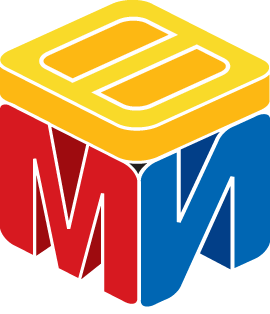
  
 **Пловдивски университет „Паисий Хилендарски”**  
 Факултет по Математика и Информатика   
 Катедра „Компютърни технологии“

  
Дипломна работа

Тема  
Система за управление на преводи

|  |  |
| --- | --- |
| **Дипломант:**  *Алекс Красимиров Московски*  *фак. № 1701757004*  *специалност:“Софтуерни технологии със специализация графични среди и потребителски интерфейс”* | **Научен ръководител:** *ас. Александър Петров* |

**Пловдив, 2018 г.**

# Съдържаниe

[Съдържаниe 2](#_Toc525554104)

[Списък с фигури 5](#_Toc525554105)

[Увод 8](#_Toc525554106)

[Цели на дипломната работа 9](#_Toc525554107)

[Структура на дипломната работа 10](#_Toc525554108)

[Глава 1. TMS платформи, използвани в уеб пространството 11](#_Toc525554109)

[1.1 Lingotek 12](#_Toc525554110)

[1.2 Wordbee 13](#_Toc525554111)

[1.3 ÜbrTà 14](#_Toc525554112)

[Глава 2. Технологии и средства за разработка 15](#_Toc525554113)

[2.1 Технологии за разработка 15](#_Toc525554116)

[2.1.1 Ubuntu OS 15](#_Toc525554117)

[2.1.2 Nginx 16](#_Toc525554118)

[2.1.3 PHP 16](#_Toc525554119)

[2.1.4 MySQL 16](#_Toc525554120)

[2.1.5 HTML 17](#_Toc525554121)

[2.1.6 CSS 18](#_Toc525554122)

[2.1.7 JavaScript и jQuery 19](#_Toc525554123)

[2.1.8 React (JavaScript library) 20](#_Toc525554124)

[2.1.9 Docker 20](#_Toc525554125)

[2.1.10 Phalcon – PHP framework 21](#_Toc525554126)

[2.1.11 Adobe Photoshop CS5 23](#_Toc525554127)

[2.1.12 Sublime 24](#_Toc525554128)

[Глава 3. Описание на системата 26](#_Toc525554129)

[3.1 Изисквания 26](#_Toc525554131)

[3.1.1 Лесна употреба 26](#_Toc525554132)

[3.1.2 Разбираемост 26](#_Toc525554133)

[3.1.3 Атрактивност 26](#_Toc525554134)

[3.1.4 Гъвкавост и възможност за разширение 26](#_Toc525554135)

[3.1.5 Достъпност и съвместимост с различни устройства 27](#_Toc525554136)

[3.2 Същност на приложението 28](#_Toc525554137)

[3.2.1 Логин 28](#_Toc525554138)

[3.2.2 Начална страница 29](#_Toc525554139)

[3.2.3 Приложения 30](#_Toc525554140)

[3.2.4 Компоненти 32](#_Toc525554141)

[3.2.5 Подпис-идентификация 34](#_Toc525554142)

[3.2.6 Преводи 35](#_Toc525554143)

[3.2.7 Генериране на файлове 36](#_Toc525554144)

[3.2.8 Езици 39](#_Toc525554146)

[3.2.9 Контрол на достъпа 40](#_Toc525554147)

[3.2.10 Управление на роли 41](#_Toc525554148)

[3.2.11 Профил 42](#_Toc525554149)

[3.2.12 Потребители 43](#_Toc525554150)

[Глава 4. Ръководство за програмиста 44](#_Toc525554151)

[4.1 Функционална схема на системата 44](#_Toc525554153)

[4.1.1 Use case 44](#_Toc525554154)

[4.1.2 Activity диаграма 45](#_Toc525554155)

[4.1.3 Sequence диаграма 46](#_Toc525554156)

[4.2 Архитектура на системата 47](#_Toc525554157)

[4.3 Архитектура на базата данни 50](#_Toc525554158)

[4.4 Релации 60](#_Toc525554159)

[4.5 Документация за api 61](#_Toc525554160)

[Глава 5. Заменяне, SEO, Домей хостинг, Тестване и усъвършенстване 62](#_Toc525554161)

[5.1 Заменяне и трансформиране на елементи 62](#_Toc525554163)

[5.2 SEO оптимизация 62](#_Toc525554164)

[5.3 Домейн и хостинг за сайта 63](#_Toc525554165)

[5.4 Тестване 63](#_Toc525554166)

[5.5 Усъвършенстване 64](#_Toc525554167)

[Авторска справка 65](#_Toc525554168)

[Заключение 66](#_Toc525554169)

[Използвана литература 67](#_Toc525554170)

# Списък с фигури

[Фигура 1. Lingotek software 12](#_Toc525549605)

[Фигура 2. Wordbee 13](#_Toc525549606)

[Фигура 3. Методи за управление на преводи 13](#_Toc525549607)

[Фигура 4. ÜbrTà 14](#_Toc525549608)

[Фигура 5. Процес 14](#_Toc525549609)

[Фигура 6. PHP работни рамки, използвани в световен мащаб 21](#_Toc525549610)

[Фигура 7. Файлова структура на проекта 22](#_Toc525549611)

[Фигура 8. Файлова структура на docker 23](#_Toc525549612)

[Фигура 9. Интегрирани среди за PHP приложения, използвани в световен мащаб 24](#_Toc525549613)

[Фигура 10. Интерфейс на Sublime 3 25](#_Toc525549614)

[Фигура 11. Съвместимост с други устройства 27](#_Toc525549615)

[Фигура 12. Логин страница 28](#_Toc525549616)

[Фигура 13. Начална страница 29](#_Toc525549617)

[Фигура 14. Приложения 30](#_Toc525549618)

[Фигура 15. Редактиране на приложение 30](#_Toc525549619)

[Фигура 16. Редактирано приложение 31](#_Toc525549620)

[Фигура 17. Компоненти 32](#_Toc525549621)

[Фигура 18. Създаване на компонент 33](#_Toc525549622)

[Фигура 19. Създаден компонент 33](#_Toc525549623)

[Фигура 20.Генериране на подпис 34](#_Toc525549624)

[Фигура 21.Генериран подпис 34](#_Toc525549625)

[Фигура 22. Създаден превод 35](#_Toc525549626)

[Фигура 23. Редактиране на превод 35](#_Toc525549627)

[Фигура 24. Превод на чужди езици 36](#_Toc525549628)

[Фигура 25. Генериране на файлове 36](#_Toc525549629)

[Фигура 26. Сваляне на файлове 37](#_Toc525549630)

[Фигура 27. Изход от генерирането 37](#_Toc525549631)

[Фигура 28. Код за свалянето на файл 38](#_Toc525549632)

[Фигура 29. Копиране на пътя 38](#_Toc525549633)

[Фигура 30. Активни езици 39](#_Toc525549634)

[Фигура 31. Регулиране на достъпа 40](#_Toc525549635)

[Фигура 32. Код за промяна на контрола 40](#_Toc525549636)

[Фигура 33. Управление на роли 41](#_Toc525549637)

[Фигура 34. Профил 42](#_Toc525549638)

[Фигура 35. Заявка за потребител 43](#_Toc525549639)

[Фигура 36. Use Case диаграма 45](#_Toc525549640)

[Фигура 37. Activity диаграма 46](#_Toc525549641)

[Фигура 38. Sequence диаграма 47](#_Toc525549642)

[Фигура 39. Обща схема на MVC 48](#_Toc525549643)

[Фигура 40. База данни 50](#_Toc525549644)

[Фигура 41. Таблица действия 51](#_Toc525549645)

[Фигура 42. Таблица позволения 51](#_Toc525549646)

[Фигура 43. Таблица ресурси 52](#_Toc525549647)

[Фигура 44. Таблица приложения 53](#_Toc525549648)

[Фигура 45. Таблица достъп до приложения 53](#_Toc525549649)

[Фигура 46. Таблица подпис на приложение 54](#_Toc525549650)

[Фигура 47. Таблица компоненти 55](#_Toc525549651)

[Фигура 48. Таблица езици 56](#_Toc525549652)

[Фигура 49. Таблица активни езици 56](#_Toc525549653)

[Фигура 50. Таблица последно генерирани файлове 57](#_Toc525549654)

[Фигура 51. Таблица преводи 58](#_Toc525549655)

[Фигура 52. Таблица потребители 59](#_Toc525549656)

[Фигура 53. Таблица роли 60](#_Toc525549657)

[Фигура 54. Връзките в базата 61](#_Toc525549658)

[Фигура 55. Документация 62](#_Toc525549659)

Речник на използваните термини:

**MVC (Model-View-Controller)** – шаблон за дизайн Модел-Изглед-Контролер;

**TMS** (Translations Management System) – система за управление на езици.

**DOM (Document Object Model)** – съдържанието на една интернет страница;

**Ajax (Аsynchronous JavaScript Аnd XML)** – асинхронните заявки към сървъра, посредством езика JavaScript;

**ORM (Object Relational Mapping)** – архитектура за работа със записите в базата данни;

**BSD (Berkeley Software Distribution) - операционна система с отворен код.**

**SaaS (System as a Service)** – в превод „Софтуер като услуга”;

**Дебъгване** – преглеждане на код, докато програмата се изпълнява;

**URL (Uniform Resource Locator)** – протокол, описващ уеб адреси;

**Slug** – уникална част от URL адреса на страница;

**Responsive уеб сайт** – уеб сайт, който се адаптира за различни екрани;

**Service класове** – класове, съдържащи бизнес логиката на приложението;

**Storage engines** – системи за съхранение на информация в MySQL

**JSON(JavaScript Object Notation)** - текстово базиран отворен стандарт създаден за човешки четим обмен на данни.

**Api(Application programming interface)** - Приложно-програмен интерфейс.

# Увод

Бъдещето е в технологиите. Те заемат изключително важна част от живота на съвременните хора и се използват в почти всяка дейност от ежедневния ни живот – в училищата, в работата, в свободното ни време. Една от основните технологии, използвана от целия свят, е Интернет. Той предоставя възможността да бъде достъпна голяма и разнообразна информация от различни места. Позволява на хората да комуникират помежду си чрез различни и бързи начини като имейли, чатове и видео разговори. Също така се използва за онлайн-магазини и дори онлайн-банкиране. Ползите от Интернет могат да бъдат безкрайни.

С навлизането на технологиите във всички сфери, все повече навлизат и системите за микрообслужване. Голяма част от служителите на различни по дейност компании работят пред компютър. Успехът на една компания, било тя голяма или малка, зависи от добрата организация на задачи и проекти, както и на комуникацията на хората, работещи върху тях. Една система за превеждане би улеснила работата. Предимствата на този тип софтуер са много. Те позволяват на служителите да работят върху задачите си и да обновяват статуса на задачите, чрез постоянния достъп до системата, по всяко време и от всяка точка на Земята.

Разработеният в дипломната работа програмен инструмент би улеснил комуникацията между потребителите в една фирма и би добавил допълнителна ефективност и бързина на потребителите му.

Друго предимство на разгледаната система е пазенето на версии на файловете генерирани от потребителя. Наличната търсачка позволява бързо и ефективно намиране на информация от потребителите.

Системата за преводи има приятен и удобен интерфейс, който го прави лесен за използване и без специално обучение. Тя се базира върху уеб технологиите, което позволява използването ѝ чрез интернет достъп.

Системата за преводи обединява предимствата на най-важните, известни, подобни инструменти, за да се преведе един проект лесно и бързо, без значение от каква мащабност е. Такива системи са подходящи за компании, които изпълняват проекти с много участници или в ИТ сферата за разработка и поддържане на различни приложения и продукти.

## Цели на дипломната работа

Основната цел на разработената дипломна работа е да се създаде уеб приложение, микросистема – система за преводи. Това наложи дефинирането на следните задачи за изпълнение:

* Да се създаде платформа, чрез която бързо, лесно и без много усилия да се изгради ефективна уеб система.
* Да се създаде платформа, която лесно се интегрира с други микросистеми.
* Да се предложи платформа, насочена към малкия и средния бизнес, които нямат възможност да вложат голям паричен ресурс.
* Да се създаде подробна информация как да се използва системата от програмистите.
* Да бъде лесна за инициализиране.

## Структура на дипломната работа

Дипломната работа се състои от увод, 5 глави, заключение, използвана литература и приложение.

* **Увод** – представя въведение в тематиката, целите и предназначението на дипломната работа.
* **Първа глава – Translations Management System платформи използвани в уеб пространството –** дава се информация за това какво е TMS и за какво се ползва, а също така се описват и няколко популярни TMS платформи.
* **Втора глава – Технологии и средства за разработка, които** разглеждат използваните технологии и средства за реализацията на дипломната работа: Apache, PHP и PHP библиотеки, MySQL, HTML, CSS, JavaScript и JavaScript библиотеки, Photoshop и Sublime.
* **Трета глава – Описание на системата**: описват се изискванията към платформата, основните модули, тяхната функционалност. Всичко това е съпроводено с детайлен снимков материал.
* **Четвърта глава – Ръководство за програмиста** – платформата се описва детайлно чрез Use Case диаграми, показана е пълна диаграма на базата данни и файловата структура, със съответните към тях описания. Описва се използваният шаблонен модел (MVC), с примери от програмния код.
* **Пета глава** – Описание на процеса на тестване, SEO, домейн, хостинг и усъвършенстване.
* **Заключение** – Дава се обобщение на цялата платформа, посочват се перспективите на проекта.
* **Използвана литература**
* **Приложение** – Към дипломната работа е приложен компакт диск, който съдържа файловете на графичния интерфейс на системата, уеб приложението, както и SQL dump на базата данни.

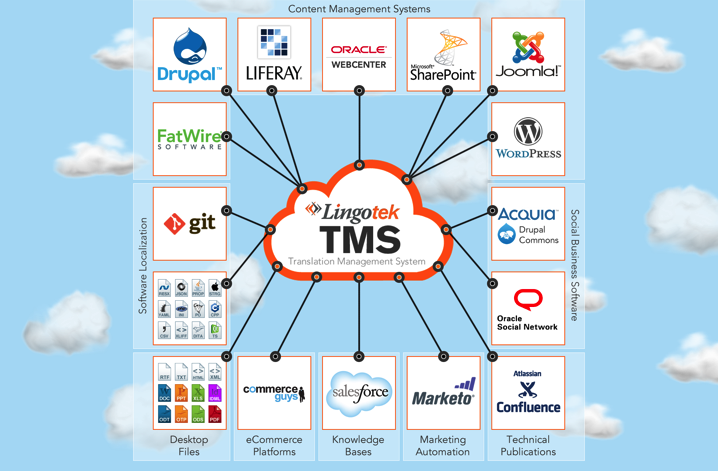
Глава 1. TMS платформи, използвани в уеб пространството

**TMS** (Translations Management System) или още система за превеждане на текстове е уеб приложение, с което посредством потребителски интерфейс, се предоставя възможност за превеждане на текстове към дадени приложения. В зависимост от сложността на софтуера, той може да управлява оценките и планирането, график, контрол на разходите и управление на бюджета, разпределение на ресурси, софтуер за сътрудничество, комуникация, вземане на решения, управление на качеството и документация или административни системи [1].

Използването на системи за превеждане на съдържание води началото си от 90-те години на 20-ти век. С навлизането на Интернет в ежедневието на все повече хора се появява и нуждата от подобни системи за изграждане и превеждане на проекти. Тогава били създадени и първите TMS платформи. В момента, след повече от 20 години, в уеб пространството съществуват десетки системи за превеждане на проекти, всяка от тях реализирана на конкретен програмен език и за определени цели.

## Lingotek

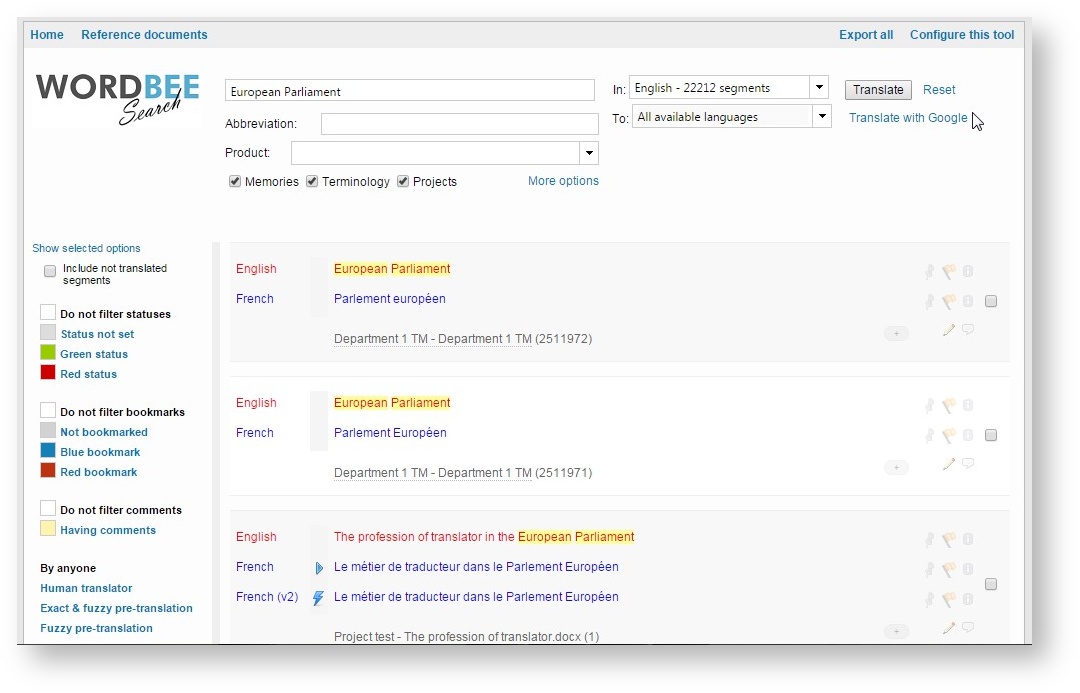
Lingotek е първият, който въвежда технологията за управление на превода на базата на клауд (TMS) в САЩ през 2006 г. Компанията стартира напълно онлайн, базирана на компютър, компютърно подпомогната преводна система (CAT) и провокира интегрирането на преводните памети TM) с машинен превод.



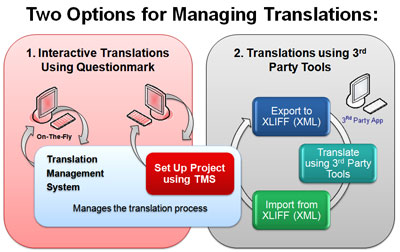
Фигура 1. Lingotek software

## Wordbee

През 2008 г. Хосе Вега и Стефан Бьоминг стартират Wordbee, за да направят точно това, като развиват Wordbee Translator като уеб-базиран инструмент CAT. Пълно управление на превода в облака: Целият софтуер, който бизнесът трябва да управлява проектите за превод.



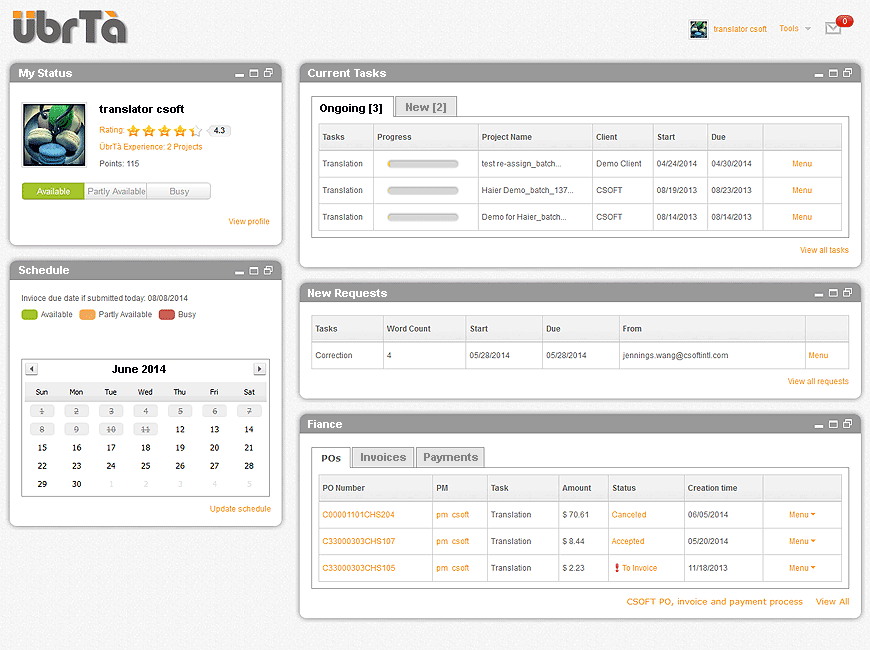
Фигура 2. Wordbee



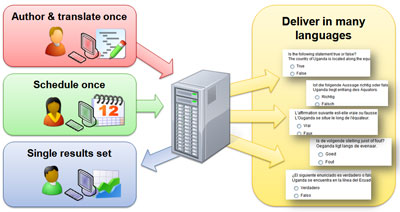
Фигура 3. Методи за управление на преводи

## ÜbrTà

ÜbrTà е вътрешната система за управление на превода на CSOFT (TMS). Достъпен за клиенти, лингвисти, мениджъри на проекти и всички други заинтересовани страни в преводите, централизираният и автоматизиран подход на ÜbrTà осигурява максимална ефективност и прозрачност.



