dropdown-menu dropdown-menu-end" aria-labelledby="navbarDropdown">

<a class="dropdown-item" href="{{ route('logout') }}"

onclick="event.preventDefault();

document.getElementById('logout-form').submit();">

{{ \_\_('Logout') }}

</a>

<form id="logout-form" action="{{ route('logout') }}" method="POST" class="d-none">

@csrf

</form>

</div>

</li>

@endguest

</ul>

</div>

</div>

</nav>

<main class="py-4">

@yield('content')

</main>

</div>

</body>

</html>

<!DOCTYPE html>

<html>

<head>

<title>answer</title>

</head>

<body>

<h1>Hello, {{ $question->email }}</h1>

<p>Your question:</p>

<p>{{ $question->question\_text }}</p>

<p>On your question I can get for you next answer:</p>

<p>{{ $response }}</p>

<p>With best wishes,<br>

{{ config('app.name') }}</p>

</body>

</html>

{{ sprintf(\_\_('Hello, %s.'), $invoice->client\_name) }}

<br>

<br>

{{ sprintf(\_\_('You have an unpaid invoice %s.'), $invoice->reference) }}

<br>

<br>

{{ sprintf(\_\_('Payment sum %s EUR.'), $invoice->getTotalPriceWithTax()) }}

<br>

<br>

<a href="{{ route('payment.show-form', $invoice) }}">{{ \_\_('Payment link') }}</a>

<br>

<br>

{{ \_\_('Please, pay as soon as possible!') }}

@extends("layout")

@section('title') {{ \_\_('Invoice Payment') }} @endsection

@section("content")

@if (session()->has('success'))

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="alert alert-success">

{{ session()->get('success') }}

</div>

</div>

</div>

</div>

@endif

<div class="container" style="text-align: center;">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card" style="margin: 50px">

<div class="card-header text-uppercase">

<strong>{{ sprintf(\_\_('Payment For Invoice: %s'), $invoice->reference) }}</strong>

</div>

<div class="card-body">

<div class="row mb-3 p-4">

<h2 class="text-center">{{ \_\_('Invoice has been paid!') }}</h2>

</div>

</div>

</div>

</div>

</div>

</div>

@endsection

@extends("layout")

@section('title') {{ \_\_('Invoice Payment') }} @endsection

@section("content")

<div class="container">

<div class="project-card" style="margin: 50px">

<div class="card-content">

<form action="{{ route('payment.execute', $invoice) }}" method="POST">

@csrf

@if (session()->has('error'))

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="alert alert-danger">

{{ session()->get('error') }}

</div>

</div>

</div>

</div>

@endif

@if (session()->has('success'))

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="alert alert-success">

{{ session()->get('success') }}

</div>

</div>

</div>

</div>

@endif

<div class="container">

<div class="row justify-content-center" style="text-align: center">

<div class="col-md-8">

<div class="card">

<div class="card-header text-uppercase">

<strong>{{ sprintf(\_\_('Payment For Invoice: %s'), $invoice->reference) }}</strong>

</div>

<div class="card-body">

<div>{{ \_\_('Reference: ') }}

{{ $invoice->reference }}</div>

<div>{{ \_\_('Payment Sum: ') }}

{{ $invoice->getTotalPriceWithTax() }} EUR </div>

</div>

<br>

<div class="card-footer text-end">

<button type="submit" class="btn btn-primary text-uppercase" style="background-color: #007bff; border-color: #007bff; color: white; margin: 0 auto;">

{{ \_\_('Pay') }}

</button>

</div>

</div>

</div>

</div>

</div>

</form>

</div>

</div>

</div>

@endsection

@if ($paginator->hasPages())

<nav>

<ul class="pagination">

{{-- Previous Page Link --}}

@if ($paginator->onFirstPage())

<li class="page-item disabled" aria-disabled="true" aria-label="@lang('pagination.previous')">

<span class="page-link" aria-hidden="true">&lsaquo;</span>

</li>

@else

<li class="page-item">

<a class="page-link" href="{{ $paginator->previousPageUrl() }}" rel="prev" aria-label="@lang('pagination.previous')">&lsaquo;</a>

</li>

@endif

{{-- Pagination Elements --}}

@foreach ($elements as $element)

{{-- "Three Dots" Separator --}}

@if (is\_string($element))

<li class="page-item disabled" aria-disabled="true"><span class="page-link">{{ $element }}</span></li>

@endif

{{-- Array Of Links --}}

@if (is\_array($element))

@foreach ($element as $page => $url)

@if ($page == $paginator->currentPage())

<li class="page-item active" aria-current="page"><span class="page-link">{{ $page }}</span></li>

@else

<li class="page-item"><a class="page-link" href="{{ $url }}">{{ $page }}</a></li>

@endif

@endforeach

@endif

@endforeach

{{-- Next Page Link --}}

@if ($paginator->hasMorePages())

<li class="page-item">

<a class="page-link" href="{{ $paginator->nextPageUrl() }}" rel="next" aria-label="@lang('pagination.next')">&rsaquo;</a>

</li>

@else

<li class="page-item disabled" aria-disabled="true" aria-label="@lang('pagination.next')">

<span class="page-link" aria-hidden="true">&rsaquo;</span>

</li>

@endif

</ul>

</nav>

@endif

@if ($paginator->hasPages())

<nav class="d-flex justify-items-center justify-content-between">

<div class="d-flex justify-content-between flex-fill d-sm-none">

<ul class="pagination">

{{-- Previous Page Link --}}

@if ($paginator->onFirstPage())

<li class="page-item disabled" aria-disabled="true">

<span class="page-link">@lang('pagination.previous')</span>

</li>

@else

<li class="page-item">

<a class="page-link" href="{{ $paginator->previousPageUrl() }}" rel="prev">@lang('pagination.previous')</a>

</li>

@endif

{{-- Next Page Link --}}

@if ($paginator->hasMorePages())

<li class="page-item">

<a class="page-link" href="{{ $paginator->nextPageUrl() }}" rel="next">@lang('pagination.next')</a>

</li>

@else

<li class="page-item disabled" aria-disabled="true">

<span class="page-link">@lang('pagination.next')</span>

</li>

@endif

</ul>

</div>

<div class="d-none flex-sm-fill d-sm-flex align-items-sm-center justify-content-sm-between">

<div>

<p class="small text-muted">

{!! \_\_('Showing') !!}

<span class="fw-semibold">{{ $paginator->firstItem() }}</span>

{!! \_\_('to') !!}

<span class="fw-semibold">{{ $paginator->lastItem() }}</span>

{!! \_\_('of') !!}

<span class="fw-semibold">{{ $paginator->total() }}</span>

{!! \_\_('results') !!}

</p>

</div>

<div>

<ul class="pagination">

{{-- Previous Page Link --}}

@if ($paginator->onFirstPage())