Фигура 4. ÜbrTà



Фигура 5. Процес

Глава 2. Технологии и средства за разработка



## Технологии за разработка

*При избор на технологиите за разработка на тази дипломна са взети под внимание факторите, от които зависи целия процес:* време за реализация, осигуряване на лесен достъп до системата с минимално необходим ресурс, съвременни технологии за постигане на висока степен на интерактивност и бързодействие. Поставената цел е системата да бъде максимално удобна за използване, от възможно най-голям брой потребители.

Гореизброените фактори, както и предназначението на системата – да се използва в уеб пространството, предполагат тя да бъде реализирана като интерактивно уеб приложение. Подходящи за тази цел програмни езици са ASP.NET и PHP. Вторият от посочените е изцяло с отворен код и напълно безплатен за разработчиците.

В комбинация с езика за сървърно програмиране е необходима и среда за управление на база от данни. За конкретния случай и изисквания към системата, най-подходяща е MySQL база от данни, която също е безплатна и широко разпространена.

За да бъде уеб приложението интерактивно, в разработката на модулите му са използвани HTML5, CSS3 и JavaScript. То е написано на PHP v7.1 – език за програмиране, използвайки Phalcon, който предоставя работната рамка и редица помощни средства за обектно-ориентиран стил на програмиране. Този стил предоставя на системата висока степен на приложимост в множество различни ситуации, улеснявайки разработването и разширяването ѝ с нови модули. Системата за управление на базата данни е описана и обработвана чрез MySQL v5.5, като базата данни е в трета нормална форма за оптимално лесен достъп. Използвани са JavaScript библиотеките jQuery v2.2.3 и jQuery UI v1.11 за постигането на по-голяма атрактивност и интерактивност на интерфейса.

За разработка и тестване на дипломната работа е използван Apache v2 – локален сървър, работещ под операционната система Ubuntu. По време на разработката на дипломната работа е използвана система за контрол на версиите – Git.

### Ubuntu OS

Ubuntu е изключително гъвкава и динамична операционна система, базирана на Unix, подобно на операционна система на Apple – OS X. За разлика от OS X и Windows, тя е безплатна, с отворен код и работеща под редица устройства – персонални компютри, таблети и смартфони. Друго предимство на Ubuntu ОС пред другите операционни системи е това, че всички необходими ресурси за разработка на едно PHP приложение са на няколко команди разстояние.

### Nginx

Nginx е HTTP сървър с отворен код, който работи на много операционни системи, включително UNIX, Linux, OS X, Windows и други. Nginx е бил разработен, за да предостави сигурен, ефикасен и гъвкав сървър, който поддържа HTTP услуга.   
От 1996 г. насам, той е най-популярният уеб сървър в Интернет и според проучване, над 63% от сайтовете работят чрез него [3]. Сървърът поддържа едни от най-използваните програмни езици – PHP, Perl, Python и Tcl. Тъй като се разпространява със свободен лиценз и притежава гъвкава архитектура, всеки разработчик може да променя Nginx според собствените си нужди [3]. Освен това в уеб пространството може да бъде намерена огромна публична библиотека с приставки (add-ons) и модули. Версията на Nginx, с която е разработен проектът е 1.14.

### PHP

PHP, чието съкращение идва от Hypertext Preprocessor – хипертекстов предпроцесор, е скриптов език, който се изпълнява върху сървър и дава възможност да се създават динамични уеб приложения. Създаден през 1994 година от Расмус Лердорф, взаимства много неща от вече утвърдените по онова време програмни езици като C, Java, и Perl. За близо четвърт век съществуване са създадени 4 главни версии, всяка подобрение на предишните [2, 3]. За настоящата дипломна работа е използвана версия 7.1, независимо че съществува версия 7, понеже седмата версия на езика все още не е изцяло стабилна и широко използвана.

За редактор на PHP код е използвана интегрираната среда Sublime, поради богатия набор от възможности, които тя предлага. Sublime е сред лидерите в средите за разработване на PHP приложения.

### MySQL

MySQL е система за управление на релационни бази от данни, написана на C и С++. Нейният SQL парсър (анализатор) от своя страна е написан на Yacc. Тя е с отворен код и затова е една от най-предпочитаните сред разработчиците. MySQL работи на много различни платформи – включващи AIX, BSDi, FreeBSD, HP-UX, GNU/Linux, Mac OS X, NetBSD, Novell NetWare, OpenBSD, OS/2 Warp, QNX, SGI IRIX, Solaris, SunOS, SCO OpenServer, SCO UnixWare, Tru64, Windows 95, Windows 98, Windows ME, Windows NT, Windows 2000, Windows XP и други версии на Windows. Програмните езици, които могат да работят с нея, чрез използването на специфични програмни интерфейси са:  
C, C++, C#, Delphi, Eiffel, Smalltalk, Java, Lisp, Perl, PHP, Python, Ruby, Tcl и  
други [2, 3, 12].

За настоящата дипломна разработка е използвана версия 5.5 на MySQL за съхранение на цялата информация, която потребителите въвеждат, под формата на таблици в 3-та нормална форма. Тази технология има редица предимства пред другите популярни storage engines разпространени с MySQL. Най-ключовите са: работата с транзакции и заключване на таблици, чиито данни биват прочитани, по-ефективно използване на паметта и процесорната мощ на сървърната машина.

### HTML

HTML е маркиращ език, който се използва за структуриране и оформяне на уеб страници. Текущата актуална версия на езика е HTML5. Тя се поддържа от последните версии на масовите браузъри, без да е необходимо използването на допълнителни скриптове и „хакове“.

HTML е съкращение от “**H**yper**T**ext **M**arkup **L**anguage” или на български „език за маркиране на хипертекст”, тъй като описанието на уеб документа става чрез елементи, наречени маркери [14]. Те се състоят от тагове и ъглови скоби, например <html>. Чрез тези елементи става оформянето на страницата – заглавна част, съдържание, навигация, списъци, връзки (препратки) и много други. Има *задължителни* елементи, без които една страница не може да се визуализира правилно в браузъра и *незадължителни*.

Едно от най-големите предимства на HTML страниците е, че те са се наложили като уеб стандарт и документите, оформени по този начин могат да се разглеждат не само на персонални компютри, но и на мобилни устройства – таблети и смартфони.

В настоящата дипломна работа, HTML се използва във всички изгледи (views) на системата. Във файловете маркиращият език се смесва с PHP, като по този начин се създават динамични изгледи на системата. Благодарение на средата за разработка YiiPHP, се постига добро ниво на абстракция и отделяне на бизнес логиката от презентационна част.

### CSS

CSS (съкращение от Cascading Style Sheets) представлява език за описание на стилове. Той се използва, за да подобри или оформи визуално съдържанието на уеб страница в браузъра, написана на език за маркиране (XML, HTML). Чрез прибавянето на CSS се разделят съдържанието и структурата на документа от неговото представяне. Освен това в CSS има редица атрибути, които се използват за елегантното и красиво оформление на елементи и текст [14].

По-горе беше посочено, че HTML страниците са изградени от елементи или тагове. На всеки един от тези елементи може да бъде присвоен уникален идентификатор (id) или клас (class). Последният може да бъде зададен на повече от един елемент. В CSS файла се описва какъв стил искаме да има елемент, притежаващ съответния идентификатор или клас, например размер и цвят на шрифт, ширина, височина, рамка, отстояние, фон и много други. Освен чрез id и class, HTML елементите могат да бъдат достъпени и чрез селектор. За тази цел се задава име на самия елемент.

Езикът CSS предлага над 100 свойства, чрез които една уеб страница може да се превърне в истинско произведение на изкуството. В помощ на разработчиците и дизайнерите в Интернет пространството има много сайтове, чрез които може да се генерират валидирани css фрагменти от код. Потребителят може да зададе структура на страницата, след което да изтегли готови css и html файлове. За дипломната разработка не са използвани автоматични генератори на код, тъй като дизайнът на системата е специфичен и преправянето на готов код би било по-трудоемко, отколкото написването му.

Във версия 3 на CSS има много нови свойства, които се засичат и разчитат от последните версии на най-разпространените браузъри. Това са различни видове анимации и трансформации на елементи и шрифт. Други нови изключително полезни свойства са Media Queries. Чрез тях се засича големината на екрана и в зависимост от размера, може да се зарежда различен стил. По този начин, без добавянето на JavaScript, страницата би могла да изглежда добре едновременно на компютър и мобилно устройство.

В CSS има десетки други полезни свойства, като например вграждане на шрифт, контрол на гласа при автоматичните четци, (използван от незрящи хора), многоколонен изглед, преоразмеряване на елементи и други.

### JavaScript и jQuery

JavaScript е динамичен прототипен език за програмиране, който е слабо типизиран. Бил е създаден, за да се имплементира в уеб браузърите, така че да може да комуникира с потребители, да контролира самия прозорец и да променя съдържанието на страницата. Синтаксисът му е повлиян от езика C, но използва много имена и конвенции от Java [14].

Най-честата употреба на JavaScript е като вградени функции в HTML код, взаимодействайки си с DOM (Document Object Model) на страницата. Някои прости примери, в които се използва:

* За зареждане на ново съдържание на страницата или изпращане на данни до сървър, чрез AJAX, без презареждане на цялата страница.
* Анимиране на елементи (плавно появяване или изчезване, преоразмеряване, преместване и други).
* За интерактивно съдържание (компютърни игри, аудио и видео).
* Валидиране на форми преди данните да се изпратят към сървъра.
* Предаване на информация за предпочитанията на потребителя, която обикновено се използва за уеб анализ и статистика.

Тъй като JavaScript е широко използван език, се поддържа от всички популярни браузъри. Понякога те реагират по различен начин, затова важна част от създаването на един добре работещ код, е тестването. Разработчикът трябва да бъде запознат със стандартите на всеки браузър или да използва код, който проверява кой е настоящият браузър и така да реагира с адекватно поведение. За улеснение, програмистите използват библиотеки, които взимат под внимание различията между браузърите. Една от популярните библиотеки е jQuery. Тя е съвкупност от най-използваните функции в уеб пространството – напълно безплатна и може да бъде изтеглена от официалния ѝ сайт – http://jquery.com/. В проекта се използва версия на библиотеката – 2.2.3. Характерното за нея е, че е премахната поддръжката на стари браузъри като Internet Explorer 6, и 7 и 8. Дори от Microsoft наскоро обявиха, че спират каквато и да е поддръжка на всички версии до 10-та и подканиха всеки един, който ползва такава версия да премине към по-нова и актуална.

Ако не използваме jQuery, а чист JavaScript и искаме да създадем някоя широко използвана функция, като например показването на скрит елемент на екрана, трябва да напишем в най-добрия случай няколко десетки реда код. С помощта на библиотеката се добавят само няколко реда:

*<script>*

*$(document).ready(function() {*

*$("#element").fadeIn(400);*

*});*

*</script>*

Този код означава, че когато страницата се е заредила, елемент с *id = "element"* ще се покаже плавно на екрана. Така jQuery предлага голям набор от различни ефекти, чрез добавяне на само един ред код. В допълнение на функционалността, която предлага библиотеката, в *Sova* *PMS* е използвано и допълнението jQuery UI. То предлага около 20 често използвани готови функционалности, но за целите на проекта е вградена малка част от тях и по този начин не се налага зареждане на цялото допълнение.

### React (JavaScript library)

React (известен също като React.js или ReactJS) е библиотека JavaScript [2] за изграждане на потребителски интерфейси. Поддържа се от Facebook и общността от отделни разработчици и компании.

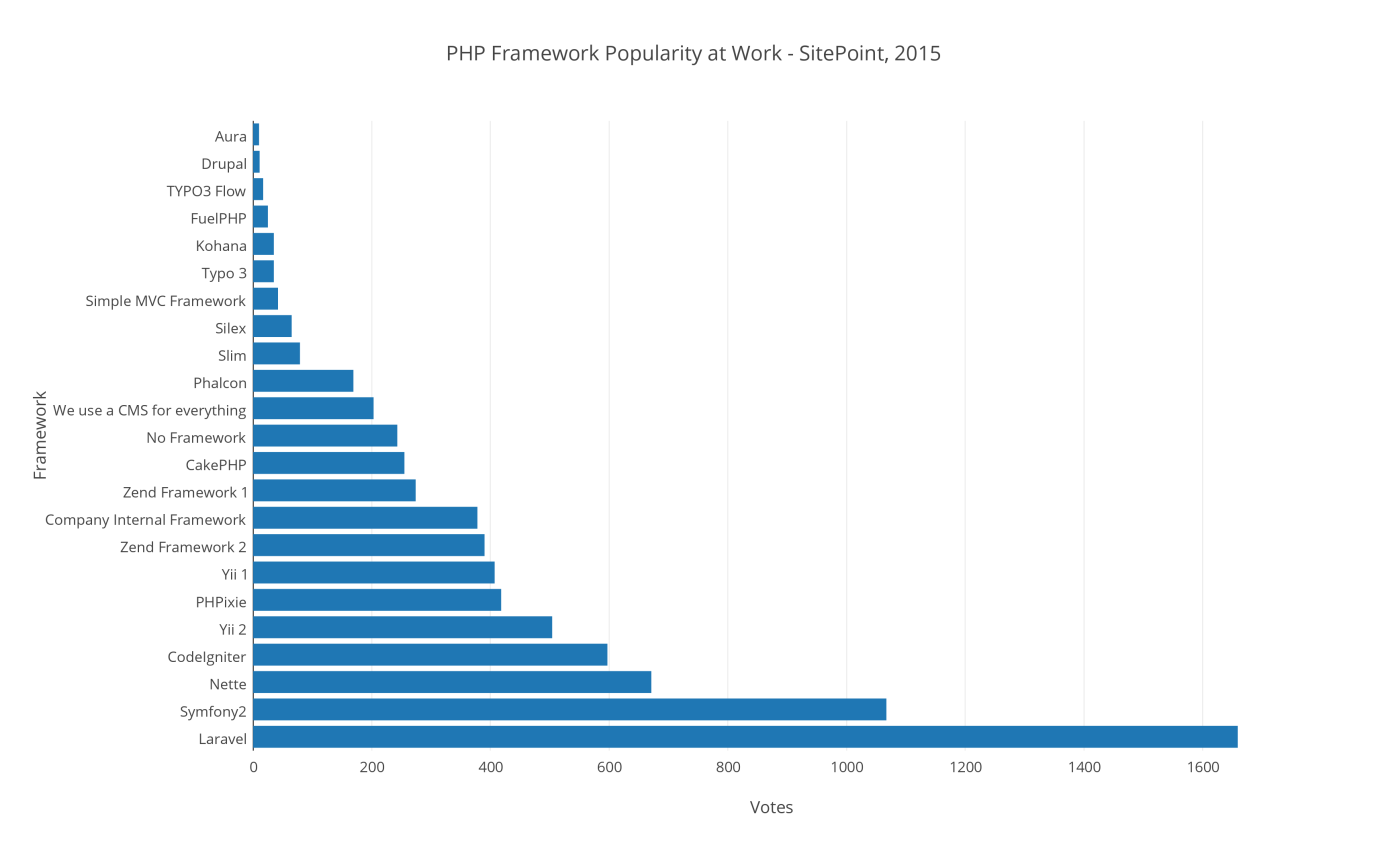
React може да се използва като основа при разработването на едностранични или мобилни приложения. Комплексните реакции обикновено изискват използването на допълнителни библиотеки за държавно управление, маршрутизиране и взаимодействие с API.

### Docker

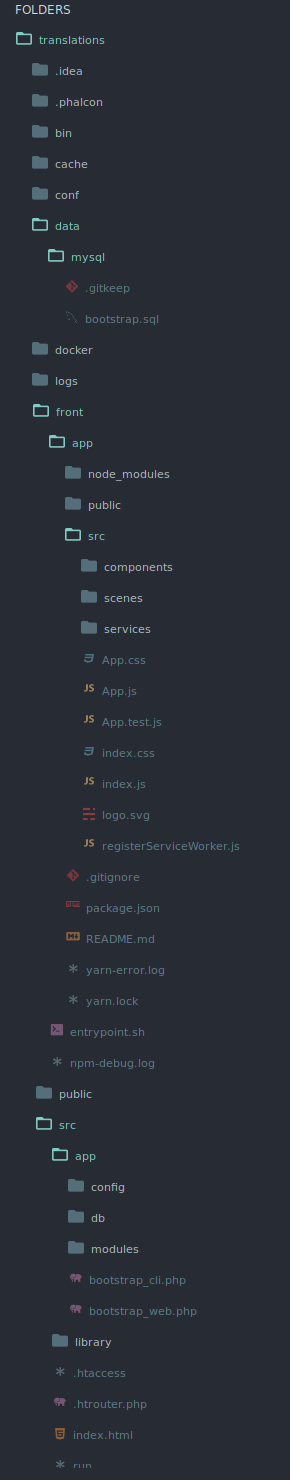
Docker е компютърна програма, която изпълнява виртуализация на ниво операционна система, известна още като "containerization". Тя беше пусната за първи път през 2013 г. и е разработена от Docker, Inc.

Докер се използва за стартиране на софтуерни пакети, наречени "контейнери". В един типичен пример за използване един контейнер изпълнява уеб сървър и уеб приложение, а втори контейнер изпълнява сървър на база данни, който се използва от уеб приложението. Контейнерите са изолирани един от друг и обединяват свои собствени инструменти, библиотеки и конфигурационни файлове; те могат да комуникират помежду си чрез добре дефинирани канали. Всички контейнери се управляват от едно ядро ​​на операционната система и по този начин са по-леки от виртуалните машини. Контейнерите се създават от "изображения", които определят точно тяхното съдържание. Изображенията често се създават чрез комбиниране и модифициране на стандартни изображения, изтеглени от хранилища.

### Phalcon – PHP framework

Phalcon е PHP уеб рамка, базирана на модел-view-controller (MVC) модел. Първоначално пуснато през 2012 г., това е рамка с отворен код, лицензирана съгласно условията на лиценза BSD. За разлика от повечето PHP рамки, Phalcon се изпълнява като разширение на уеб сървър, написано на Zephir и C, което има за цел да увеличи скоростта на изпълнение, да намали използването на ресурсите и да обработва повече HTTP заявки на секунда от съпоставимите рамки, написани основно в PHP. Един от недостатъците на този подход е, че на сървъра е необходим корен / административен достъп, за да се инсталира Phalcon чрез изграждане на потребителски двоичен код или използване на предварително компилирано устройство. 

Фигура 6. PHP работни рамки, използвани в световен мащаб



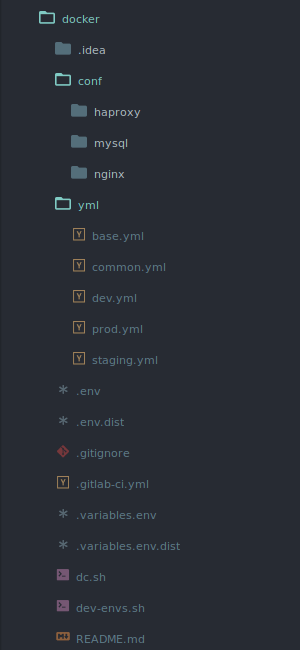
Файловата структура на работната рамка е следната:

* app – папка, в която е поместена основната бизнес логика.
* db – папка, съдържаща скриптове, инициализиращи и попълващи базата данни с информация.
* public – папка с публичен достъп, която обикновено съдържа CSS файлове, JavaScript скриптове, шрифтове, изображения и други подобни файлове.
* front – папка, в която се пазят презентационните файлове (views) на системата. Също така в нея се намират файловете, които съдържат превод на текстове за многоезични сайтове.
* library – папка, в която се съхраняват множество библиотеки с цел улесняване процеса на разработка.
* modules – съдържа няколко модула на приложението един за админската част и един за api.
* vendors – в нея се съхраняват всякакви допълнителни за системата файлове като библиотеки и разширения, предназначени за подобряване на функционалността на системата. Обикновено тези файлове биват добавяни в папката от Composer, който е отговорен за управлението на допълнителните ресурси.

Фигура 7. Файлова структура на проекта

Файловата стуктура на папката Docker:

* conf- в папката за конфигурация се съдържат още три папки за настройки върху
  + haproxy
  + mysql
  + nginx
* yml- в тази папка се съдържат няколко файла за конфигурацията на различни среди.