<li class="page-item disabled" aria-disabled="true" aria-label="@lang('pagination.previous')">

<span class="page-link" aria-hidden="true">&lsaquo;</span>

</li>

@else

<li class="page-item">

<a class="page-link" href="{{ $paginator->previousPageUrl() }}" rel="prev" aria-label="@lang('pagination.previous')">&lsaquo;</a>

</li>

@endif

{{-- Pagination Elements --}}

@foreach ($elements as $element)

{{-- "Three Dots" Separator --}}

@if (is\_string($element))

<li class="page-item disabled" aria-disabled="true"><span class="page-link">{{ $element }}</span></li>

@endif

{{-- Array Of Links --}}

@if (is\_array($element))

@foreach ($element as $page => $url)

@if ($page == $paginator->currentPage())

<li class="page-item active" aria-current="page"><span class="page-link">{{ $page }}</span></li>

@else

<li class="page-item"><a class="page-link" href="{{ $url }}">{{ $page }}</a></li>

@endif

@endforeach

@endif

@endforeach

{{-- Next Page Link --}}

@if ($paginator->hasMorePages())

<li class="page-item">

<a class="page-link" href="{{ $paginator->nextPageUrl() }}" rel="next" aria-label="@lang('pagination.next')">&rsaquo;</a>

</li>

@else

<li class="page-item disabled" aria-disabled="true" aria-label="@lang('pagination.next')">

<span class="page-link" aria-hidden="true">&rsaquo;</span>

</li>

@endif

</ul>

</div>

</div>

</nav>

@endif

@if ($paginator->hasPages())

<nav>

<ul class="pagination">

{{-- Previous Page Link --}}

@if ($paginator->onFirstPage())

<li class="disabled" aria-disabled="true" aria-label="@lang('pagination.previous')">

<span aria-hidden="true">&lsaquo;</span>

</li>

@else

<li>

<a href="{{ $paginator->previousPageUrl() }}" rel="prev" aria-label="@lang('pagination.previous')">&lsaquo;</a>

</li>

@endif

{{-- Pagination Elements --}}

@foreach ($elements as $element)

{{-- "Three Dots" Separator --}}

@if (is\_string($element))

<li class="disabled" aria-disabled="true"><span>{{ $element }}</span></li>

@endif

{{-- Array Of Links --}}

@if (is\_array($element))

@foreach ($element as $page => $url)

@if ($page == $paginator->currentPage())

<li class="active" aria-current="page"><span>{{ $page }}</span></li>

@else

<li><a href="{{ $url }}">{{ $page }}</a></li>

@endif

@endforeach

@endif

@endforeach

{{-- Next Page Link --}}

@if ($paginator->hasMorePages())

<li>

<a href="{{ $paginator->nextPageUrl() }}" rel="next" aria-label="@lang('pagination.next')">&rsaquo;</a>

</li>

@else

<li class="disabled" aria-disabled="true" aria-label="@lang('pagination.next')">

<span aria-hidden="true">&rsaquo;</span>

</li>

@endif

</ul>

</nav>

@endif

@if ($paginator->hasPages())

<div class="ui pagination menu" role="navigation">

{{-- Previous Page Link --}}

@if ($paginator->onFirstPage())

<a class="icon item disabled" aria-disabled="true" aria-label="@lang('pagination.previous')"> <i class="left chevron icon"></i> </a>

@else

<a class="icon item" href="{{ $paginator->previousPageUrl() }}" rel="prev" aria-label="@lang('pagination.previous')"> <i class="left chevron icon"></i> </a>

@endif

{{-- Pagination Elements --}}

@foreach ($elements as $element)

{{-- "Three Dots" Separator --}}

@if (is\_string($element))

<a class="icon item disabled" aria-disabled="true">{{ $element }}</a>

@endif

{{-- Array Of Links --}}

@if (is\_array($element))

@foreach ($element as $page => $url)

@if ($page == $paginator->currentPage())

<a class="item active" href="{{ $url }}" aria-current="page">{{ $page }}</a>

@else

<a class="item" href="{{ $url }}">{{ $page }}</a>

@endif

@endforeach

@endif

@endforeach

{{-- Next Page Link --}}

@if ($paginator->hasMorePages())

<a class="icon item" href="{{ $paginator->nextPageUrl() }}" rel="next" aria-label="@lang('pagination.next')"> <i class="right chevron icon"></i> </a>

@else

<a class="icon item disabled" aria-disabled="true" aria-label="@lang('pagination.next')"> <i class="right chevron icon"></i> </a>

@endif

</div>

@endif

@if ($paginator->hasPages())

<nav>

<ul class="pagination">

{{-- Previous Page Link --}}

@if ($paginator->onFirstPage())

<li class="page-item disabled" aria-disabled="true">

<span class="page-link">@lang('pagination.previous')</span>

</li>

@else

<li class="page-item">

<a class="page-link" href="{{ $paginator->previousPageUrl() }}" rel="prev">@lang('pagination.previous')</a>

</li>

@endif

{{-- Next Page Link --}}

@if ($paginator->hasMorePages())

<li class="page-item">

<a class="page-link" href="{{ $paginator->nextPageUrl() }}" rel="next">@lang('pagination.next')</a>

</li>

@else

<li class="page-item disabled" aria-disabled="true">

<span class="page-link">@lang('pagination.next')</span>

</li>

@endif

</ul>

</nav>

@endif

@if ($paginator->hasPages())

<nav role="navigation" aria-label="Pagination Navigation">

<ul class="pagination">

{{-- Previous Page Link --}}

@if ($paginator->onFirstPage())

<li class="page-item disabled" aria-disabled="true">

<span class="page-link">{!! \_\_('pagination.previous') !!}</span>

</li>

@else

<li class="page-item">

<a class="page-link" href="{{ $paginator->previousPageUrl() }}" rel="prev">

{!! \_\_('pagination.previous') !!}

</a>

</li>

@endif

{{-- Next Page Link --}}

@if ($paginator->hasMorePages())

<li class="page-item">

<a class="page-link" href="{{ $paginator->nextPageUrl() }}" rel="next">{!! \_\_('pagination.next') !!}</a>

</li>

@else

<li class="page-item disabled" aria-disabled="true">

<span class="page-link">{!! \_\_('pagination.next') !!}</span>

</li>

@endif

</ul>

</nav>

@endif

@if ($paginator->hasPages())

<nav>

<ul class="pagination">

{{-- Previous Page Link --}}

@if ($paginator->onFirstPage())

<li class="disabled" aria-disabled="true"><span>@lang('pagination.previous')</span></li>

@else

<li><a href="{{ $paginator->previousPageUrl() }}" rel="prev">@lang('pagination.previous')</a></li>

@endif

{{-- Next Page Link --}}

@if ($paginator->hasMorePages())

<li><a href="{{ $paginator->nextPageUrl() }}" rel="next">@lang('pagination.next')</a></li>

@else

<li class="disabled" aria-disabled="true"><span>@lang('pagination.next')</span></li>

@endif

</ul>

</nav>

@endif

@if ($paginator->hasPages())

<nav role="navigation" aria-label="Pagination Navigation" class="flex justify-between">

{{-- Previous Page Link --}}

@if ($paginator->onFirstPage())

<span class="relative inline-flex items-center px-4 py-2 text-sm font-medium text-gray-500 bg-white border border-gray-300 cursor-default leading-5 rounded-md dark:text-gray-600 dark:bg-gray-800 dark:border-gray-600">

{!! \_\_('pagination.previous') !!}

</span>

@else

<a href="{{ $paginator->previousPageUrl() }}" rel="prev" class="relative inline-flex items-center px-4 py-2 text-sm font-medium text-gray-700 bg-white border border-gray-300 leading-5 rounded-md hover:text-gray-500 focus:outline-none focus:ring ring-gray-300 focus:border-blue-300 active:bg-gray-100 active:text-gray-700 transition ease-in-out duration-150 dark:bg-gray-800 dark:border-gray-600 dark:text-gray-300 dark:focus:border-blue-700 dark:active:bg-gray-700 dark:active:text-gray-300">

{!! \_\_('pagination.previous') !!}

</a>

@endif

{{-- Next Page Link --}}

@if ($paginator->hasMorePages())

<a href="{{ $paginator->nextPageUrl() }}" rel="next" class="relative inline-flex items-center px-4 py-2 text-sm font-medium text-gray-700 bg-white border border-gray-300 leading-5 rounded-md hover:text-gray-500 focus:outline-none focus:ring ring-gray-300 focus:border-blue-300 active:bg-gray-100 active:text-gray-700 transition ease-in-out duration-150 dark:bg-gray-800 dark:border-gray-600 dark:text-gray-300 dark:focus:border-blue-700 dark:active:bg-gray-700 dark:active:text-gray-300">

{!! \_\_('pagination.next') !!}

</a>

@else

<span class="relative inline-flex items-center px-4 py-2 text-sm font-medium text-gray-500 bg-white border border-gray-300 cursor-default leading-5 rounded-md dark:text-gray-600 dark:bg-gray-800 dark:border-gray-600">

{!! \_\_('pagination.next') !!}

</span>

@endif

</nav>

@endif

@if ($paginator->hasPages())

<nav role="navigation" aria-label="{{ \_\_('Pagination Navigation') }}" class="flex items-center justify-between">

<div class="flex justify-between flex-1 sm:hidden">

@if ($paginator->onFirstPage())

<span class="relative inline-flex items-center px-4 py-2 text-sm font-medium text-gray-500 bg-white border border-gray-300 cursor-default leading-5 rounded-md dark:text-gray-600 dark:bg-gray-800 dark:border-gray-600">

{!! \_\_('pagination.previous') !!}

</span>

@else

<a href="{{ $paginator->previousPageUrl() }}" class="relative inline-flex items-center px-4 py-2 text-sm font-medium text-gray-700 bg-white border border-gray-300 leading-5 rounded-md hover:text-gray-500 focus:outline-none focus:ring ring-gray-300 focus:border-blue-300 active:bg-gray-100 active:text-gray-700 transition ease-in-out duration-150 dark:bg-gray-800 dark:border-gray-600 dark:text-gray-300 dark:focus:border-blue-700 dark:active:bg-gray-700 dark:active:text-gray-300">

{!! \_\_('pagination.previous') !!}

</a>

@endif

@if ($paginator->hasMorePages())

<a href="{{ $paginator->nextPageUrl() }}" class="relative inline-flex items-center px-4 py-2 ml-3 text-sm font-medium text-gray-700 bg-white border border-gray-300 leading-5 rounded-md hover:text-gray-500 focus:outline-none focus:ring ring-gray-300 focus:border-blue-300 active:bg-gray-100 active:text-gray-700 transition ease-in-out duration-150 dark:bg-gray-800 dark:border-gray-600 dark:text-gray-300 dark:focus:border-blue-700 dark:active:bg-gray-700 dark:active:text-gray-300">

{!! \_\_('pagination.next') !!}

</a>

@else

<span class="relative inline-flex items-center px-4 py-2 ml-3 text-sm font-medium text-gray-500 bg-white border border-gray-300 cursor-default leading-5 rounded-md dark:text-gray-600 dark:bg-gray-800 dark:border-gray-600">

{!! \_\_('pagination.next') !!}

</span>

@endif

</div>

<div class="hidden sm:flex-1 sm:flex sm:items-center sm:justify-between">

<div>

<p class="text-sm text-gray-700 leading-5 dark:text-gray-400">

{!! \_\_('Showing') !!}

@if ($paginator->firstItem())

<span class="font-medium">{{ $paginator->firstItem() }}</span>

{!! \_\_('to') !!}

<span class="font-medium">{{ $paginator->lastItem() }}</span>

@else

{{ $paginator->count() }}

@endif

{!! \_\_('of') !!}

<span class="font-medium">{{ $paginator->total() }}</span>

{!! \_\_('results') !!}

</p>

</div>

<div>

<span class="relative z-0 inline-flex rtl:flex-row-reverse shadow-sm rounded-md">

{{-- Previous Page Link --}}

@if ($paginator->onFirstPage())

<span aria-disabled="true" aria-label="{{ \_\_('pagination.previous') }}">

<span class="relative inline-flex items-center px-2 py-2 text-sm font-medium text-gray-500 bg-white border border-gray-300 cursor-default rounded-l-md leading-5 dark:bg-gray-800 dark:border-gray-600" aria-hidden="true">

<svg class="w-5 h-5" fill="currentColor" viewBox="0 0 20 20">

<path fill-rule="evenodd" d="M12.707 5.293a1 1 0 010 1.414L9.414 10l3.293 3.293a1 1 0 01-1.414 1.414l-4-4a1 1 0 010-1.414l4-4a1 1 0 011.414 0z" clip-rule="evenodd" />