* .env- в този файл се пази в каква среда се работи в момента.
* .variables.env- в този файл се съдържат всички променливи изнесени от конфигурационните файлове на проекта с цел, централизирано пазене на променливите.

Фигура 8. Файлова структура на docker

### Adobe Photoshop CS5

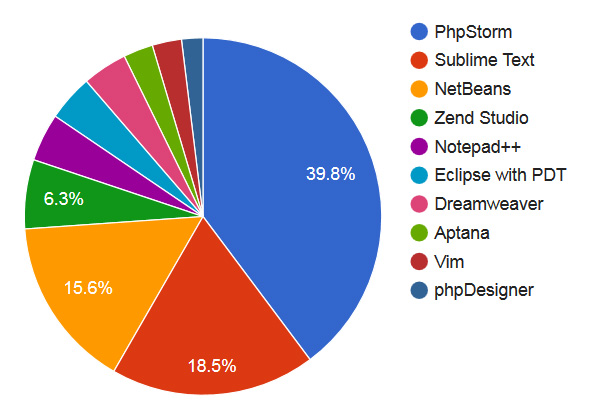
Adobe Photoshop e професионална програма за обработка на растерна графика. Продуктът е избран като основен софтуер за създаването на дизайна на системата, логото, както и всички изображения, включени в дипломната работа.

Adobe Photoshop е една от водещите програми в световен мащаб за дигитална обработка на изображения, както и за създаване на графики, илюстрации, скици, шаблони и дизайни за уеб сайтове. В началото е била разработена само за компютри от серията Макинтош, но днес работи на най-разпространените операционни системи като Windows, ОS X, дори Linux и Ubuntu, (тук се използва специално десктоп приложение, което създава виртуален диск, на който могат да се стартират програми за Windows).

Photoshop поддържа различни цветови режими: *RGB* (red, green, blue), *CMYK* (cyan, magenta, yellow, key (за черен ключов цвят), *Greyscale* (степени на сивото), *Lab* (за печат) и други.

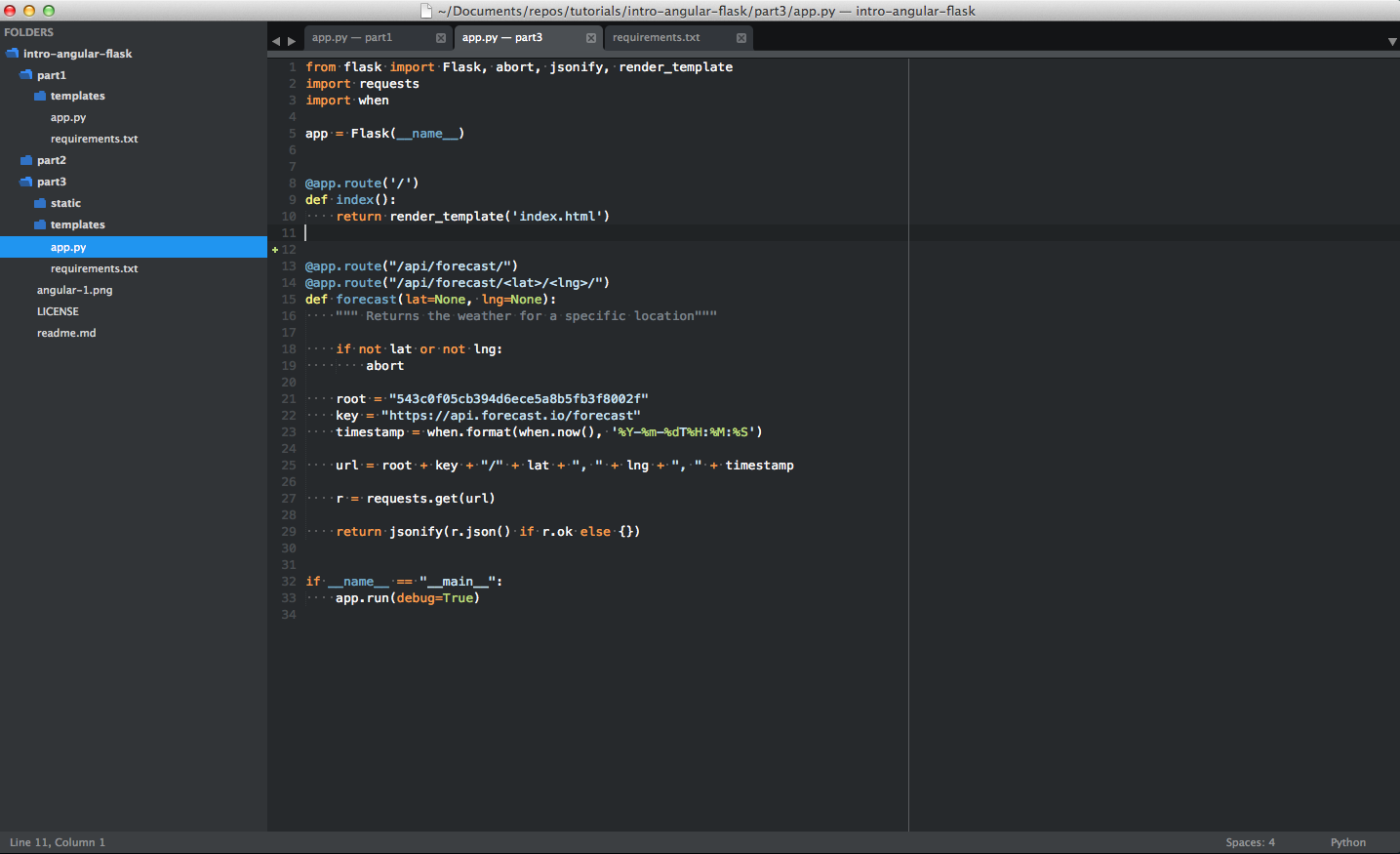
### Sublime

За настоящата дипломна работа е използвана версия 3 на Sublime. Той е междуплатформен софтуер за обработка на изходен код и текстов редактор с приложно-програмен интерфейс(API) написан на Python. Функционалността му може да бъде допълвана от потребителите чрез плъгини. Повечето от допълненията са с лиценз за свободен софтуер и се разработват и поддържат от потребителите.Според проучване направено от SitePoint през 2014 година, Sublime е на второ място в класацията за най-използвана интегрирана среда за разработка на софтуер с PHP [11].

****

Фигура 9. Интегрирани среди за PHP приложения, използвани в световен мащаб

Всичко това се дължи на възможностите, които програмата дава на разработчиците, които я ползват. Сред едни от най-добрите ѝ страни са: изключително доброто подсказване (type hinting) на функционалности по време на писане на код; добрата работа с *xDebug* (софтуер за дебъгване – преглеждане на код, докато програмата се изпълнява) и *Profiling* (софтуер, който дава възможност за следене на бързодействието на програмата и откриване на вероятни слаби места); удобният интерфейс и възможностите за търсене в целия проект; големия брой добавки налични за програмата и още много други.



Фигура 10. Интерфейс на Sublime 3

Изгледът с панели и инструменти в Sublime е изключително удобен и интуитивен. В дясната част се намира самия редактор на код. Вляво е разположена файловата структура на проекта, представена в дървовиден вид, а най-отгоре са разположени бутоните за бързи операции, както и пътя на отворения файл в проекта.

Глава 3. Описание на системата



## Изисквания

Преди да бъде започната разработката на системата, към нея са формулирани няколко основните изисквания. Това са:

* лесна употреба;
* лесна разбираемост;
* атрактивност;
* гъвкавост и възможност за разширение;
* достъпност и съвместимост с различни устройства.

### Лесна употреба

*Системата трябва да бъде максимално изчистена и опростена* – потребителите трябва да достигат до крайната цел без да се въвличат в ненужни и сложни операции. За постигане на удобство и лекота при употреба, се взима под внимание не толкова графичния дизайн на елементите, колкото тяхното разположение и достъпност. Целта е потребителите да се запознаят и да свикнат с интерфейса за максимално кратко време.

### Разбираемост

Системата трябва да бъде интуитивна и потребителите да се справят успешно, без да са преминали специално обучение. Ето защо тя е създадена с помошта на няколко езика за програмиране, а за тези, които изпитват трудности, на редица места в сайта е предоставена подробна обяснителна информация.

### Атрактивност

За да може да се превърне в успешен продукт, платформата трябва да привлича и да задържа потребителите, които я ползват. Това се постига като се създаде добро потребителско преживяване благодарение на JavaScript и CSS 3.

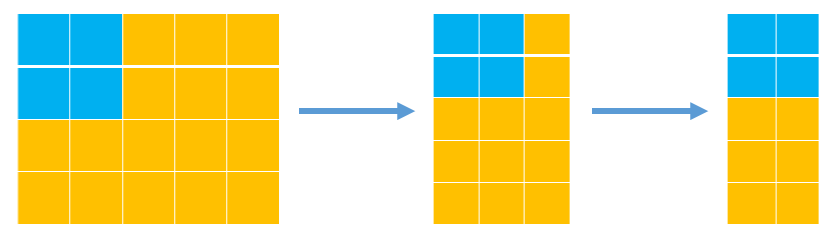
### Гъвкавост и възможност за разширение

Системата за управление на преводи е разработена върху идеята за постоянно усъвършенстване. Тя трябва да е изключително гъвкава и да отговаря на постоянно променящата се уеб среда. Версия 1 на платформата включва всичко необходимо за създаване на обикновен проект с негови компоненти и преводи, като за в бъдеще идеите за нейното разширение са многобройни.

### Достъпност и съвместимост с различни устройства

В най-близко бъдеще все по-голяма част от трафика в интернет ще се формира от мобилните устройства. За да е възможно потребителите да използват уеб страницата от всякакви устройства, трябва да се проектира дизайн, така че той да бъде адаптивен за всички разделителни способности или така нареченият Responsive дизайн. Идеята е да се постигне лесно четене и навигация с минимално оразмеряване за потребителите. Разбира се могат да се направят отделни дизайни за отделните видове устройства, но това е доста по-скъпо и трудно за изпълнение, а в този конкретен случай и ненужно. CSS фреймуъркът Bootstrap има изключително много функционалности, които ще ни помогнат да направим дизайна адаптивен и лесен за работа от всички устройства.

На фигурата по-долу е показано как би могло да бъде имплементирано преподреждането на елементите в една уеб страница при различните разделителни способности на отделните типове устройства.

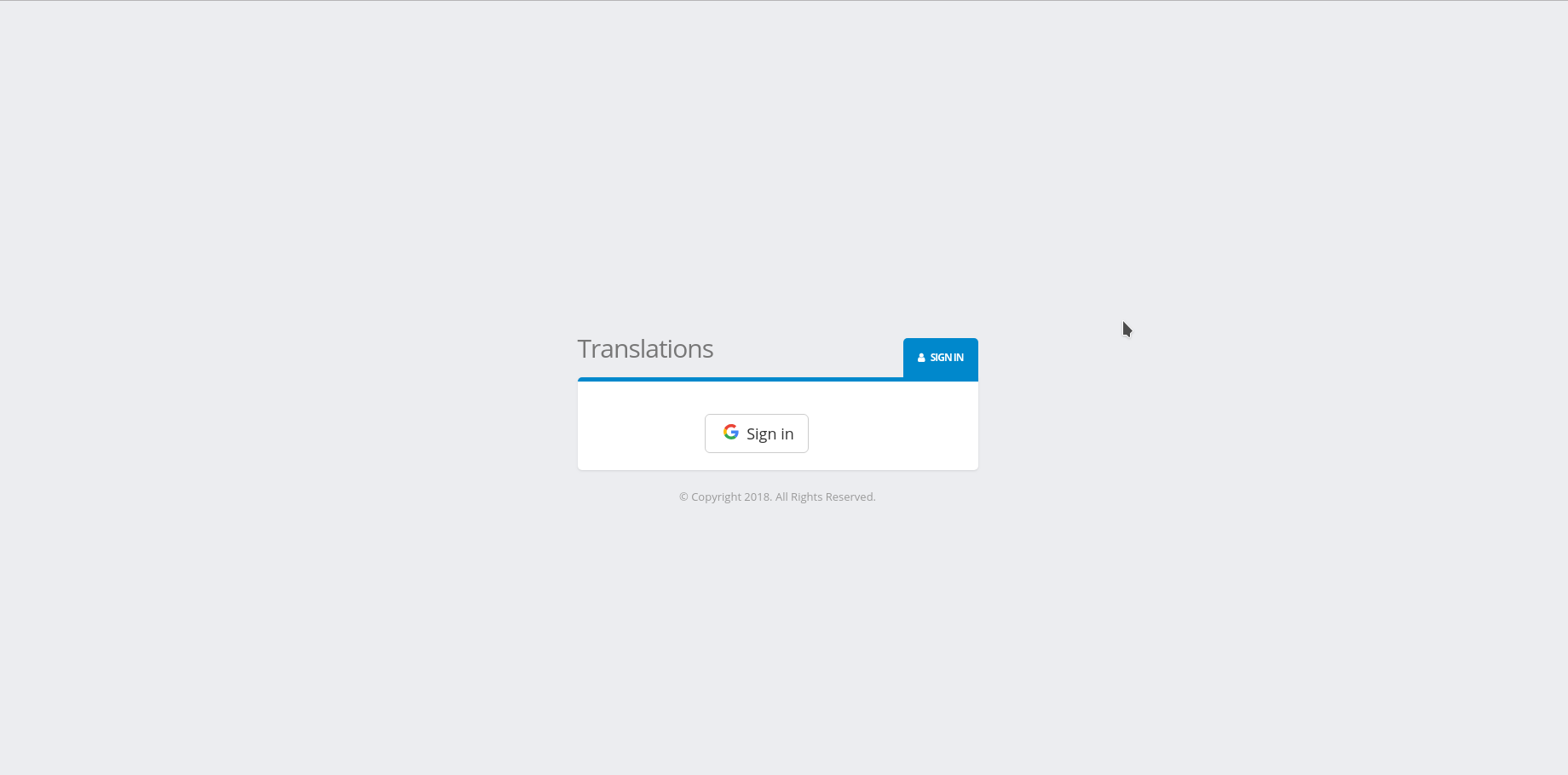


Фигура 11. Съвместимост с други устройства

## Същност на приложението

### Логин

Първата страница, на която попада потребителят след зареждане на уеб адреса в браузъра, е логин страницата.



Фигура 12. Логин страница

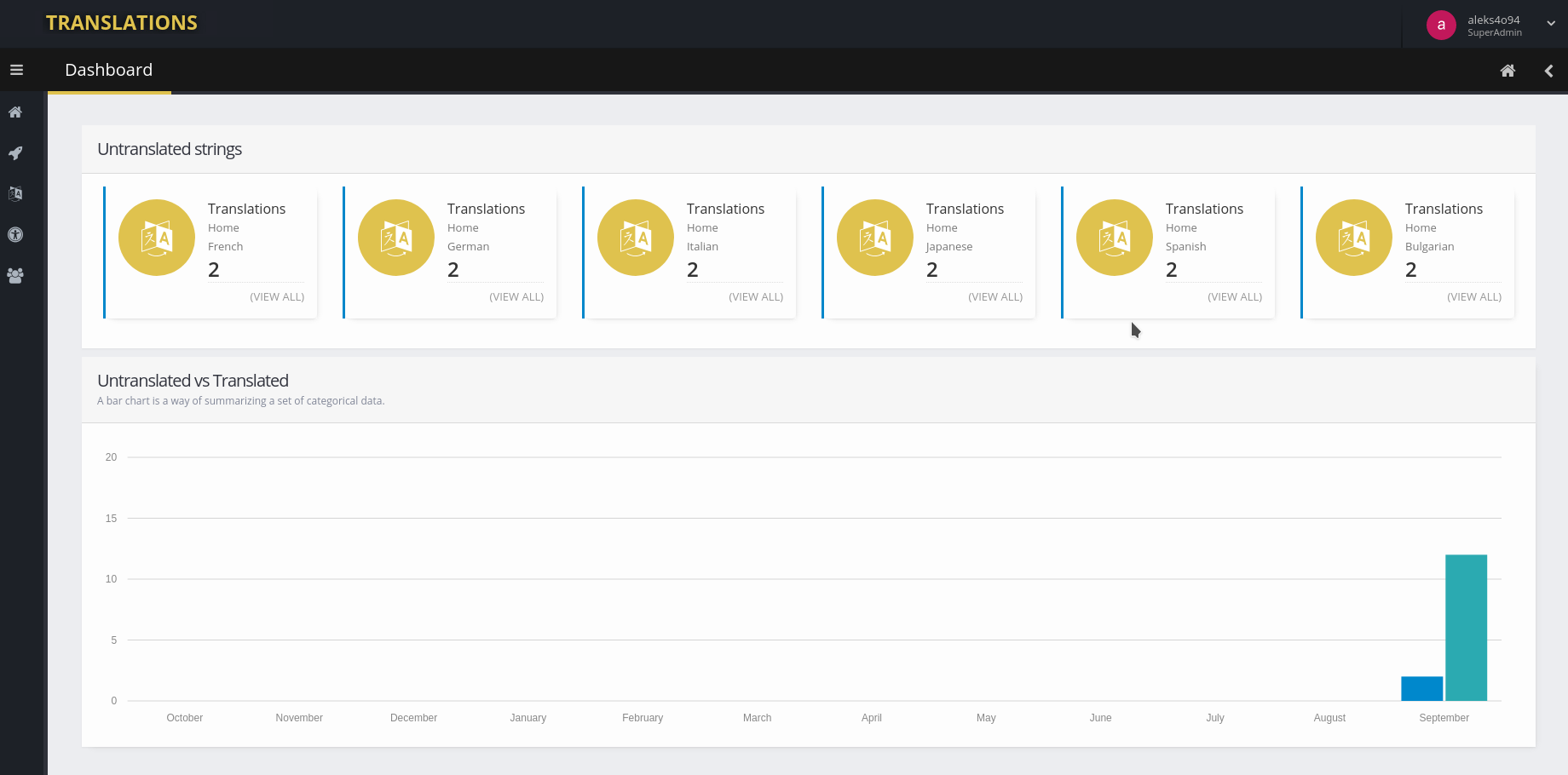
На страницата се визуализира бутон за влизане с Google профил. При вече успешно влизане в системата, се запазва в бисквитките.

При първоначално вписване се визуализира форма. Формата представлява две текстови полета за имейл и парола, както и линк към забравена парола.

При написан правилен имейл се визуализира съобщение с поздрав и се зарежда снимката на потребителя. Създава се сесия с името на потребителя.

При попълване на информация, несъответстваща на тази в базата данни за съответния потребител, се появява съобщение, оповестявайки за грешка.

### Начална страница



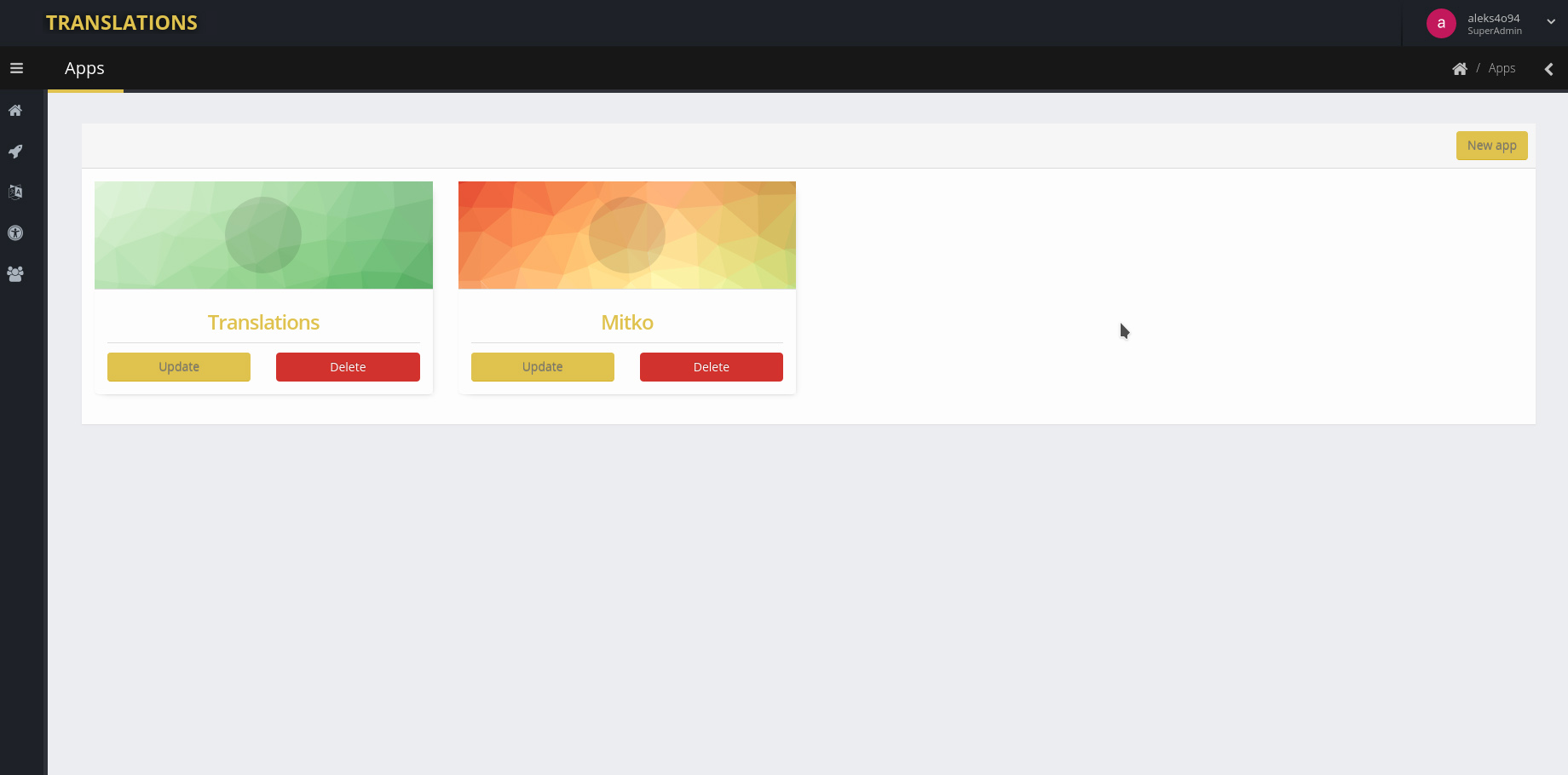
Фигура 13. Начална страница

В началната страница на приложението се визуализират две кутийки относно преведените и непреведените ключове.