</svg>

</span>

</span>

@else

<a href="{{ $paginator->previousPageUrl() }}" rel="prev" class="relative inline-flex items-center px-2 py-2 text-sm font-medium text-gray-500 bg-white border border-gray-300 rounded-l-md leading-5 hover:text-gray-400 focus:z-10 focus:outline-none focus:ring ring-gray-300 focus:border-blue-300 active:bg-gray-100 active:text-gray-500 transition ease-in-out duration-150 dark:bg-gray-800 dark:border-gray-600 dark:active:bg-gray-700 dark:focus:border-blue-800" aria-label="{{ \_\_('pagination.previous') }}">

<svg class="w-5 h-5" fill="currentColor" viewBox="0 0 20 20">

<path fill-rule="evenodd" d="M12.707 5.293a1 1 0 010 1.414L9.414 10l3.293 3.293a1 1 0 01-1.414 1.414l-4-4a1 1 0 010-1.414l4-4a1 1 0 011.414 0z" clip-rule="evenodd" />

</svg>

</a>

@endif

{{-- Pagination Elements --}}

@foreach ($elements as $element)

{{-- "Three Dots" Separator --}}

@if (is\_string($element))

<span aria-disabled="true">

<span class="relative inline-flex items-center px-4 py-2 -ml-px text-sm font-medium text-gray-700 bg-white border border-gray-300 cursor-default leading-5 dark:bg-gray-800 dark:border-gray-600">{{ $element }}</span>

</span>

@endif

{{-- Array Of Links --}}

@if (is\_array($element))

@foreach ($element as $page => $url)

@if ($page == $paginator->currentPage())

<span aria-current="page">

<span class="relative inline-flex items-center px-4 py-2 -ml-px text-sm font-medium text-gray-500 bg-white border border-gray-300 cursor-default leading-5 dark:bg-gray-800 dark:border-gray-600">{{ $page }}</span>

</span>

@else

<a href="{{ $url }}" class="relative inline-flex items-center px-4 py-2 -ml-px text-sm font-medium text-gray-700 bg-white border border-gray-300 leading-5 hover:text-gray-500 focus:z-10 focus:outline-none focus:ring ring-gray-300 focus:border-blue-300 active:bg-gray-100 active:text-gray-700 transition ease-in-out duration-150 dark:bg-gray-800 dark:border-gray-600 dark:text-gray-400 dark:hover:text-gray-300 dark:active:bg-gray-700 dark:focus:border-blue-800" aria-label="{{ \_\_('Go to page :page', ['page' => $page]) }}">

{{ $page }}

</a>

@endif

@endforeach

@endif

@endforeach

{{-- Next Page Link --}}

@if ($paginator->hasMorePages())

<a href="{{ $paginator->nextPageUrl() }}" rel="next" class="relative inline-flex items-center px-2 py-2 -ml-px text-sm font-medium text-gray-500 bg-white border border-gray-300 rounded-r-md leading-5 hover:text-gray-400 focus:z-10 focus:outline-none focus:ring ring-gray-300 focus:border-blue-300 active:bg-gray-100 active:text-gray-500 transition ease-in-out duration-150 dark:bg-gray-800 dark:border-gray-600 dark:active:bg-gray-700 dark:focus:border-blue-800" aria-label="{{ \_\_('pagination.next') }}">

<svg class="w-5 h-5" fill="currentColor" viewBox="0 0 20 20">

<path fill-rule="evenodd" d="M7.293 14.707a1 1 0 010-1.414L10.586 10 7.293 6.707a1 1 0 011.414-1.414l4 4a1 1 0 010 1.414l-4 4a1 1 0 01-1.414 0z" clip-rule="evenodd" />

</svg>

</a>

@else

<span aria-disabled="true" aria-label="{{ \_\_('pagination.next') }}">

<span class="relative inline-flex items-center px-2 py-2 -ml-px text-sm font-medium text-gray-500 bg-white border border-gray-300 cursor-default rounded-r-md leading-5 dark:bg-gray-800 dark:border-gray-600" aria-hidden="true">

<svg class="w-5 h-5" fill="currentColor" viewBox="0 0 20 20">

<path fill-rule="evenodd" d="M7.293 14.707a1 1 0 010-1.414L10.586 10 7.293 6.707a1 1 0 011.414-1.414l4 4a1 1 0 010 1.414l-4 4a1 1 0 01-1.414 0z" clip-rule="evenodd" />

</svg>

</span>

</span>

@endif

</span>

</div>

</div>

</nav>

@endif

@extends("layout")

@section('title')

About

@endsection

@section("content")

<!--

- #ABOUT

-->

<section class="about" aria-labelledby="about-label">

<div class="container">

<figure class="about-banner">

<img src="./assets/images/about-banner.png" width="800" height="580" loading="lazy" alt="about banner"

class="w-100">

</figure>

<div class="about-content">

<p class="section-subtitle" id="about-label">Why Choose Us?</p>

<h2 class="h2 section-title">

We bring solutions to make life easier for our clients.

</h2>

<ul>

<li class="about-item">

<div class="accordion-card expanded" data-accordion>

<h3 class="card-title">

<button class="accordion-btn" data-accordion-btn>

<ion-icon name="chevron-down-outline" aria-hidden="true" class="down"></ion-icon>

<spna class="span h5">Professional Design</spna>

</button>

</h3>

<p class="accordion-content">

Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum

massa justo

sit amet risus. Cras mattis consectetur purus sit amet fermentum. Praesent commodo

cursus magna,

vel.

</p>

</div>

</li>

<li class="about-item">

<div class="accordion-card" data-accordion>

<h3 class="card-title">

<button class="accordion-btn" data-accordion-btn>

<ion-icon name="chevron-down-outline" aria-hidden="true" class="down"></ion-icon>

<spna class="span h5">Top-Notch Support</spna>

</button>

</h3>

<p class="accordion-content">

Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum

massa justo

sit amet risus. Cras mattis consectetur purus sit amet fermentum. Praesent commodo

cursus magna,

vel.

</p>

</div>

</li>

<li class="about-item">

<div class="accordion-card" data-accordion>

<h3 class="card-title">

<button class="accordion-btn" data-accordion-btn>

<ion-icon name="chevron-down-outline" aria-hidden="true" class="down"></ion-icon>

<spna class="span h5">Exclusive Assets</spna>

</button>

</h3>

<p class="accordion-content">

Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum

massa justo

sit amet risus. Cras mattis consectetur purus sit amet fermentum. Praesent commodo

cursus magna,

vel.

</p>

</div>

</li>

</ul>

</div>

</div>

</section>

{{--

<div class="container text-center">

<div class="row justify-content-center">

<div class="col-md-10 col-lg-6">

<h3 class="mb-4" style="text-shadow: 0 0 3px rgba(255, 255, 255, 1), 0 0 1px rgba(255, 255, 255, 1), 0 0 1px rgba(255, 255, 255, 1);">― EuroCipari ―</h3>

<h1 class="display-4 mb-4"><strong style="font-weight: bold;">About Us</strong></h1>

<p class="mb-4">

EuroCipari is an online accounting automation company that helps businesses and individuals save time and money.

We offer a wide range of services including web accounting software, bookkeeping services, tax consulting and financial analysis.

<br>

<br>

Our mission is to help our clients succeed by providing them with the tools

and resources they need to effectively manage their finances.

<br>

<br>

We believe that everyone should have access to high-quality accounting services,

regardless of their budget or experience level.

That's why we offer many affordable and flexible solutions.

<br>

<br>

If you are looking for a way to simplify your accounting

and streamline your business operations, EuroCipari is the perfect solution for you.

<br>

<br>

Contact us today on social media to learn more about our services or visit our services section.

</p>

</div>

</div>

</div>

<div class="container-fluid p-0">

<section class="map-section text-center">

<div class="embed-responsive embed-responsive-16by9">

<iframe class="embed-responsive-item w-100" frameborder="0" style="border:0" src="https://www.google.com/maps/embed/v1/place?key=AIzaSyCy9r70T3NYf3PhvVflTo0\_zdif2\_IoIYs&amp;q=place\_id:ChIJn6wOs6lZwokRLKy1iqRcoKw" allowfullscreen=""></iframe>

</div>

</section>

</div>

--}}

@endsection

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>@yield('title')</title>

<!-- Scripts -->

@vite(['resources/sass/app.scss', 'resources/js/app.js'])

</head>

<body class="bg-secondary">

<div class="container py-3">

<header>

<div class="d-flex flex-column flex-md-row align-items-center pb-3 mb-4 border-bottom">

<a href="/" class="d-flex align-items-center text-light text-decoration-none">

<span class="fs-4">EuroCipari</span>

</a>

<nav class="d-inline-flex mt-2 mt-md-0 ms-md-auto">

<form action="{{ route('companies.logout') }}" method="POST" class="me-3 py-2 text-light text-decoration-none">

@csrf

<button type="submit" class="btn btn-primary">Logout</button>

</form>

</nav>

</div>

</header>

@if (session()->has('success'))

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="alert alert-success">

{{ session()->get('success') }}

</div>

</div>

</div>

</div>

@endif

@if (session()->has('error'))

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="alert alert-danger">

{{ session()->get('error') }}

</div>

</div>

</div>

</div>

@endif

<div class="container text-light">@yield("main\_content")</div>

</div>

<!--

<footer class="d-flex flex-wrap justify-content-between align-items-center py-3 border-top">

<div class="col-md-4 d-flex align-items-center">

<a href="/" class="mb-3 me-2 mb-md-0 lh-1 text-light">

<svg class="bi" width="30" height="24"><use xlink:href="#bootstrap"></use></svg>

</a>

<span class="mb-3 mb-md-0 text-light">© 2023 Company "EuroCipari", Inc - All Rights Reserved</span>

</div>

<div class="bg-transparent text-center text-white">

<div class="container p-4 pb-0">

<section class="mb-4">

<a class="btn text-white btn-floating m-1" href="https://www.facebook.com/V.a.l.i.k.63?locale=ru\_RU" role="button">

<img src="{{ asset('/images/facebook.png') }}" alt="" />

</a>

<a class="btn text-white btn-floating m-1" href="https://www.instagram.com/v.a.l.i.k\_63/" role="button">

<img src="{{ asset('/images/instagram.png') }}" alt="" />

</a>

<a class="btn text-white btn-floating m-1" href="https://www.linkedin.com/in/valentin-kasper-7a9939286/" role="button">

<img src="{{ asset('/images/linkedin.png') }}" alt="" />

</a>

</section>

</div>

</div>

</footer>

-->

</body>

</html>

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>@yield('title')</title>

<!-- Scripts -->

@vite(['resources/sass/app.scss', 'resources/js/app.js'])

</head>

<body class="bg-dark">

<div class="container py-3">

<header>

<div class="d-flex flex-column flex-md-row align-items-center pb-3 mb-4 border-bottom">

<a href="/" class="d-flex align-items-center text-light text-decoration-none">

<span class="fs-4">EuroCipari</span>

</a>

<nav class="d-inline-flex mt-2 mt-md-0 ms-md-auto"> <!-- Создает навигационное меню. -->

<a class="me-3 py-2 text-light text-decoration-none" href="{{route('admin.companies.index')}}">Companies</a> <!-- Создает ссылку "". -->

<a class="me-3 py-2 text-light text-decoration-none" href="{{route('admin.users.index')}}">Administrators</a> <!-- Создает ссылку "". -->

<a class="me-3 py-2 text-light text-decoration-none" href="{{route('admin.posts.index')}}">Posts</a> <!-- Создает ссылку "". -->

<a class="me-3 py-2 text-light text-decoration-none" href="{{route('admin.questions.index')}}">Questions</a> <!-- Создает ссылку "". -->

<a class="me-3 py-2 text-light text-decoration-none" href="{{route('admin.invoices.index')}}">Invoices</a> <!-- Создает ссылку "". -->

</nav>

<nav class="d-inline-flex mt-2 mt-md-0 ms-md-auto">

<form action="{{ route('logout') }}" method="POST" class="me-3 py-2 text-light text-decoration-none">

@csrf

<button type="submit" class="btn btn-primary">Logout</button>

</form>

</nav>

</div>

</header>

@if (session()->has('success'))

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="alert alert-success">

{{ session()->get('success') }}

</div>

</div>

</div>

</div>

@endif

@if (session()->has('error'))

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8">

<div class="alert alert-danger">

{{ session()->get('error') }}

</div>

</div>

</div>

</div>

@endif

<div class="container text-light">@yield("main\_content")</div>

</div>

<!--

<footer class="d-flex flex-wrap justify-content-between align-items-center py-3 border-top">

<div class="col-md-4 d-flex align-items-center">

<a href="/" class="mb-3 me-2 mb-md-0 lh-1 text-light">

<svg class="bi" width="30" height="24"><use xlink:href="#bootstrap"></use></svg>

</a>

<span class="mb-3 mb-md-0 text-light">© 2023 Company "EuroCipari", Inc - All Rights Reserved</span>

</div>

<div class="bg-transparent text-center text-white">

<div class="container p-4 pb-0">

<section class="mb-4">

<a class="btn text-white btn-floating m-1" href="https://www.facebook.com/V.a.l.i.k.63?locale=ru\_RU" role="button">