В първата кутийка се показва кои са непреведените текстове с линк към тях, както и към кое приложение и кой компонент са.

На втората кутийка се изобразява диаграма със съотношението преведени непреведени преводи за последните месеци.

### Приложения

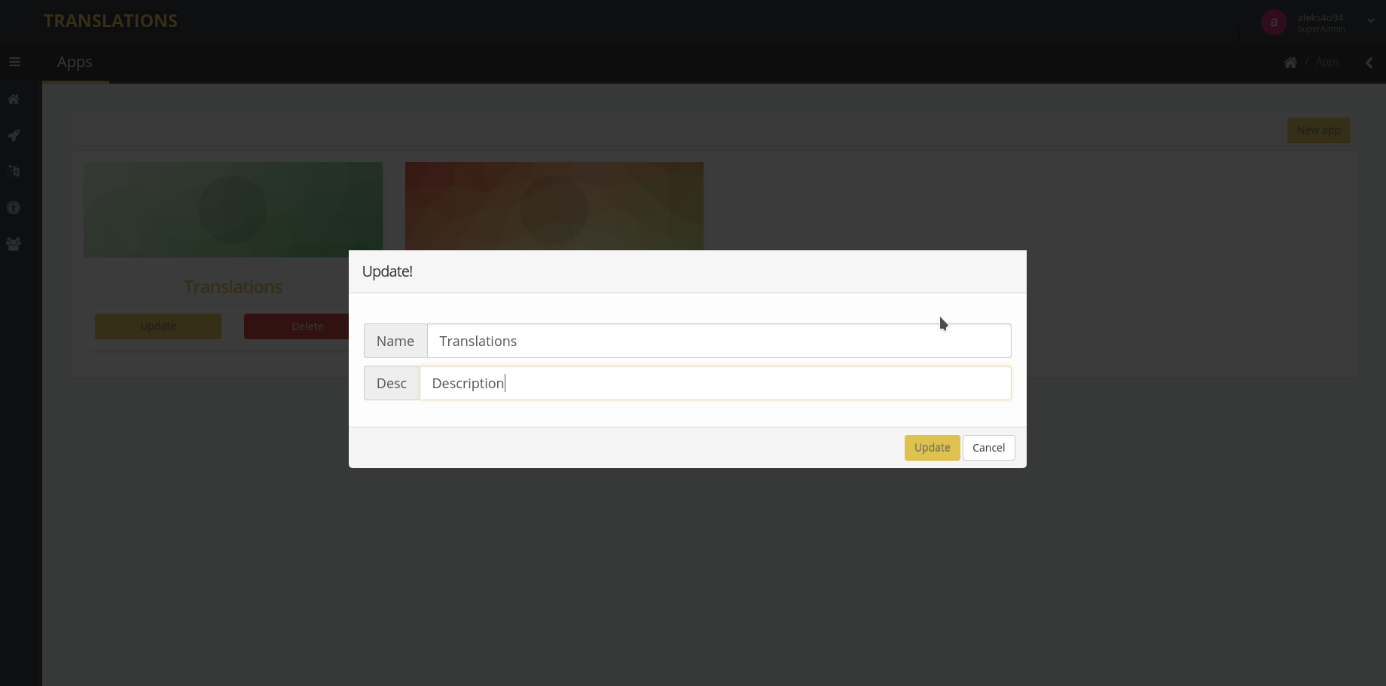


Фигура 14. Приложения

В страницата с приложения се вижда главен прозорец, който обединява всички приложения като в десния му край има бутон за създаване на ново приложение.

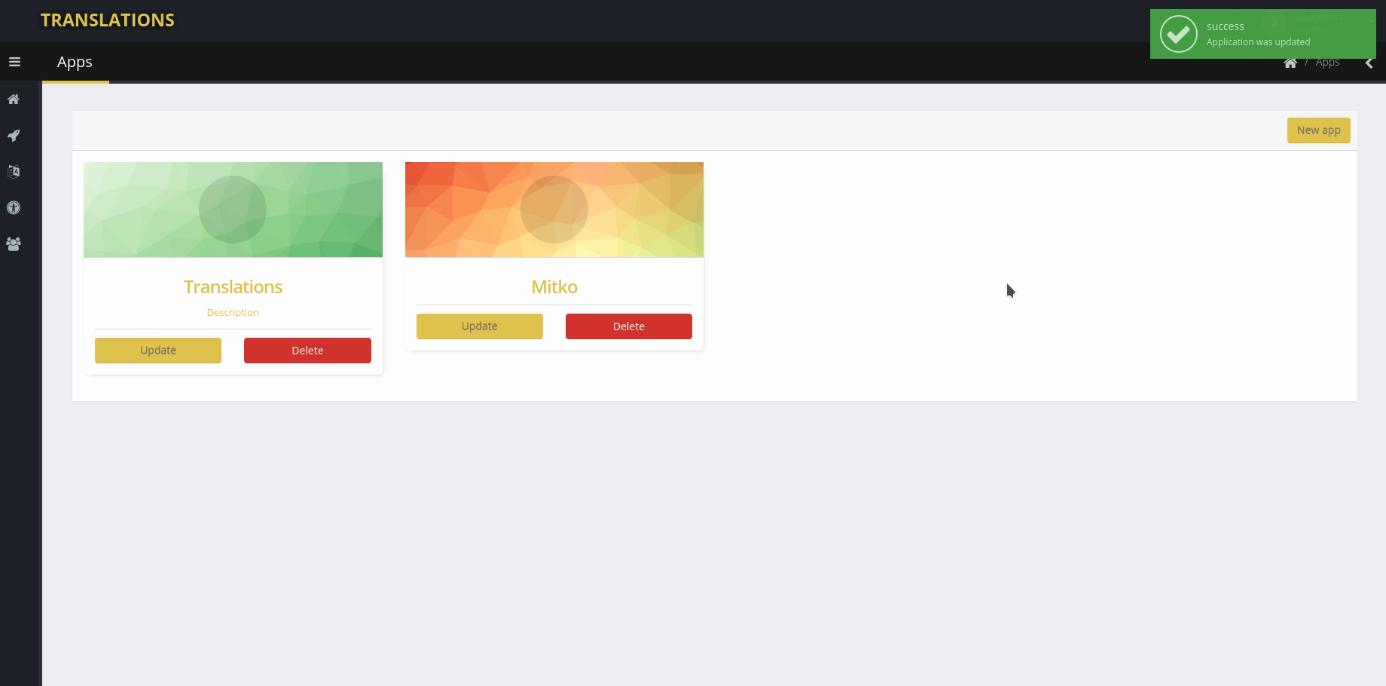
Всяко приложение се изчертава с различна графика и цветова гама спрямо името му както и два бутона за изтриване и за редактиране.

#### Редактиране на приложение



Фигура 15. Редактиране на приложение

При редактиране на приложението се визуалзира кутийка с две текстови полета име и описание, както и два бутона за успешно записване и за отказване.



Фигура 16. Редактирано приложение

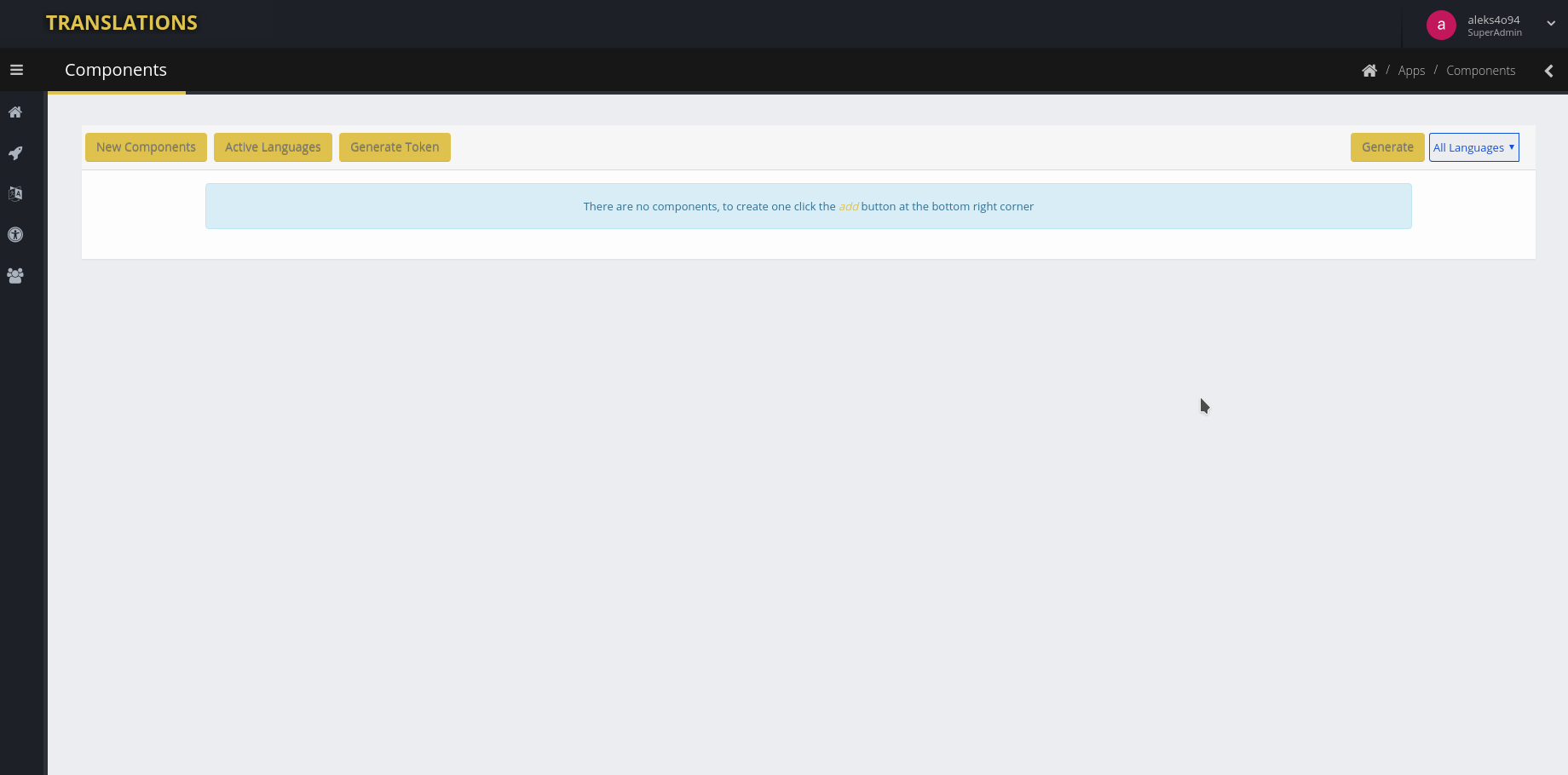
След успешно редактиране, в горения десен ъгъл се вижда съобщение, че приложението е успешно редактирано.

В модула за приложения се визуализират приложенията, само ако потребителят има достъп до тях. При създаване на приложение, потребителят става и собственик на приложението и само той има правото да го редактира или изтрие, докато не даде права на друг потребител.

Всеки потребител има достъп до този модул, но за създаване и редактиране на приложения, достъпът е ограничен до суперадмин и админ.

### Компоненти

Компонент се нарича дадена страница на приложението, за пример ще се вземе началната страница.



Фигура 17. Компоненти

В тази страница се визуализират четири бутона:

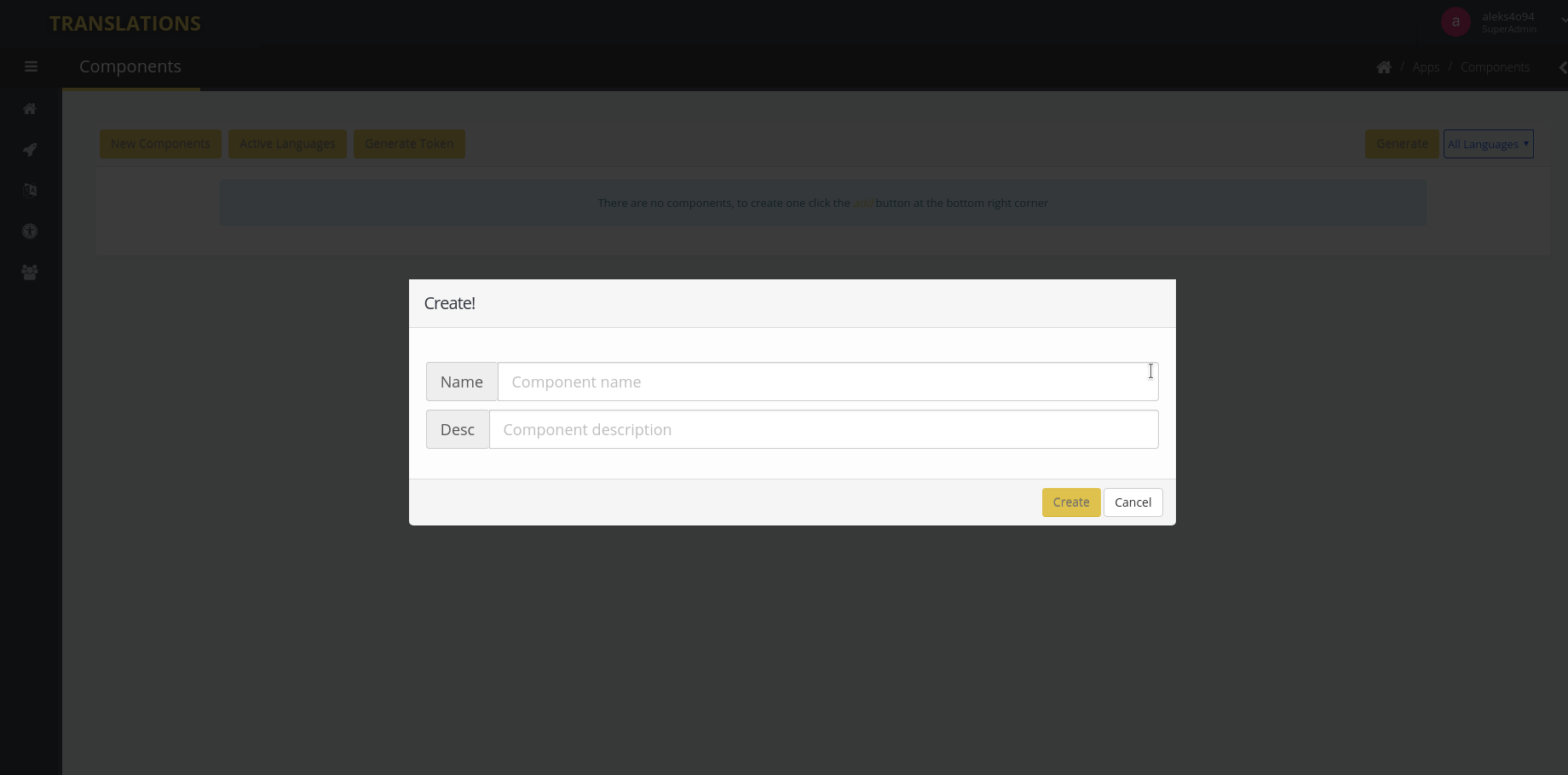
* Нов компонент.
* Активни езици.
* Генериране на подпис.
* Генериране.

Падащо меню и кутийка за изчертаване на самите компоненти.

Има съобщение, което оповестява, че няма създадени компоненти. Към текста има и линк към създаването на нов компонент.

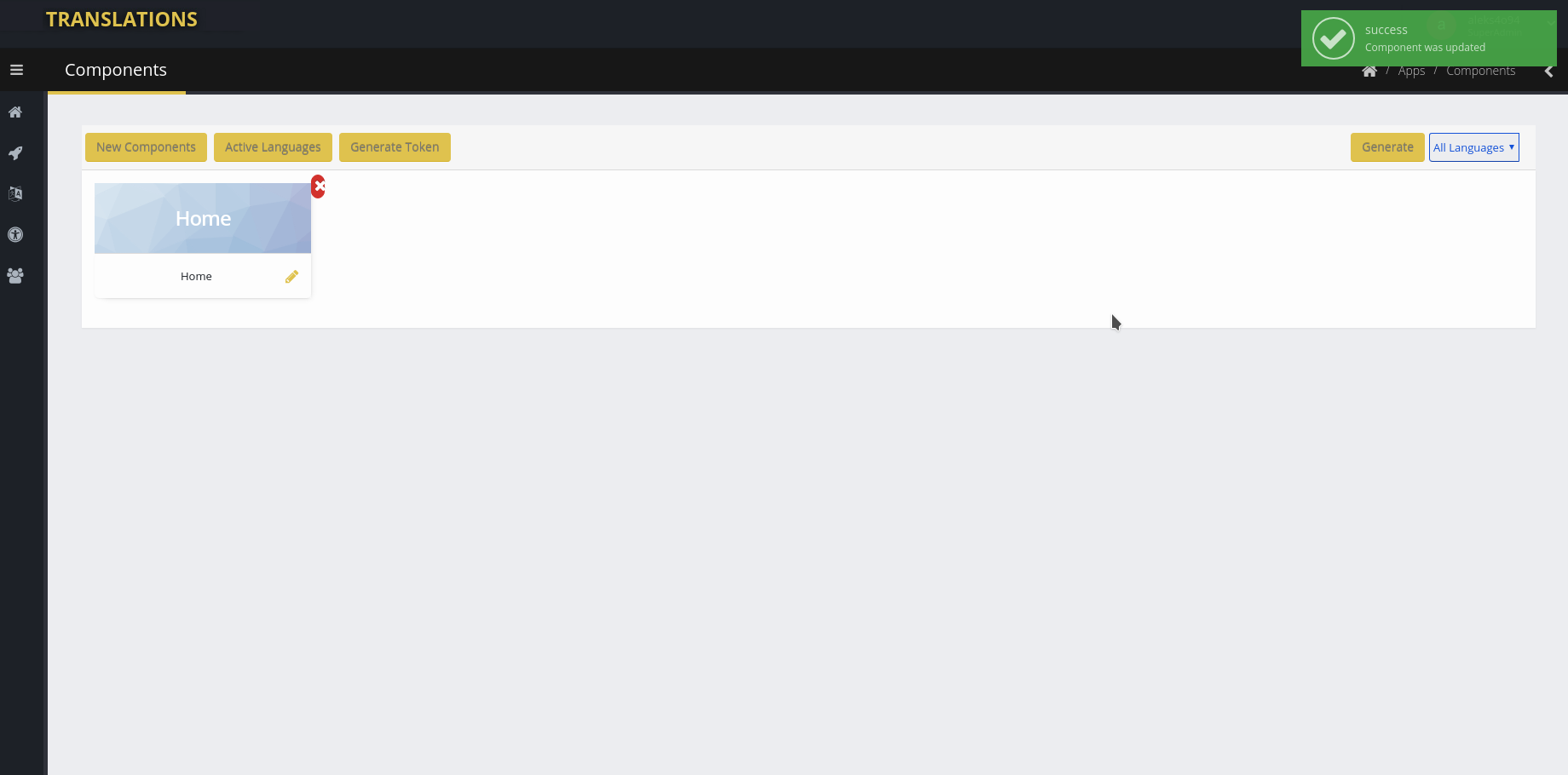
Достъп до компонентите имат всички потребители, но за създаване и редактиране, имат достъп само: суперадмин, админ и програмист.