<img src="{{ asset('/images/facebook.png') }}" alt="" />

</a>

<a class="btn text-white btn-floating m-1" href="https://www.instagram.com/v.a.l.i.k\_63/" role="button">

<img src="{{ asset('/images/instagram.png') }}" alt="" />

</a>

<a class="btn text-white btn-floating m-1" href="https://www.linkedin.com/in/valentin-kasper-7a9939286/" role="button">

<img src="{{ asset('/images/linkedin.png') }}" alt="" />

</a>

</section>

</div>

</div>

</footer>

-->

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<!--

- primary meta tags

-->

<title>EuroCipari</title>

<meta name="title" content="EuroCipari">

<meta name="description" content="This is a business agency html template made by codewithsadee">

<!--

- favicon

-->

<link rel="shortcut icon" href="{{ asset('favicon.svg') }}" type="image/svg+xml">

<!--

- google font link

-->

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Manrope:wght@500;700&display=swap" rel="stylesheet">

<!--

- custom css link

-->

<link rel="stylesheet" href="{{ asset('assets/css/style.css') }}">

<!--

- preload images

-->

<link rel="preload" as="image" href="{{ asset('assets/images/hero-bg.jpg') }}">

<link rel="preload" as="image" href="{{ asset('assets/images/hero-slide-1.jpg') }}">

<link rel="preload" as="image" href="{{ asset('assets/images/hero-slide-2.jpg') }}">

</head>

<body>

<!--

- #HEADER

-->

<header class="header" data-header>

<div class="container">

<a href="{{ route( 'main' ) }}" class="logo">

<img src="{{ asset('assets/images/logo.png') }}" width="74" height="24" alt="EuroCipari" class="logo-light">

</a>

<nav class="navbar" data-navbar>

<div class="navbar-top">

<a href="#" class="logo">

<img src="{{ asset('assets/images/logo.png') }}" width="74" height="24" alt="Adex home">

</a>

<button class="nav-close-btn" aria-label="close menu" data-nav-toggler>

<ion-icon name="close-outline" aria-hidden="true"></ion-icon>

</button>

</div>

<ul class="navbar-list">

<li>

<a href="{{ route('contacts.index') }}" class="navbar-link">Questions</a>

</li>

</ul>

</nav>

<a href="{{ route('companies.login') }}" class="btn btn-primary">Companies panel</a>

<button class="nav-open-btn" aria-label="open menu" data-nav-toggler>

<ion-icon name="menu-outline" aria-hidden="true"></ion-icon>

</button>

<div class="overlay" data-nav-toggler data-overlay></div>

</div>

</header>

<main>

<article>

<!--

- #HERO

-->

<section class="section hero has-bg-image" aria-label="home"

style="background-image: url({{ asset('assets/images/hero-bg.jpg') }}">

<div class="container">

<div class="hero-content">

<h1 class="h1 hero-title">Tailored solutions with professionalism.</h1>

<p class="hero-text">

We are a creative company that values long-term relationships with clients.

</p>

</div>

<div class="hero-slider" data-slider>

<div class="slider-inner">

<ul class="slider-container" data-slider-container>

<li class="slider-item">

<figure class="img-holder" style="--width: 575; --height: 550;">

<img src="{{ asset('assets/images/hero-slide-1.jpg') }}" width="575" height="550" alt="" class="img-cover">

</figure>

</li>

<li class="slider-item">

<div class="hero-card">

<figure class="img-holder" style="--width: 575; --height: 550;">

<img src="{{ asset('assets/images/hero-slide-2.jpg') }}" width="575" height="550" alt="hero banner"

class="img-cover">

</figure>

</div>

</li>

</ul>

</div>

<button class="slider-btn prev" aria-label="slide to previous" data-slider-prev>

<ion-icon name="arrow-back"></ion-icon>

</button>

<button class="slider-btn next" aria-label="slide to next" data-slider-next>

<ion-icon name="arrow-forward"></ion-icon>

</button>

</div>

</div>

</section>

@yield('content')

<!--

- #CTA

-->

<section class="cta" aria-label="call to action">

<div class="container">

<h2 class="h2 section-title">

Using our services and grow your business.

</h2>

</div>

</section>

</article>

</main>

<!--

- #FOOTER

-->

<footer class="footer">

<div class="container grid-list">

<div class="footer-brand">

<p class="footer-text">

&copy; 2023 EuroCipari. <br> All rights reserved.

</p>

</div>

</div>

</footer>

<!--

- custom js link

-->

<script src="{{ asset('assets/js/script.js') }}"></script>

<!--

- ionicon

-->

<script type="module" src="https://unpkg.com/ionicons@5.5.2/dist/ionicons/ionicons.esm.js"></script>

<script nomodule src="https://unpkg.com/ionicons@5.5.2/dist/ionicons/ionicons.js"></script>

</body>

</html>

@extends('layout')

@section('content')

<!--

- #SERVICE

-->

<section class="section service" aria-labelledby="service-label">

<div class="container">

<p class="section-subtitle" id="service-label">WHAT ARE WE DOING ?</p>

<h2 class="h2 section-title">

Our service is specifically designed to meet your needs.

</h2>

<ul class="grid-list">

<li>

<div class="service-card">

<div class="card-icon">

<ion-icon name="call-outline" aria-hidden="true"></ion-icon>

</div>

<h3 class="h4 card-title">24/7 support</h3>

<p class="card-text">

We are always ready to help you. Our team of specialists works around the clock to provide you with the best service.

</p>

</div>

</li>

<li>

<div class="service-card">

<div class="card-icon">

<ion-icon name="shield-checkmark-outline" aria-hidden="true"></ion-icon>

</div>

<h3 class="h4 card-title">Secure Payments</h3>

<p class="card-text">

We guarantee the security of your payments. Our system provides reliable protection of your financial data.

</p>

</div>

</li>

<li>

<div class="service-card">

<div class="card-icon">

<ion-icon name="cloud-download-outline" aria-hidden="true"></ion-icon>

</div>

<h3 class="h4 card-title">Daily Updates</h3>

<p class="card-text">

We are constantly working to improve our service. You will always be aware of the latest news and updates.

</p>

</div>

</li>

<li>

<div class="service-card">

<div class="card-icon">

<ion-icon name="pie-chart-outline" aria-hidden="true"></ion-icon>

</div>

<h3 class="h4 card-title">Market Research</h3>

<p class="card-text">

We conduct thorough market research to offer you the most effective and relevant solutions.