#### Създаване на Компонент



Фигура 18. Създаване на компонент

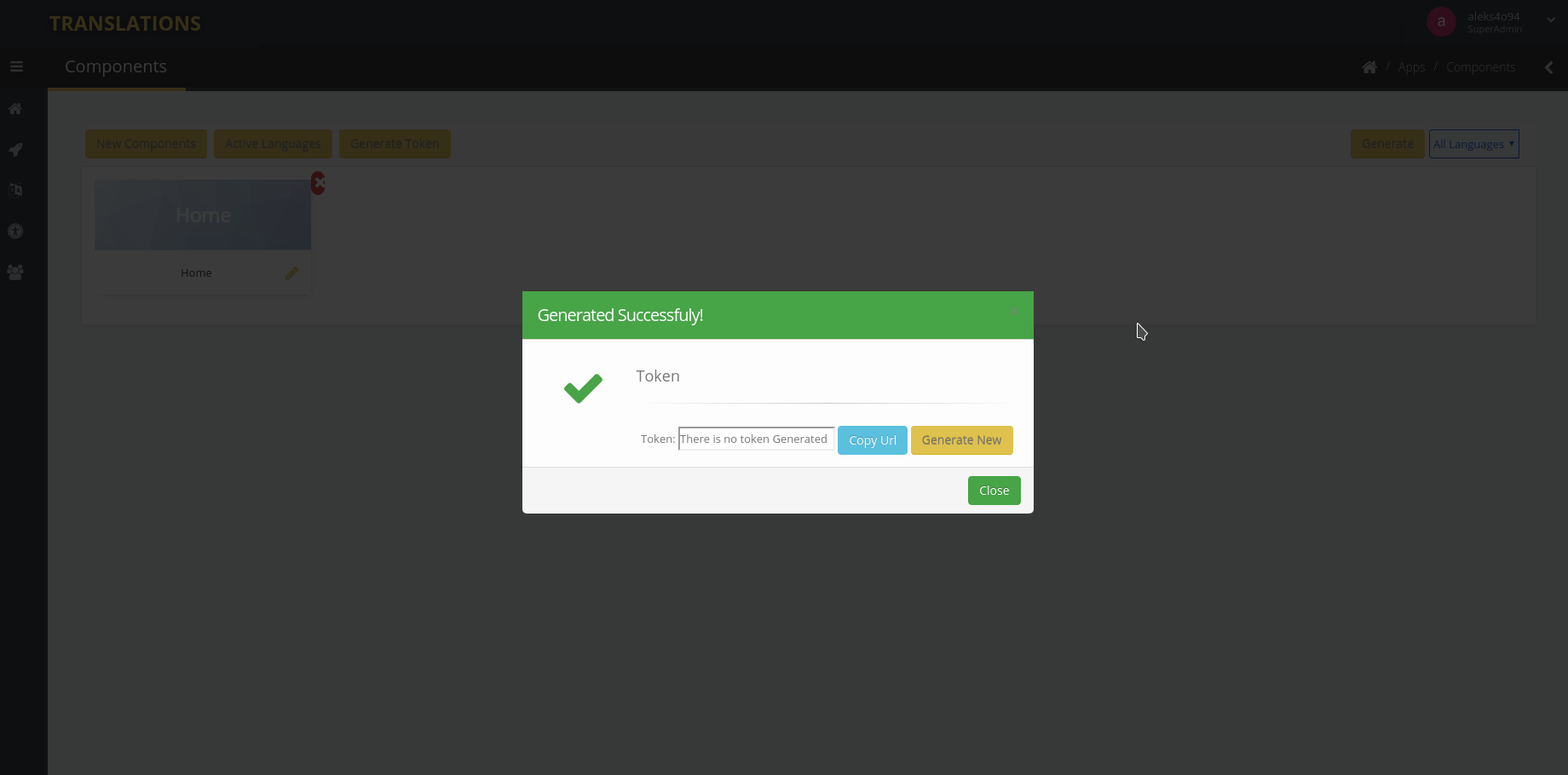
При натискане на бутона за създаване на нов компонент или линка се визуалзира кутийка с две текстови полета име и описание както и два бутона за успешно записване и за отказване. При успешно създаване се визуализира стикер със съобщение.



Фигура 19. Създаден компонент

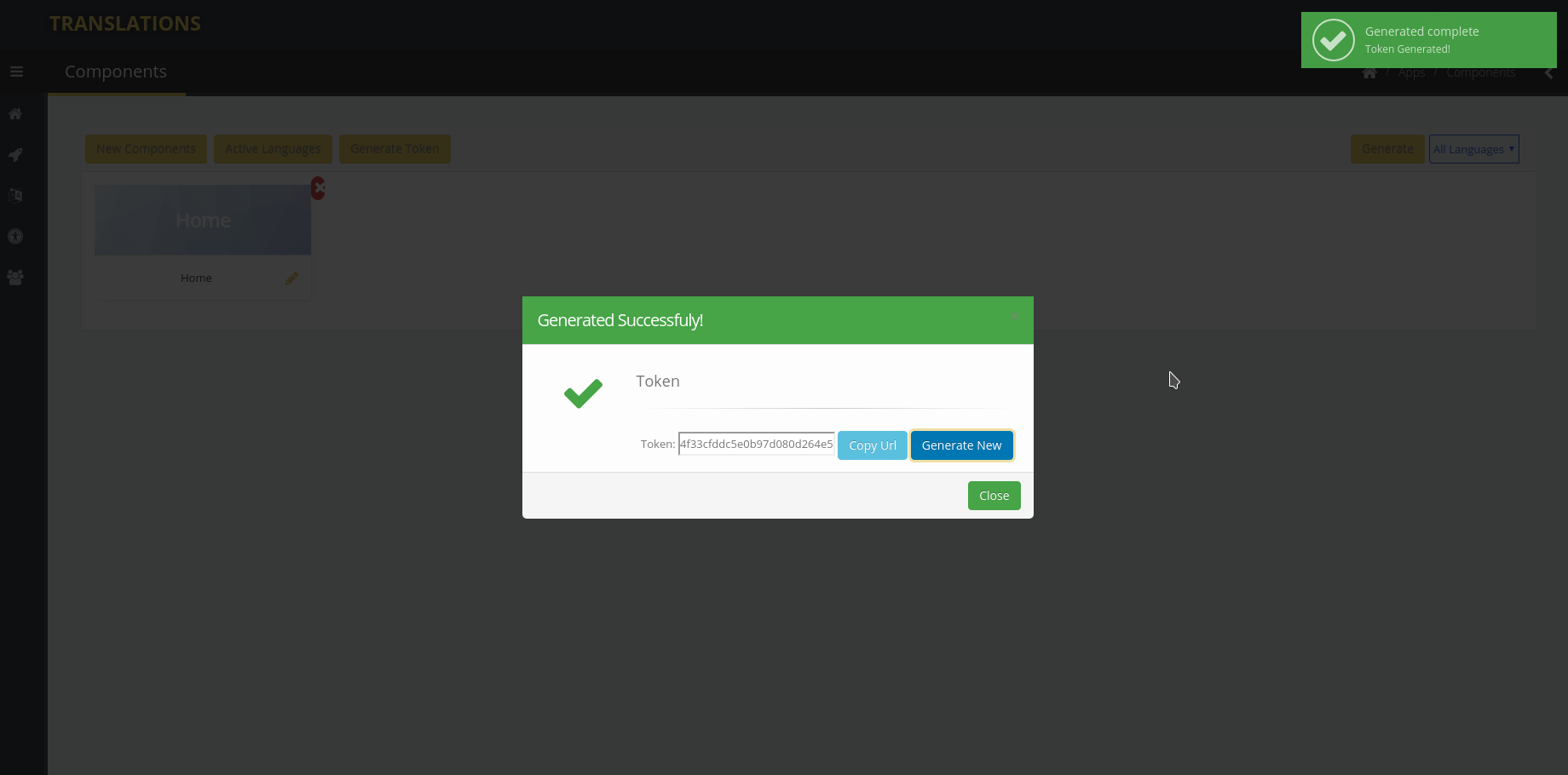
### Подпис-идентификация

При натискане на бутона за създаване на подпис се визуалзира кутийка с текстово поле и три бутона, съотвено за копиране на текста в текстовото поле, за генериране на нов подпис и за затваряне на кутийката. В текстовото поле има съобщение, което ни показва, че нямаме генериран подпис за момента.



Фигура 20.Генериране на подпис

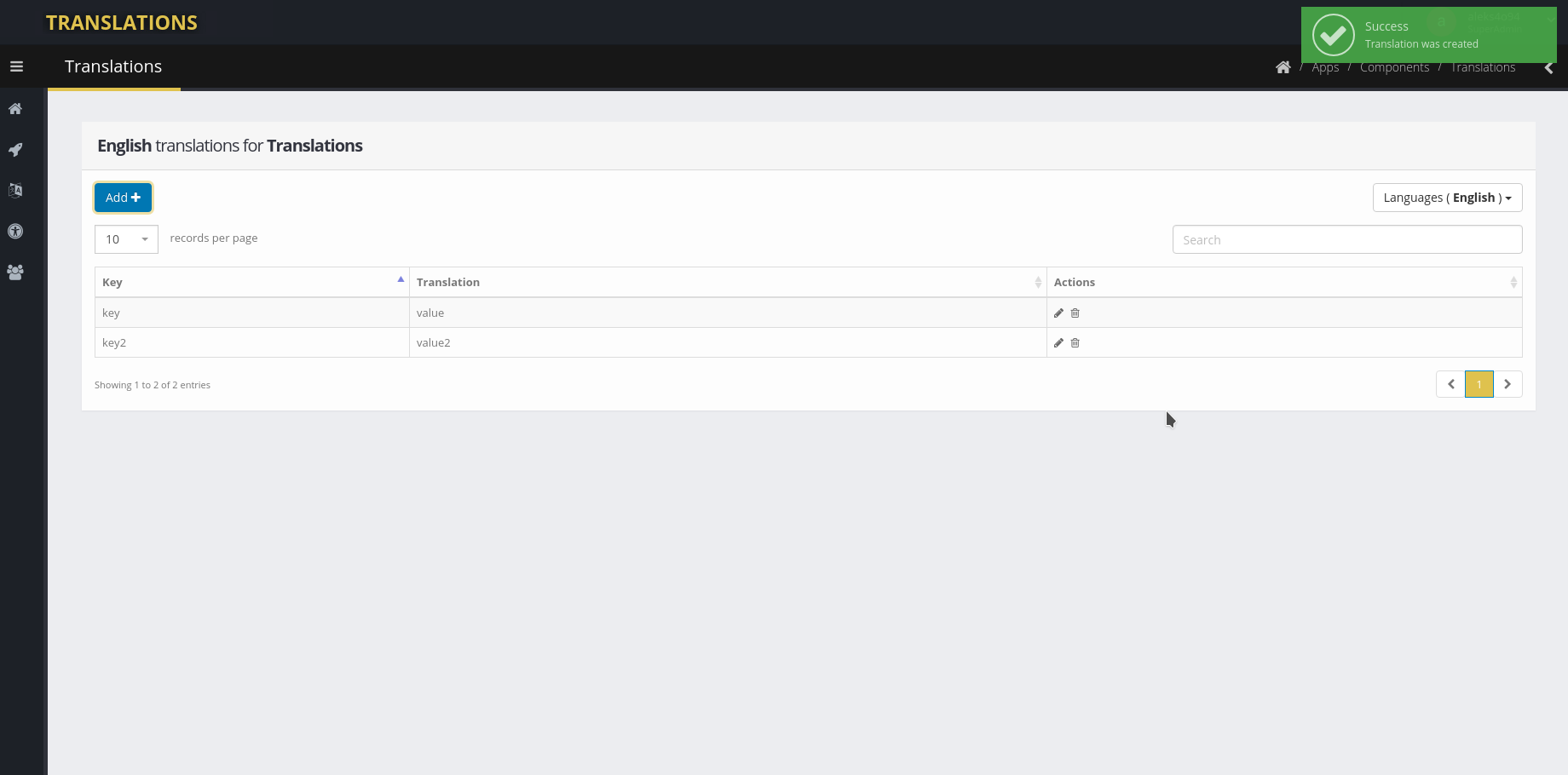
#### Създаване на Подпис



Фигура 21.Генериран подпис

При натискане на бутона за генериране на нов подпис, се създава уникален ключ, който ще служи за достъп до api, както и се вижда стикер със съобщение за успешно генериране.

### Преводи

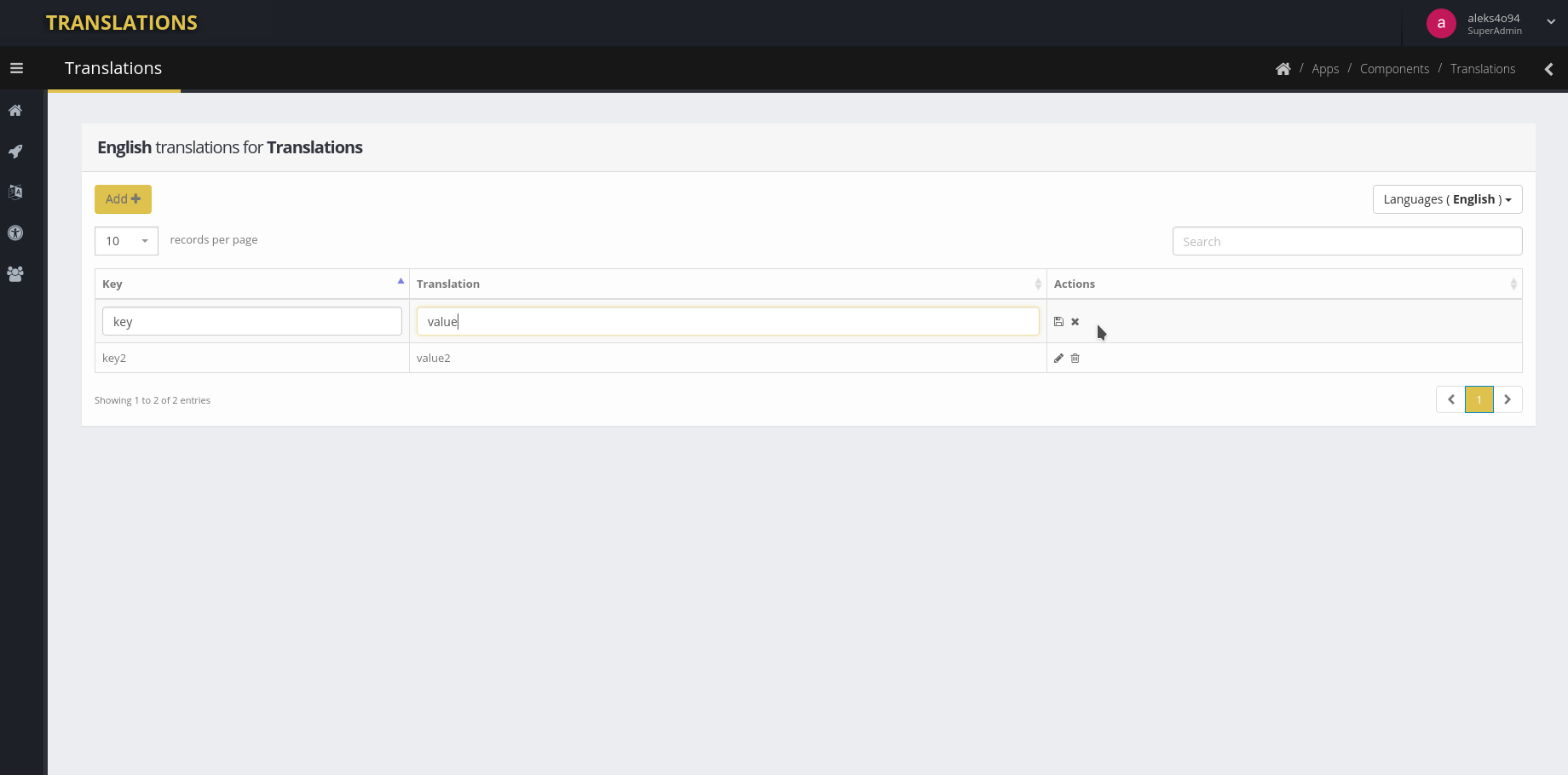


Фигура 22. Създаден превод

В страницата за преводи се визуализира кутийка с бутон за добавяне на нов ключ, два падащи бутона за избиране на езика и за броя редове на таблицата.

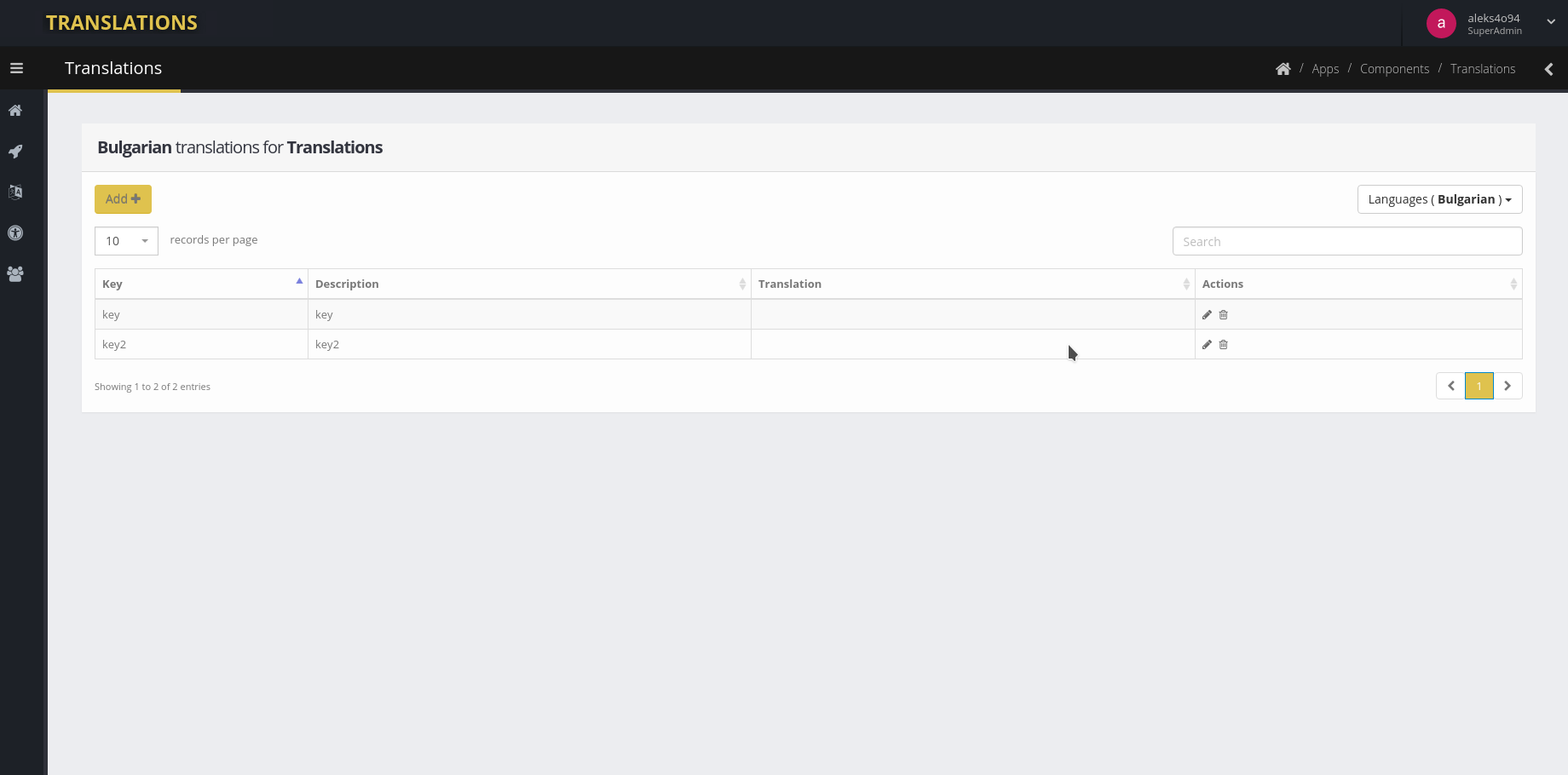
Всеки ред от таблицата има три колони. Първата колона е за ключа. Той се въвежда от програмиста и се използва директно в кода. Всеки ключ сочи към неговите преводи на различни езици.