</p>

</div>

</li>

</ul>

</div>

</section>

<!--

- #ABOUT

-->

<section class="about" aria-labelledby="about-label">

<div class="container">

<figure class="about-banner">

<img src="{{ asset('assets/images/about-banner.png') }}" width="800" height="580" loading="lazy" alt="about banner"

class="w-100">

</figure>

<div class="about-content">

<p class="section-subtitle" id="about-label">WHY CHOOSE US ?</p>

<h2 class="h2 section-title">

We offer solutions that make our customers' lives easier.

</h2>

<ul>

<li class="about-item">

<div class="accordion-card" data-accordion>

<h3 class="card-title">

<button class="accordion-btn" data-accordion-btn>

<ion-icon name="chevron-down-outline" aria-hidden="true" class="down"></ion-icon>

<spna class="span h5">Efficiency</spna>

</button>

</h3>

<p class="accordion-content">

Our payment system automates all your payments, saving you time and effort.

</p>

</div>

</li>

<li class="about-item">

<div class="accordion-card" data-accordion>

<h3 class="card-title">

<button class="accordion-btn" data-accordion-btn>

<ion-icon name="chevron-down-outline" aria-hidden="true" class="down"></ion-icon>

<spna class="span h5">Integration</spna>

</button>

</h3>

<p class="accordion-content">

Our payment system integrates easily with other systems to ensure your business runs smoothly and efficiently.

</p>

</div>

</li>

</ul>

</div>

</div>

</section>

<!--

- #FEATURE

-->

<section class="section feature" aria-labelledby="feature-label">

<div class="container">

<figure class="feature-banner">

<img src="{{ asset('assets/images/feature-banner.png') }}" width="800" height="531" loading="lazy" alt="feature banner"

class="w-100">

</figure>

<div class="feature-content">

<p class="section-subtitle" id="feautre-label">Our Solutions</p>

<h2 class="h2 section-title">

We make your business process flawless so that you can control everything perfectly.

</h2>

<p class="section-text">

Your business processes become more efficient with our system:

</p>

<ul class="feature-list">

<li>

<div class="feature-card">

<div class="card-icon">

<ion-icon name="checkmark" aria-hidden="true"></ion-icon>

</div>

<span class="span">

Provides seamless integration with your current operations.

</span>

</div>

</li>

<li>

<div class="feature-card">

<div class="card-icon">

<ion-icon name="checkmark" aria-hidden="true"></ion-icon>

</div>

<span class="span">

Allows you to easily track and manage your business processes.

</span>

</div>

</li>

<li>

<div class="feature-card">

<div class="card-icon">

<ion-icon name="checkmark" aria-hidden="true"></ion-icon>

</div>

<span class="span">

Offers an intuitive interface that makes it easy to use.

</span>

</div>

</li>

<li>

<div class="feature-card">

<div class="card-icon">

<ion-icon name="checkmark" aria-hidden="true"></ion-icon>

</div>

<span class="span">

Provides a reliable and secure environment for your data.

</span>

</div>

</li>

</ul>

</div>

</div>

</section>

<!--

- #STATS

-->

<section class="stats" aria-label="our stats">

<div class="container">

<ul class="stats-card has-bg-image" style="background-image: url({{ asset('assets/images/stats-bg.jpg') }}">

<li>

<p class="card-text">

<span class="h1">{{$paymentsQuantity}}</span>

<spna class="span">Processed Payments</spna>

</p>

</li>

<li>

<p class="card-text">

<span class="h1">{{$companiesQuantity}}</span>

<spna class="span">companies chose us</spna>

</p>

</li>

</ul>

</div>

</section>

<!--

- #PROJECT

-->

<section class="section project" aria-labelledby="project-label">

<div class="container">

<ul class="grid-list">

@foreach($posts as $post)

<li>

<div class="project-card">

<figure class="card-banner img-holder" style="--width: 560; --height: 350;">

<img src="{{ $post->getImageFullUrl() }}" width="560" height="350" loading="lazy" alt="Ultricies fusce porta elit" class="img-cover">

</figure>

<div class="card-content">

<h3 class="h3">

<a href="{{ route('blog.show', $post) }}" class="card-title">{{ $post->title }}</a>

</h3>

<p class="card-text">{{ Str::limit(strip\_tags($post->text), 500) }}</p>

<ul class="card-meta-list">

<li class="card-meta-item">

<ion-icon name="calendar-outline" aria-hidden="true" role="img" class="md hydrated"></ion-icon>

<time class="meta-text" datetime="{{ $post->created\_at }}">{{ $post->created\_at }}</time>

</li>

</ul>

<ul class="card-meta-list">

</ul>

</div>

</div>

</li>

@endforeach

</ul>

</div>

</section>

@endsection

@extends("layout")

@section('title')

Services

@endsection

@section("main\_content")

<div class="pricing-header px-3 py-3 pt-md-5 pb-md-4 mx-auto text-center">

<h1 class="display-4">Pricing</h1>

<p class="lead">Quickly build an effective pricing table for your potential customers with this Bootstrap

example. It's built with default Bootstrap components and utilities with little customization.</p>

</div>

<div class="container">

<div class="row">

<div class="col-md-4">

<div class="card mb-4 box-shadow d-flex flex-column align-items-center">

<div class="card-header">

<h4 class="my-0 font-weight-normal">Free</h4>

</div>

<div class="card-body">

<h1 class="card-title pricing-card-title">$0 <small class="text-muted">/ mo</small></h1>

<ul class="list-unstyled mt-3 mb-4 text-center">

<li>10 users included</li>

<li>2 GB of storage</li>

<li>Email support</li>

<li>Help center access</li>

</ul>

<button type="button" class="btn btn-lg btn-block btn-outline-primary">Sign up for free</button>

</div>

</div>

</div>

<div class="col-md-4">

<div class="card mb-4 box-shadow d-flex flex-column align-items-center">

<div class="card-header">

<h4 class="my-0 font-weight-normal">Pro</h4>

</div>

<div class="card-body">

<h1 class="card-title pricing-card-title">$15 <small class="text-muted">/ mo</small></h1>

<ul class="list-unstyled mt-3 mb-4 text-center">

<li>20 users included</li>

<li>10 GB of storage</li>

<li>Priority email support</li>

<li>Help center access</li>

</ul>

<button type="button" class="btn btn-lg btn-block btn-primary">Get started</button>

</div>

</div>

</div>

<div class="col-md-4">

<div class="card mb-4 box-shadow d-flex flex-column align-items-center">

<div class="card-header">

<h4 class="my-0 font-weight-normal">Enterprise</h4>

</div>

<div class="card-body">

<h1 class="card-title pricing-card-title">$29 <small class="text-muted">/ mo</small></h1>