#### Редактиране на преводи



Фигура 23. Редактиране на превод

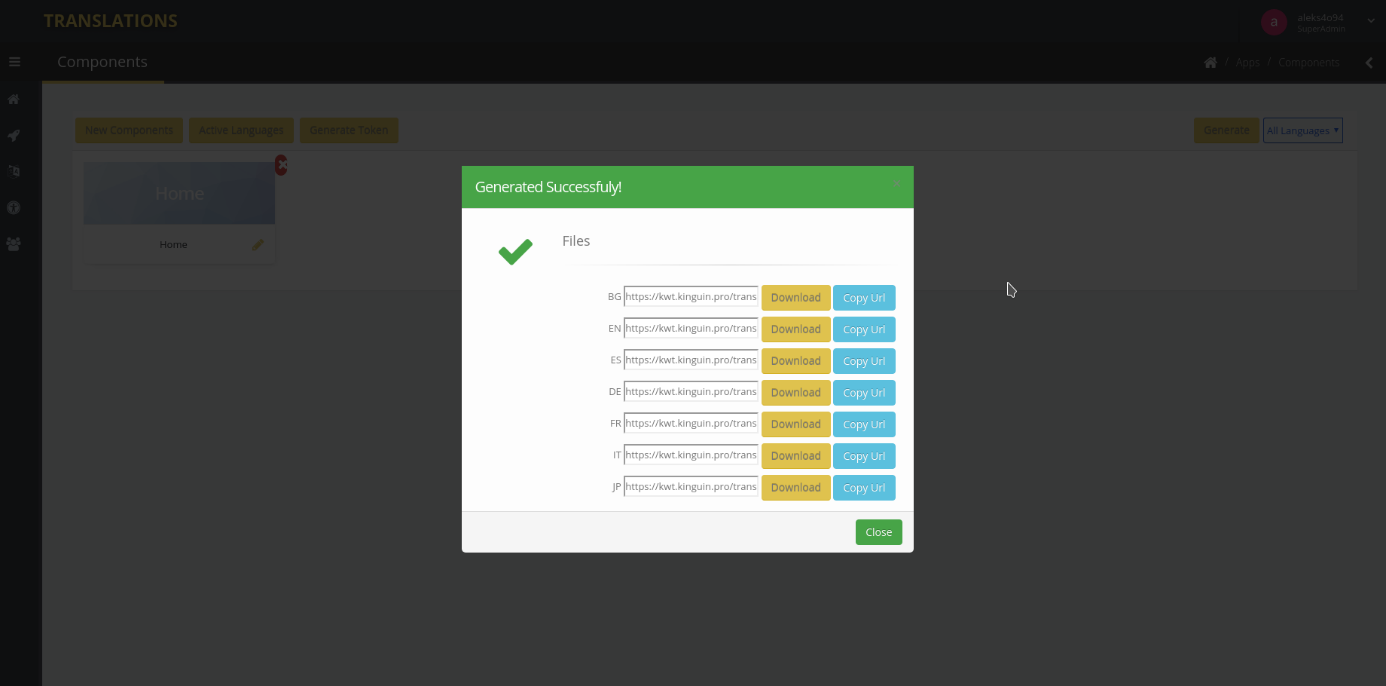
#### Преводи на чужди езици



Фигура 24. Превод на чужди езици

При смяна на езика се показва четвърта колона, която служи за описание на превода.

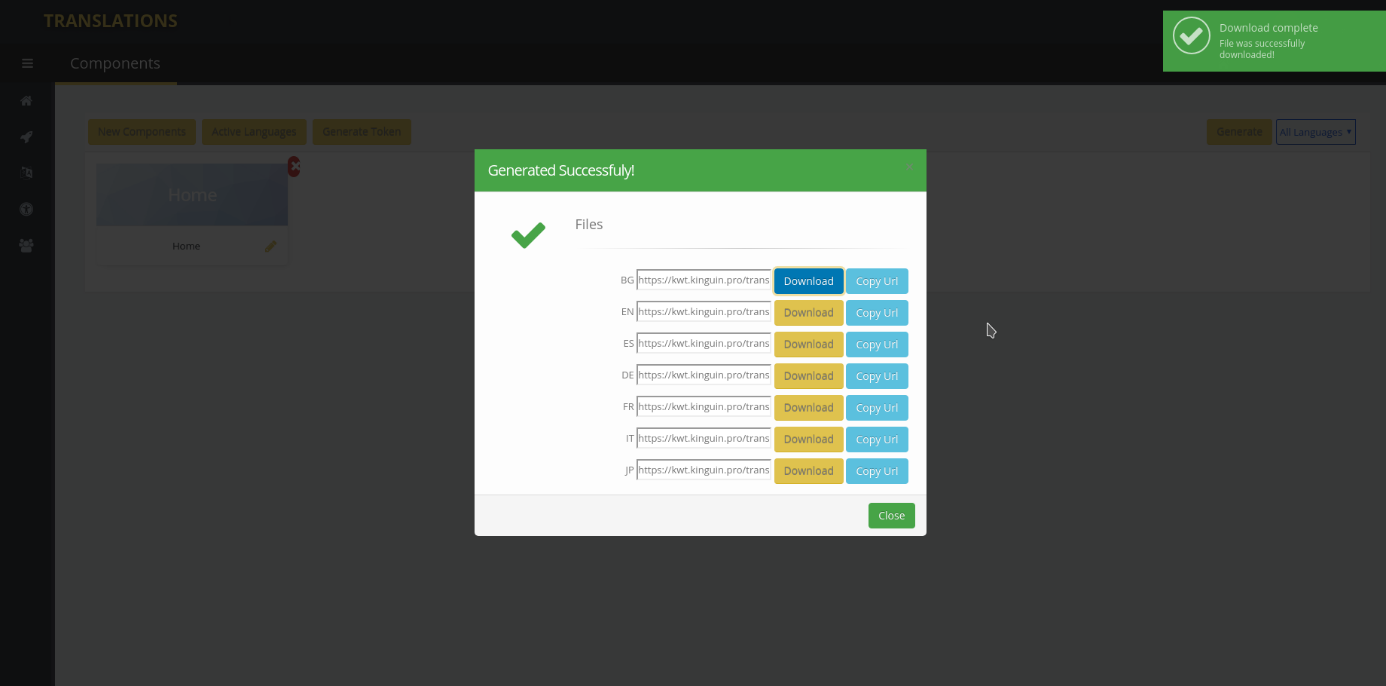
### Генериране на файлове



Фигура 25. Генериране на файлове

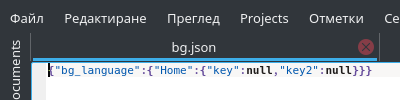
При натискане на бутон за генериране, се визуализира кутийка с текстово поле, бутон за сваляне и бутон за копиране на линка от текстовото поле. Редовете зависят от това, колко са активните езици за самото приложение. При избиране от падащият бутон в близост до бутона за генериране, може да се избере само един език и само него да генерира.

#### Сваляне на файлове



Фигура 26. Сваляне на файлове

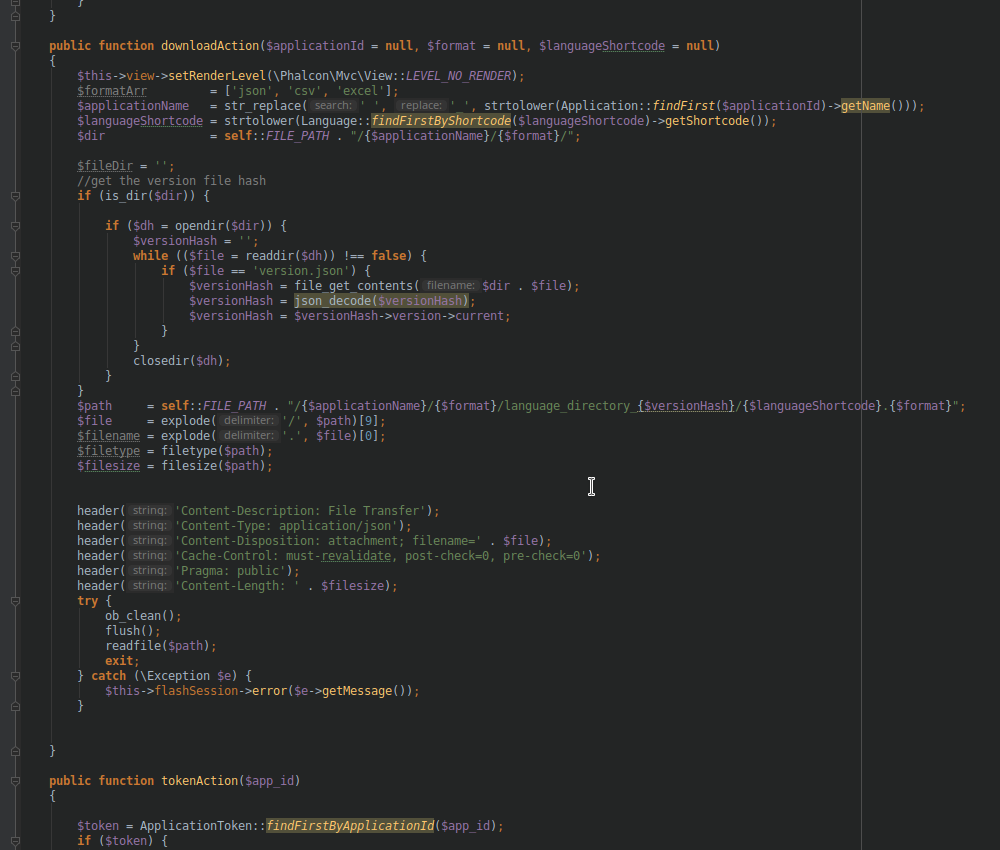
При натискане на бутона за сваляне, се генерира файл с разширение .json, който се записва на файловата система както и на google s3bucket с цел по-добра защита на данните.



Фигура 27. Изход от генерирането

За момента системата предлага само запис на файлове с разширение .json, но в бъдеще се очакват:

* .csv
* excel



Фигура 28. Код за свалянето на файл

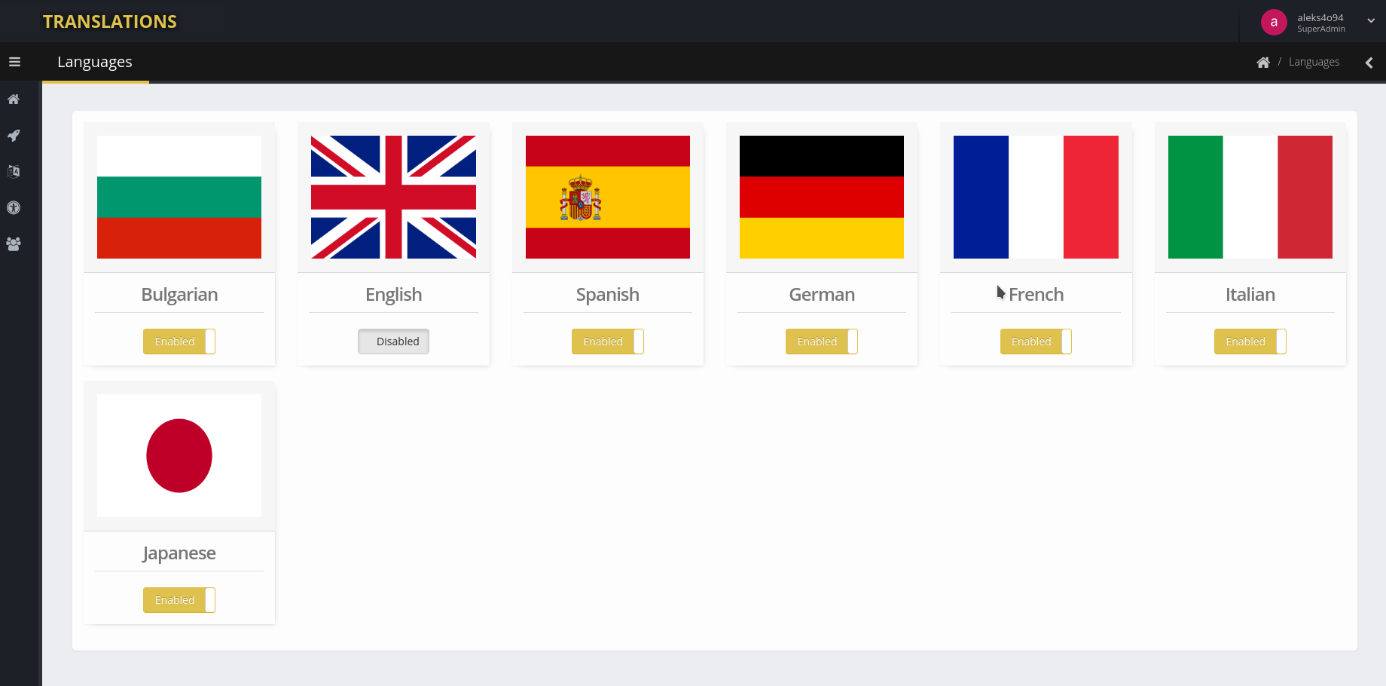
#### Копиране на пътя към файловете

### C:\Users\admin\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot_20180905_101hg628.png

Фигура 29. Копиране на пътя

При натискане на бутона копиране се копира информацията от текстовото поле. Линкът представлява директен достъп до файла, който може да се отвори през браузъра.

### Езици



Фигура 30. Активни езици

В този модул може да се променят езиците, които са активни към самото приложение.

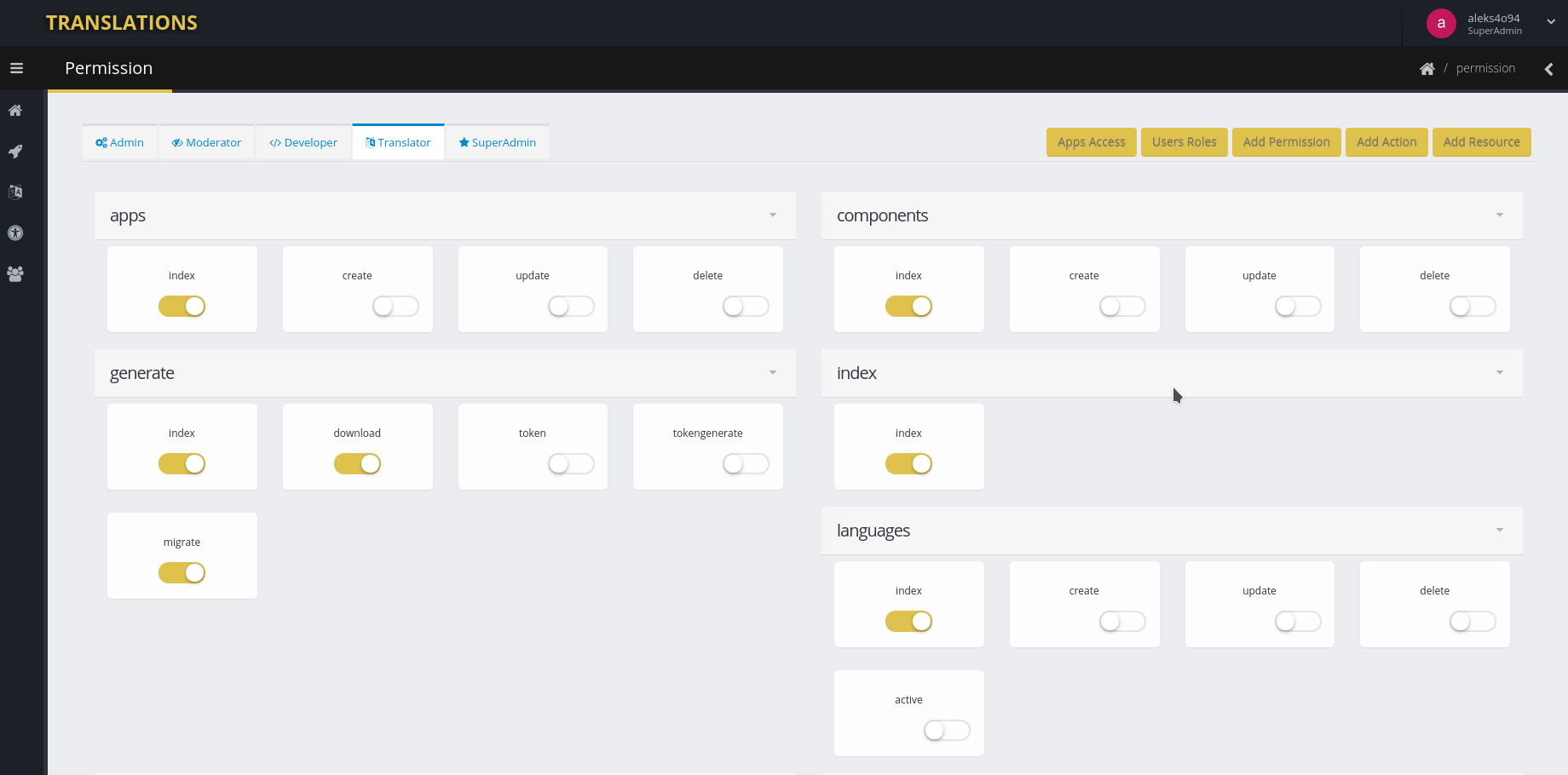
Достъп до този модул имат само суперадмините и админите.

При създаване на нов език се попълват полетата:

* Име
* Съкращение
* Флаг-снимка

След успешно създаване на езика, той автоматично се слага с отметка за активен език.

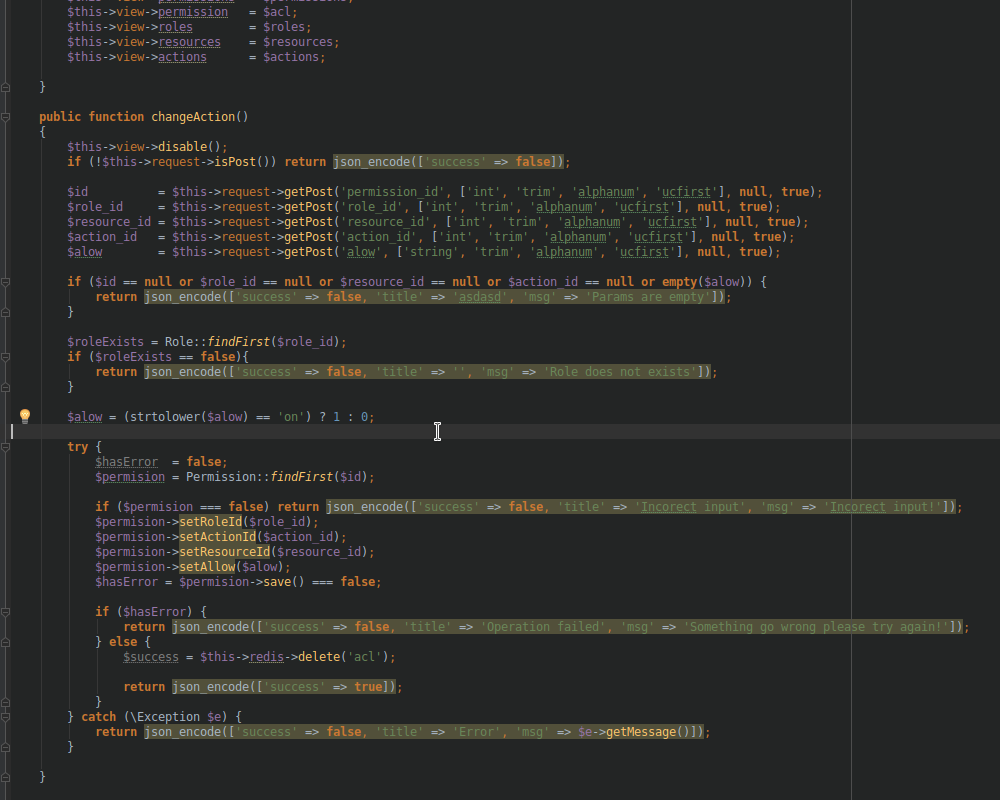
### Контрол на достъпа



Фигура 31. Регулиране на достъпа

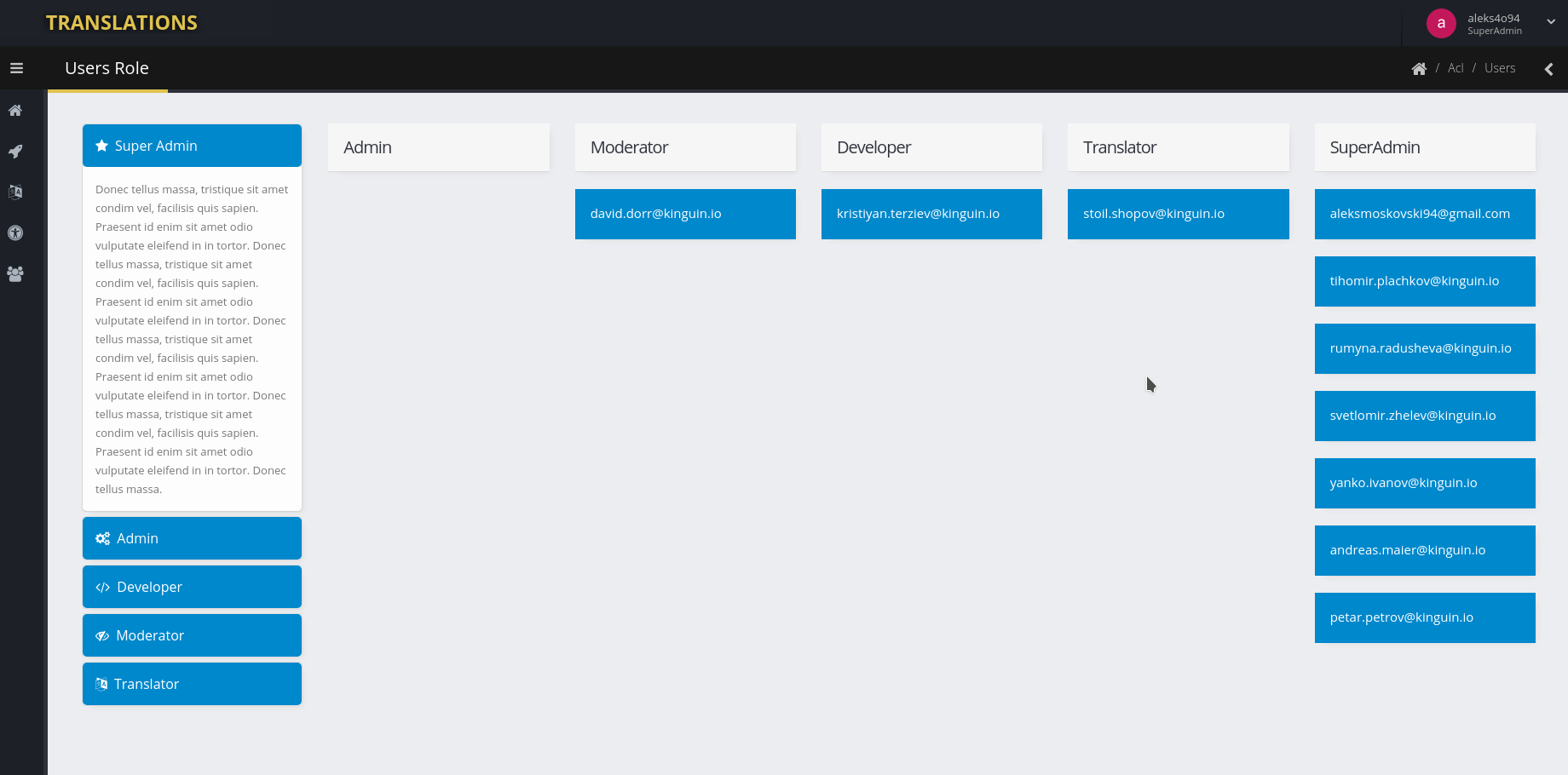
В този модул се определя всяка роля до какво има достъп. Всяка страница от приложението има различни процеси, всеки от които с бутонче за включено-изключено.

В горната част на модула се визуализират всички роли, като при кликане се зареждат различни нива на достъп към всяка роля. От дясната страна има пет бутона.



Фигура 32. Код за промяна на контрола

### Управление на роли



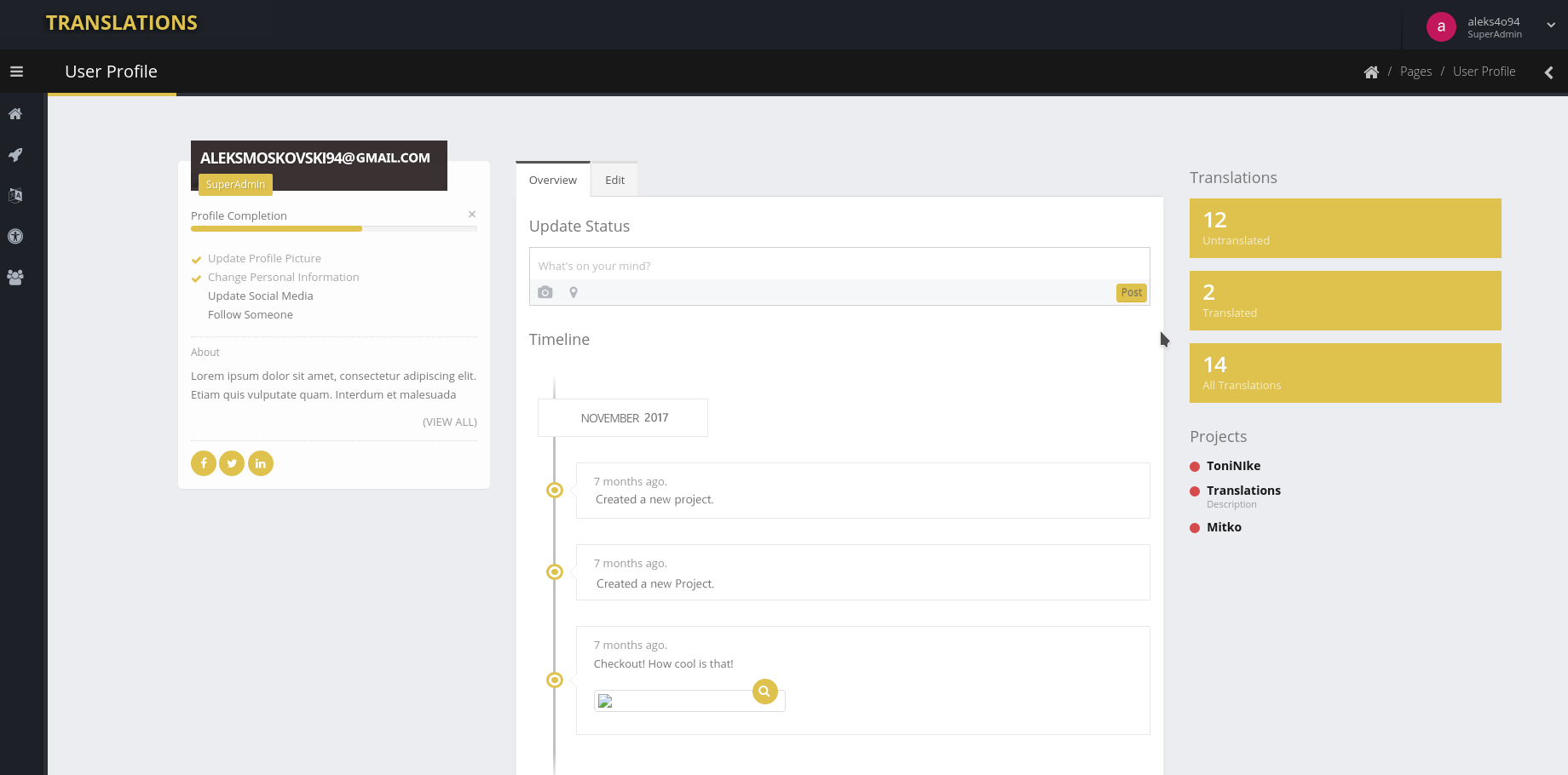
Фигура 33. Управление на роли

В този модул се определя всеки потребител на системата към коя роля ще е спрямо колонката в която стои. При добавяне на нова роля ще се визуализира и нова кутия, към която може да се разпределят потребителите. Смяната на ролята става лесно с придърпване на кутията с имейла на потребителя към колонката с роля, която желае. От лявата страна има кутии с обяснения за всяка роля. Те се отварят и се затварят лесно -тип акордеон.

До този модул имат достъп само суперадмините.

### Профил

В тази страница се визуализира информация за потребителя.



Фигура 34. Профил

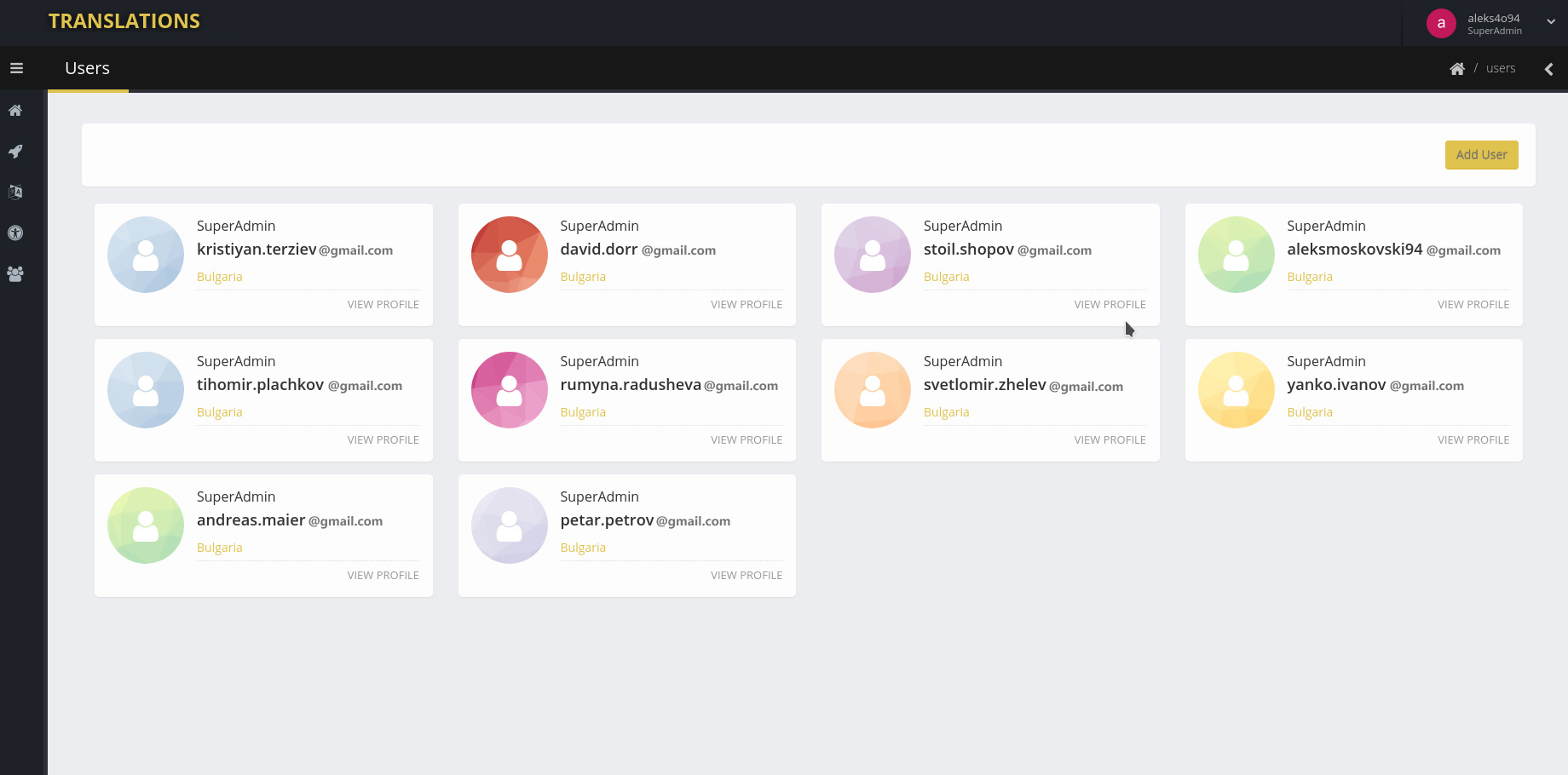
Към фигура 34 може да се проследи колко проекта е създал всеки потребител, всеки проект колко задачи има.



Фигура 35. Заявка за потребител

Заявката се изпълнява, след проверка дали компонентът е заредил. Посредством ajax заявка се взима информацията от сървъра за този потребител.

### Потребители



Фигура 36. Потребители

В този модул се визуализират кутийки с всички потребители. Всяка кутийка има:

* Роля
* Имейл
* Държава
* Линк към профила

Най-отгоре има бутон за добавяне на нови потребители. При натискане на бутона се визуализира текстово поле за добавяне на имейла, падащо меню за избиране на ролята и бутон за запазване или отказване.

Глава 4. Ръководство за програмиста



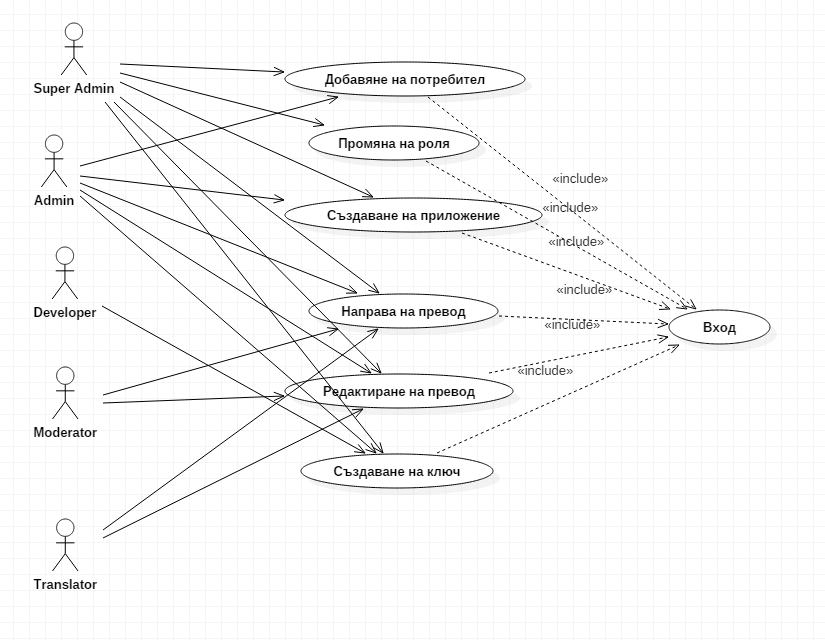
## Функционална схема на системата

**UML** – унифициран език за моделиране (*Unified Modeling Language*), e графичен език за визуализиране, специфициране, конструиране и документиране на елементите на една софтуерна система. Диаграмата на класове се използва за описване на статичната структура на дадена система. Стандартен език, с набор от знаци за означаване, предназначен за визуализиране, специфициране, конструиране и документиране на софтуерен дизайн. Използва концепции от моделирането на данни (Entity Relationship Diagrams), бизнес моделиране (work flow), обектно моделиране, компонентно моделиране.

Чрез нея се описват класове, обекти, пакети и връзките между тях. Използвани са Use Case диаграма за илюстриране правата на различните потребители, както и Sequence диаграма за това как един регистриран потребител работи в системата и какви заявки прави самата система. За изрисуването на диаграмите е използвана StarUML.

### Use case

Техника за определяне нa функционалните изисквания на една система. Те описват типичните взаимодействия между потребителите и системата, предоставят описание на начина, по който тя се използва. В тази диаграма са представени правата на потребителите и администратора.

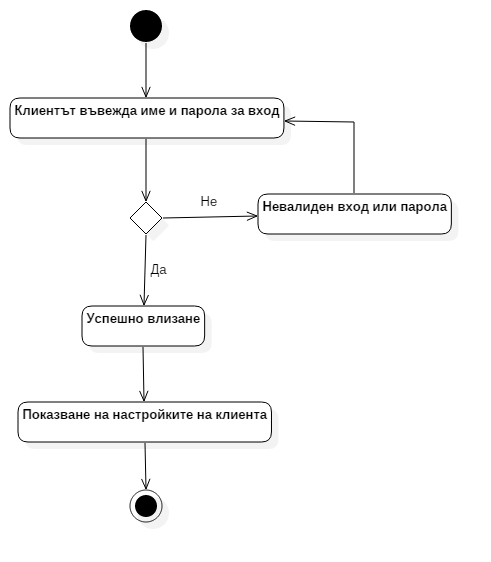


Фигура 37. Use Case диаграма

### Activity диаграма

Activity диаграми (диаграми на дейността) са техника за описване на процедурната логика, бизнес процесите и работния поток. Activity диаграмата е динамична диаграма, която показва активностите и събитията, които са причина обектите да се намират в определено състояние.

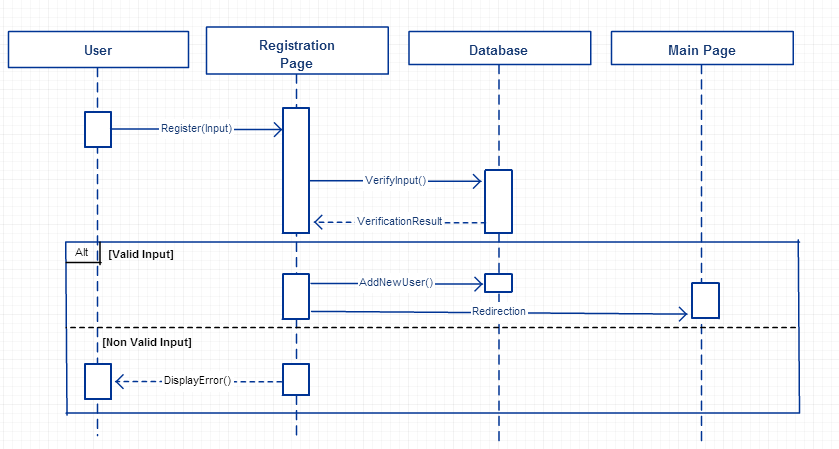
Този тип диаграми дават възможност на този, който извършва процеса, да избере реда, в който да се изпълняват нещата. С други думи диаграмата просто указва съществените правила за последователност, които трябва да се следват. Също така е от изключително важно значение, отразяването на нишки от процесите, които могат да се изпълняват паралелно. В тази диаграма е представен процесът на влизане в системата.



Фигура 38. Activity диаграма

### Sequence диаграма

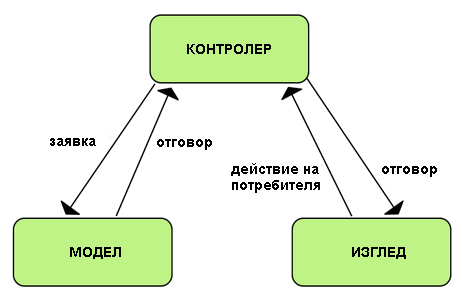
Диаграмите на взаимодействие описват начина, по който групи от обекти постигат съвместно някакво поведение.Случаите на употреба обхващат взаимодействието между потребителите и системата. Диаграмата на взаимодействието описва поведението на един единствен случай на употреба, показвайки сътрудничеството на обектите в системата. Тази диаграма показва обектите в системата и съобщенията, които са разменят между тях. В Sequence диаграмата е представен процесът на регистрация на нов потребител.



Фигура 39. Sequence диаграма

## Архитектура на системата

**MVC (Модел – Изглед - Контролер)** е шаблон за дизайн на софтуерна архитектура, използван при реализацията на потребителски интерфейси, който представя софтуерното приложение като три взаимо-свързани части, разделяйки вътрешното представяне на информацията от информацията, представена на потребителя. Приложенията, реализирани чрез прилагането на MVC са по-лесни разработка и поддръжка, особено от екип - дизайнери и разработчици. Чрез него се разделя бизнес логиката от графичният интерфейс и данните в дадено приложение. Той е разделен на три части – Модел-Изглед-Контролер. Моделът е независим и може да бъде планиран и осъществен независимо от другите части на системата. Друго предимство при този шаблон е че контролерът и изгледът могат да бъдат променени, без да се налага промяна в модела. MVC е най-често срещаният архитектурен шаблон използван в известните PHP рамки.



Фигура 40. Обща схема на MVC

**Моделът (Model)** съдържа основна част от бизнес логиката на приложението и изгражда функционалностите на отделните модули. Той се състои от данни, бизнес правила, логика, и функции. Моделът комуникира с базата данни и с програмните интерфейси на трети страни. Той уведомява своите асоциирани изгледи и контролери, когато е налице промяна в състоянието му. Ядрото на приложението, предопределено от областта, за която се разработва, обикновено това са данните от реалния свят, които сме моделирали и над които искаме да работим – да въвеждаме, променяме, визуализираме и т.н. Трябва да се направи разлика между реалния обкръжаващ ни свят и въображаемия абстрактен моделен свят, който е продукт на нашият разум, който ние възприемаме във вид на твърдения, формули, математическа символика , схеми и други помощни средства.

**Изгледът (View)** се грижи за представянето на данните, създадени от модела, към потребителя. Изгледът е тази част от изходния код на приложението, отговорна за показването на данните от модела. Зависи от това какъв графичен интерфейс се прави и каква платформа се използва. В него са включени презентационни и графични шаблони и стилове за представяне на данните в различни типове страници, и почти не се съдържа бизнес логика. Презентационният слой има и допълнителна много важна функция, която не се вписва стриктно в методологията на трислойната архитектура. При отделни модули на приложението, презентационният слой съдържа програмен код, който осъществява връзка с програмен интерфейс на трета страна и изпраща и получава данни от нея. Изгледът няма право да генерира данни и да реализира логика освен тази която е необходима за презентиране на информацията.

**Контролерът (Controler)** служи за приемане и обработка на входящите заявки към уеб приложението. Той комуникира както с изгледа, така и с модела.Контролерът взима данните от модела или извиква допълнителни методи върху модела, предварително обработва данните, и чак след това ги дава на изгледа. Софтуерът за прогнозиране на успеха използва техника, наречена една входна точка (single entry point), в резултат на което, всички заявки, изпращани от уеб браузъра на потребителя, се предават от уеб сървъра към един и същи програмен файл на приложението. Този файл съдържа логиката за разпределяне на заявките вътре в приложението и, на базата на подадените параметри в заявката, решава кой метод за обработка на данните да стартира. Архитектурата с една входна точка се счита за добра практика, тъй като всички файлове, съдържащи бизнес логиката на приложението, остават скрити от външните потребители. За правилната реализация на този похват се правят и някои 105 специфични настройки на уеб сървъра. Създават се правила за пренаписване на заявките (rewrite rules) и се добавят декларации и файлове за ограничаване на достъпа до определени директории на уеб сървъра.