<ul class="list-unstyled mt-3 mb-4 text-center">

<li>30 users included</li>

<li>15 GB of storage</li>

<li>Phone and email support</li>

<li>Help center access</li>

</ul>

<button type="button" class="btn btn-lg btn-block btn-primary">Contact us</button>

</div>

</div>

</div>

</div>

</div>

@endsection

**app/routes:**

<?php

use App\Http\Controllers\API\InvoiceController;

use App\Http\Middleware\APIAuthorizationMiddleware;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Route;

/\*

|--------------------------------------------------------------------------

| API Routes

|--------------------------------------------------------------------------

|

| Here is where you can register API routes for your application. These

| routes are loaded by the RouteServiceProvider and all of them will

| be assigned to the "api" middleware group. Make something great!

|

\*/

Route::group(['middleware' => 'api.auth'], function () {

Route::apiResource('invoices', InvoiceController::class);

});

<?php

use Illuminate\Support\Facades\Broadcast;

/\*

|--------------------------------------------------------------------------

| Broadcast Channels

|--------------------------------------------------------------------------

|

| Here you may register all of the event broadcasting channels that your

| application supports. The given channel authorization callbacks are

| used to check if an authenticated user can listen to the channel.

|

\*/

Broadcast::channel('App.Models.User.{id}', function ($user, $id) {

return (int) $user->id === (int) $id;

});

<?php

use Illuminate\Foundation\Inspiring;

use Illuminate\Support\Facades\Artisan;

/\*

|--------------------------------------------------------------------------

| Console Routes

|--------------------------------------------------------------------------

|

| This file is where you may define all of your Closure based console

| commands. Each Closure is bound to a command instance allowing a

| simple approach to interacting with each command's IO methods.

|

\*/

Artisan::command('inspire', function () {

$this->comment(Inspiring::quote());

})->purpose('Display an inspiring quote');

<?php

use App\Http\Controllers\Admin\AdminController;

use App\Http\Controllers\Admin\CompanyController;

use App\Http\Controllers\Admin\DashboardController;

use App\Http\Controllers\Admin\InvoiceController;

use App\Http\Controllers\Admin\PostController;

use App\Http\Controllers\Admin\QuestionController;

use App\Http\Controllers\BlogController;

use App\Http\Controllers\CompanyAuth\ForgotPasswordController;

use App\Http\Controllers\CompanyAuth\LoginController;

use App\Http\Controllers\CompanyAuth\ResetPasswordController;

use App\Http\Controllers\ContactsController;

use App\Http\Controllers\MainController;

use App\Http\Controllers\PaymentController;

use Illuminate\Support\Facades\Auth;

use Illuminate\Support\Facades\Route;

/\*

|--------------------------------------------------------------------------

| Web Routes

|--------------------------------------------------------------------------

|

| Here is where you can register web routes for your application. These

| routes are loaded by the RouteServiceProvider and all of them will

| be assigned to the "web" middleware group. Make something great!

|

\*/

Route::get('/',[MainController::class, 'main'])->name('main');

Route::get('/about',[MainController::class, 'about'])->name('about');

Route::get('/services',[MainController::class, 'services'])->name('services');

Route::get('/payments/{invoice}',[PaymentController::class, 'showPaymentForm'])->name('payment.show-form');

Route::post('/payments/{invoice}/execute',[PaymentController::class, 'executePayment'])->name('payment.execute');

Route::any('/payments/{invoice}/finish',[PaymentController::class, 'finish'])->name('payment.finish');

Route::any('/payments/{invoice}/cancel',[PaymentController::class, 'cancel'])->name('payment.cancel');

Route::group(['prefix' => '/admin', 'middleware' => 'auth', 'as' => 'admin.'], static function() {

Route::get('/', [DashboardController::class, 'index']);

Route::resource("companies", CompanyController::class)->except("show");

Route::resource("users", AdminController::class)->except("show");

Route::resource("posts", PostController::class)->except("show");

Route::resource("invoices", InvoiceController::class);

Route::resource('questions', QuestionController::class)->except('show', 'create', 'store');

Route::post('/questions/{question}/response',[QuestionController::class, 'response'])->name('questions.response');

Route::post('/invoices/{invoice}/send-email',[InvoiceController::class, 'sendEmail'])->name('invoices.send-email');

});

Route::group(['prefix' => '/companies-admin', 'middleware' => 'company.auth', 'as' => 'companies-admin.'], static function() {

Route::get('/', [\App\Http\Controllers\Company\DashboardController::class, 'index'])->name('index');

Route::get('/invoices/{invoice}', [\App\Http\Controllers\Company\DashboardController::class, 'show'])->name('show');

Route::get('/invoices/{invoice}/pdf', [\App\Http\Controllers\Company\DashboardController::class, 'showPdf'])->name('show-pdf');

});

Auth::routes();

// companies login and logout

Route::post('companies-auth/login', [LoginController::class, 'login']);

Route::get('companies-auth/login', [LoginController::class, 'showLoginForm'])->name('companies.login');

Route::post('companies-auth/logout', [LoginController::class, 'logout'])->name('companies.logout');

// companies password reset logic

Route::get('companies-auth/password/reset', [ForgotPasswordController::class, 'showLinkRequestForm'])

->name('companies.show\_password\_link\_request\_form');

Route::post('companies-auth/password/email', [ForgotPasswordController::class, 'sendResetLinkEmail'])

->name('companies.send\_password\_reset\_link\_email');

Route::get('companies-auth/password/reset/{token}', [ResetPasswordController::class, 'showResetForm'])

->name('companies.show\_password\_reset\_form');

Route::post('companies-auth/password/reset', [ResetPasswordController::class, 'reset'])

->name('companies.reset\_password');

Route::get('/home', [App\Http\Controllers\HomeController::class, 'index'])->name('home');

Route::controller(BlogController::class)->group(function () {

Route::get('/posts/{post}', 'show')->name('blog.show');

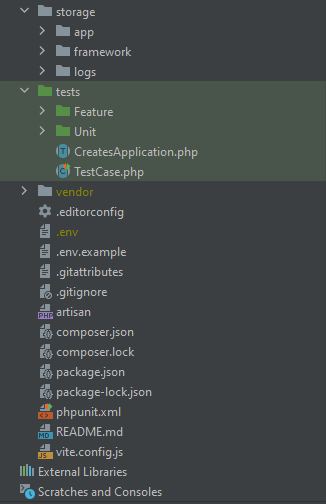
});

Route::controller(ContactsController::class)->group(function () {

Route::get('/contacts', 'index')->name('contacts.index');

Route::post('/contacts', 'submit')->name('contacts.submit');

});

Pārejie faili – tas ir framework faili

2.att. framework faili