**Предимства на MVC архитектурата:**

* Изгледът не зависи от модела. Т.е дизайнерът може лесно да модифицира потребителския интерфейс (HTML страница), без да му се налага да навлиза в бизнес логиката на системата.
* Моделът е независим и може да бъде планиран и разработван без да пречи на другите части на системата.
* След като сме създали модела, можем да променяме контролерите и изгледите, без това да окаже влияние върху него.
* Всеки модел може да използва различен интерфейс.
* Архитектурата Service-Repository прави системата още по-гъвкава като разделя Модела на 2 слоя.

**Недостатъци**

* Нивото на абстракция на кода се повишава след използване на Service-Repository архитектурата, което може да затрудни разработчици, които нямат опит с подобен начин на структуриране на кода.

## Архитектура на базата данни

За приложения от такъв тип, съхранението на данните и бързия достъп до тях е приоритет. В системата има наличие на голяма по обем информация и заради това тя трябва да бъде правилно разположена в различни таблици. За целта се нуждаем от добре проектирана база от данни. Информацията в базите данни е подредена по определен начин, което прави възможна компютърната й обработка.

Например релационни бази данни подреждат въведената в тях информация под формата на таблици. Ще разгледаме отделните таблиците, които са използвани за съхранение на информацията в приложението. То използва 14 отделни таблици, за да съхранява информацията но главните са :

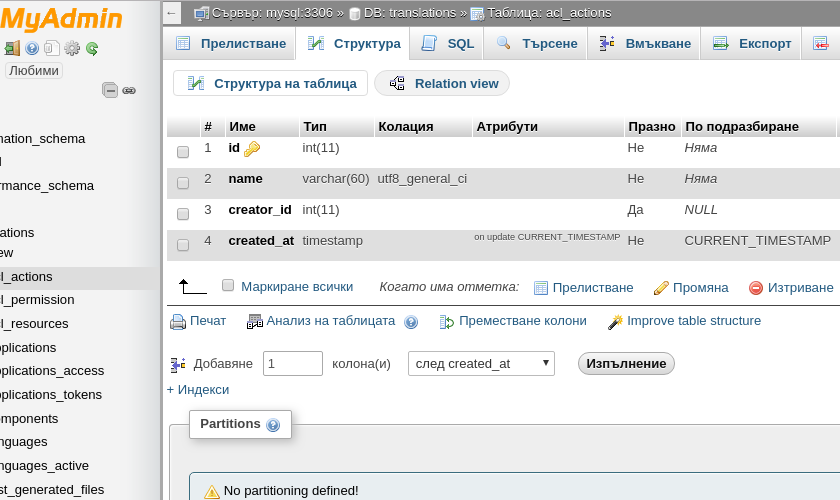


Фигура 41. База данни

Acl Actions

В таблицата с действия се запазват:

* Индетификационен номер.
* Име на действието.
* Индетификационен номер на потребителя, създал действието.

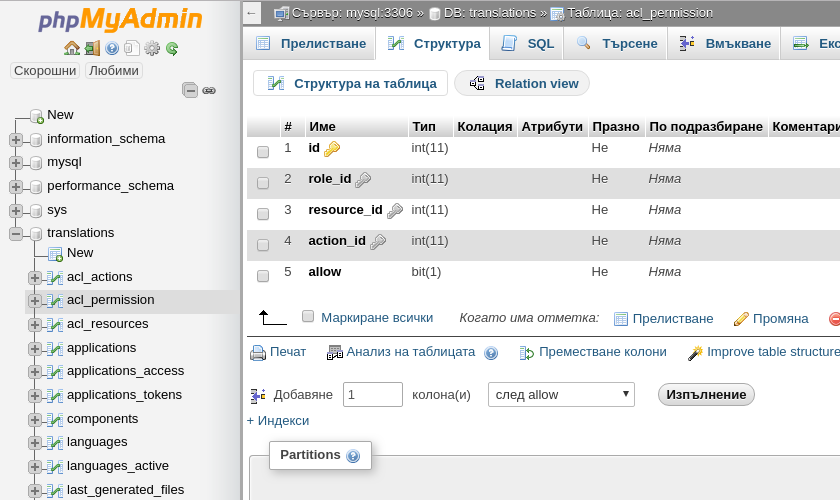


Фигура 42. Таблица действия

Acl Permission

В таблицата с позволения се запазва:

* Индетификационен номер.
* Индетификационен номер на ролята.
* Индетификационен номер на ресурса.
* Индетификационен номер на действието.
* Дали е позволено или не.

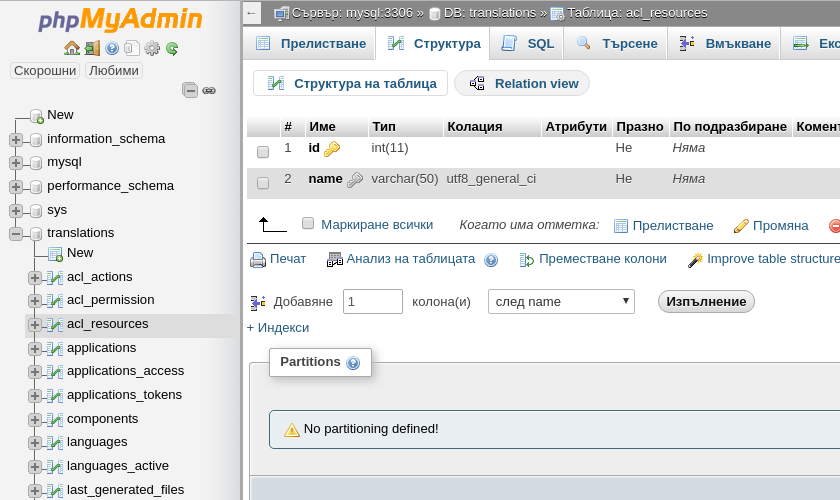


Фигура 43. Таблица позволения

Acl Resources

В таблицата с ресурси се запазва:

* Индетификационен номер.
* Име на ресурса.

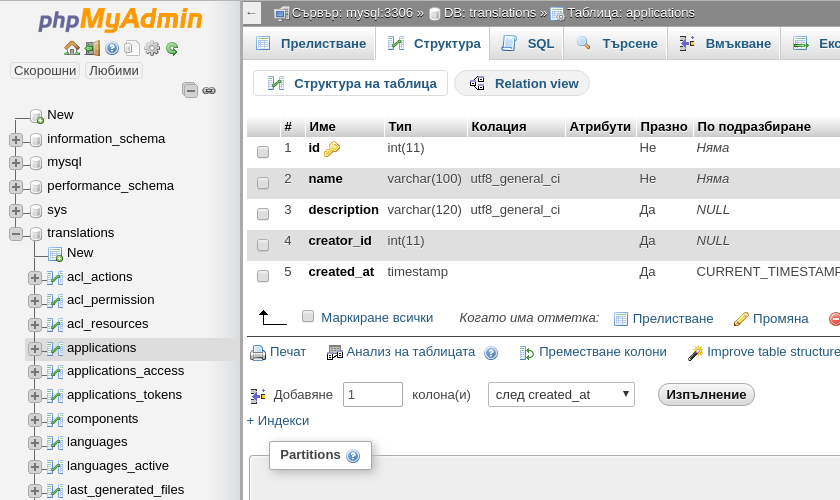


Фигура 44. Таблица ресурси

Applications

В таблицата с приложения се пазят:

* Индетификационен номер.
* Име на приложението.
* Описание на приложението.
* Индетификационен номер на потребителя, който го е създал.
* Дата на създаване.

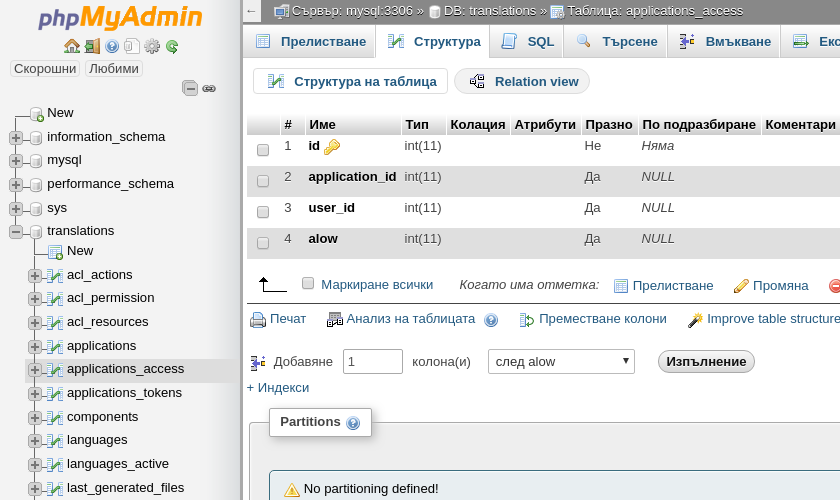


Фигура 45. Таблица приложения

Applications access

В таблицата с достъп до приложения се пазят:

* Индетификационен номер.
* Индетификационен номер на приложението.
* Индетификационен номер на потребителя.
* Дали е позволено или не.

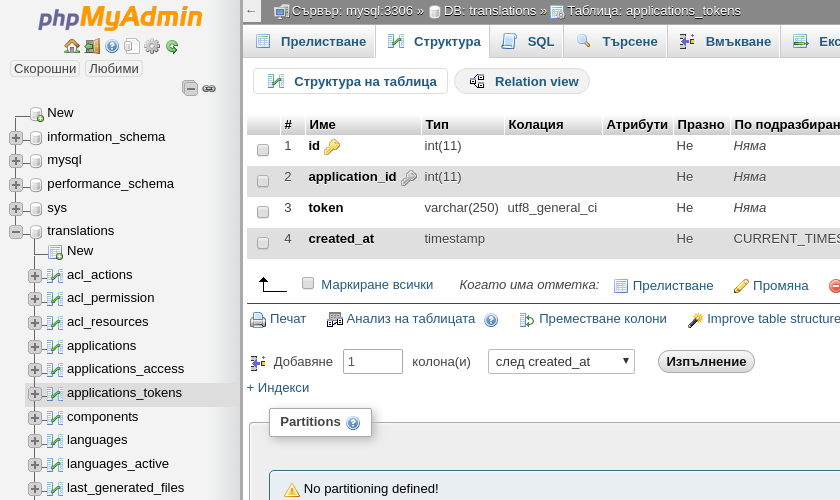


Фигура 46. Таблица достъп до приложения

Application token

В таблицата с уникален подпис на приложение се пазят:

* Индетификационен номер.
* Индетификационен номер на приложението.
* Подпис.
* Кога е създадена.

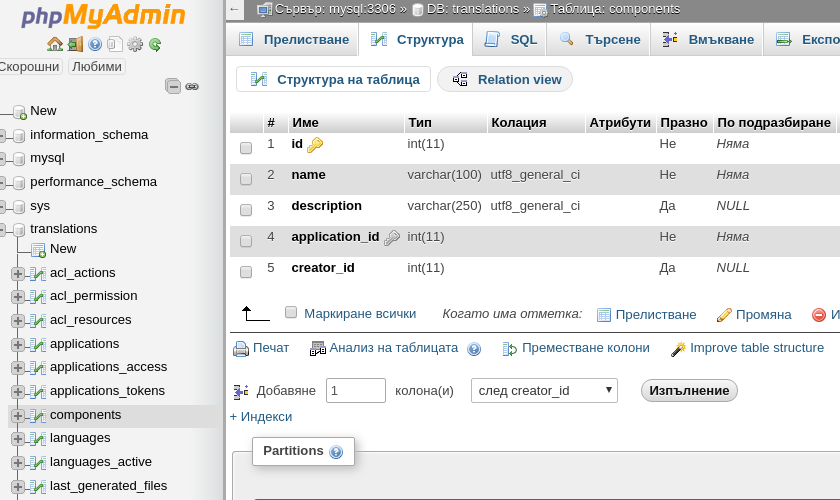


Фигура 47. Таблица подпис на приложение

Components

В таблицата с компоненти се пазят:

* Индетификационен номер.
* Име на компонента.
* Обяснение към компонента.
* Индетификационен номер на приложението.
* Индетификационен номер на потребителят, който го е създал.

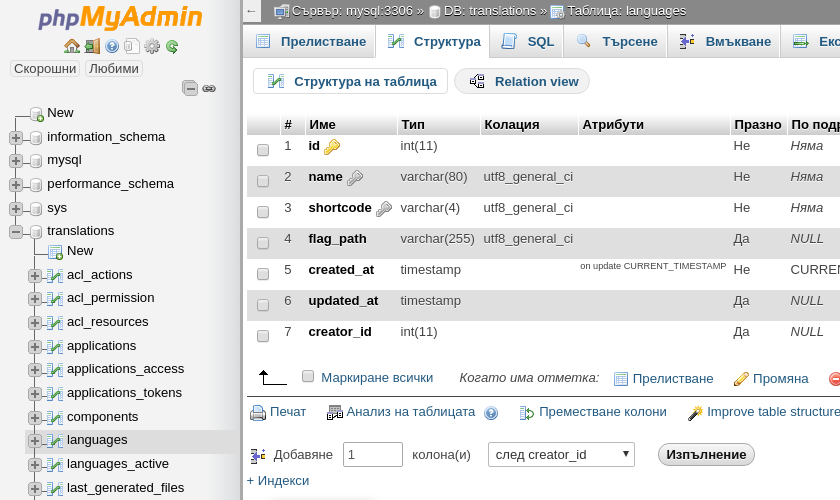


Фигура 48. Таблица компоненти

Languages

В таблицата с езици се пазят:

* Индетификационен номер.
* Име на езика.
* Съкращение на езика.
* Път към снимка на флага.
* Кога е създаден езика.
* Кога е редактиран езика.
* Индетификационен номер на създателя на езика.

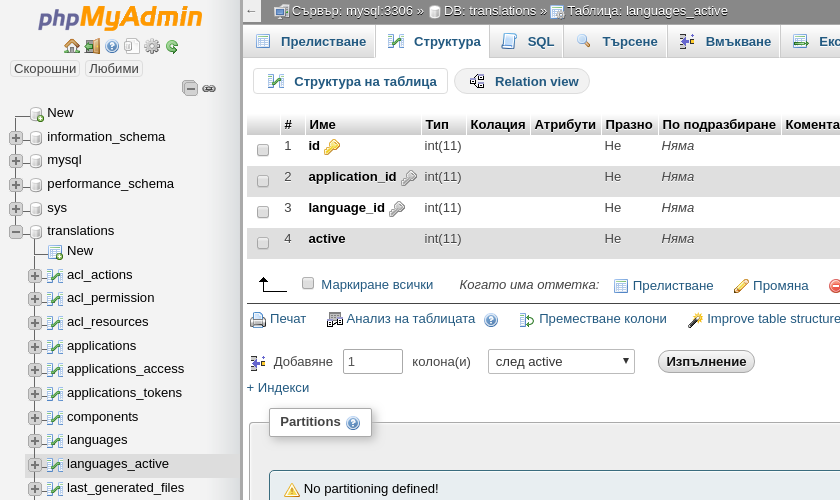


Фигура 49. Таблица езици

Languages active

В таблицата с активни езици се пазят:

* Индетификационен номер.
* Индетификационен номер на приложението.
* Индетификационен номер на езика.
* Дали е активно или не.

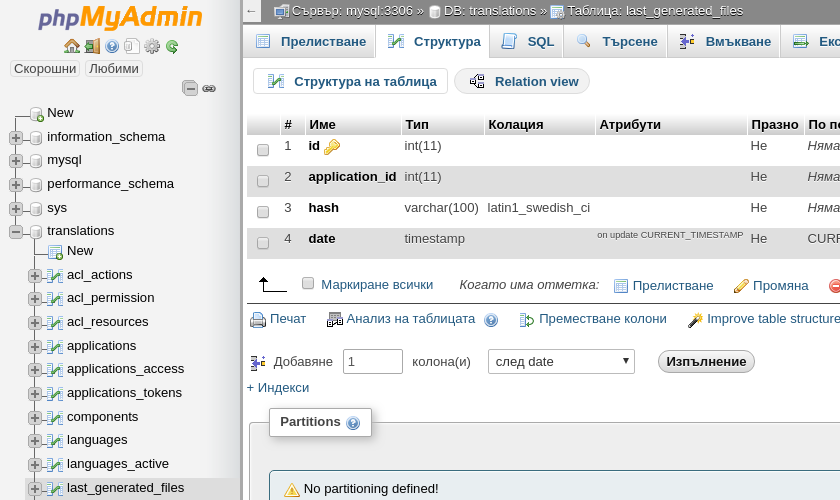


Фигура 50. Таблица активни езици

Last generated files

В таблицата с последно генериране файлове се пазят:

* Индетификационен номер.
* Индетификационен номер на приложението.
* Уникален номер.
* Дата на създаване

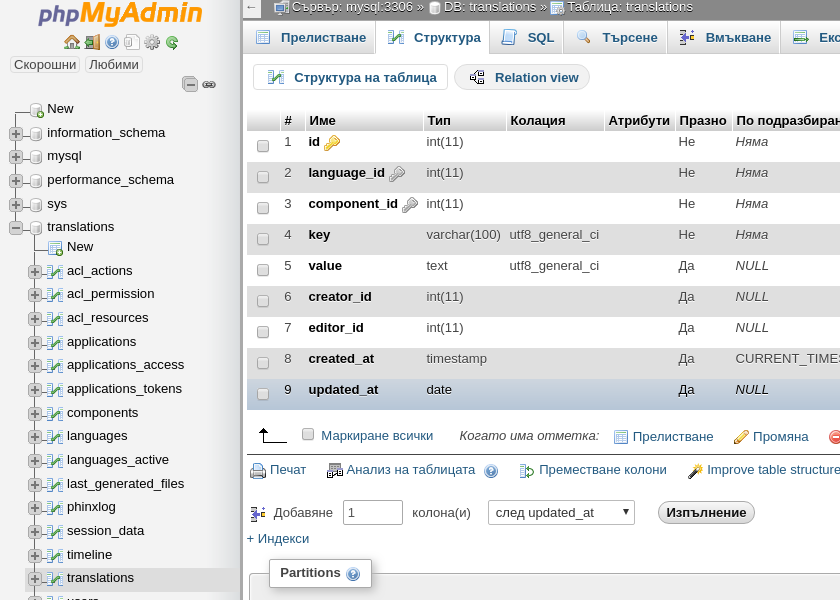


Фигура 51. Таблица последно генерирани файлове

Translations

В таблицата с преводи се пазят:

* Индетификационен номер.
* Индетификационен номер на езика.
* Индетификационен номер на компонента.
* Ключ.
* Стойност.
* Индетификационен номер на създателя.
* Индетификационен номер на редактиралия.
* Кога е създаден.
* Кога е редактиран.

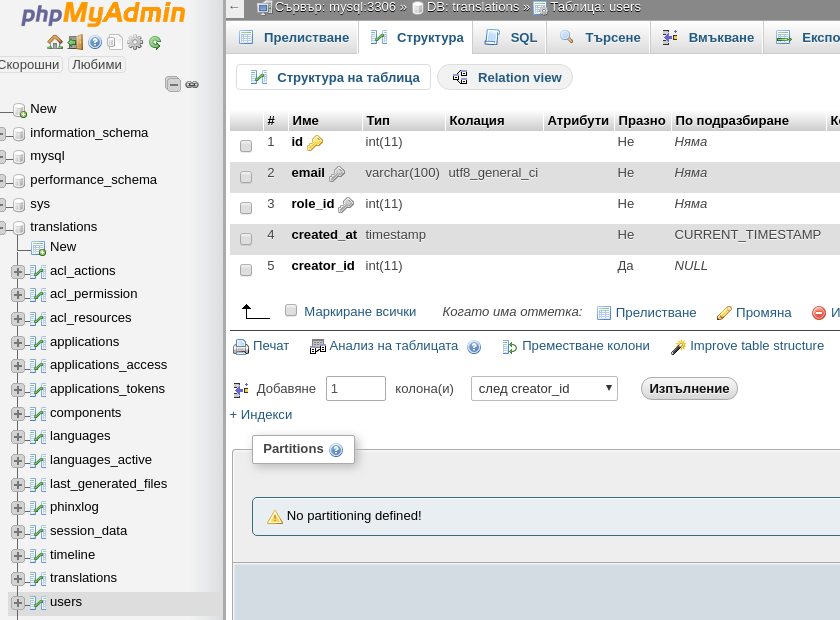


Фигура 52. Таблица преводи

Users

В таблицата с потребители се пазят:

* Индетификационен номер.
* Имейл на потребителя.
* Индетификационен номер на ролята.
* Кога е създаден профила.
* Кой е създал профила.

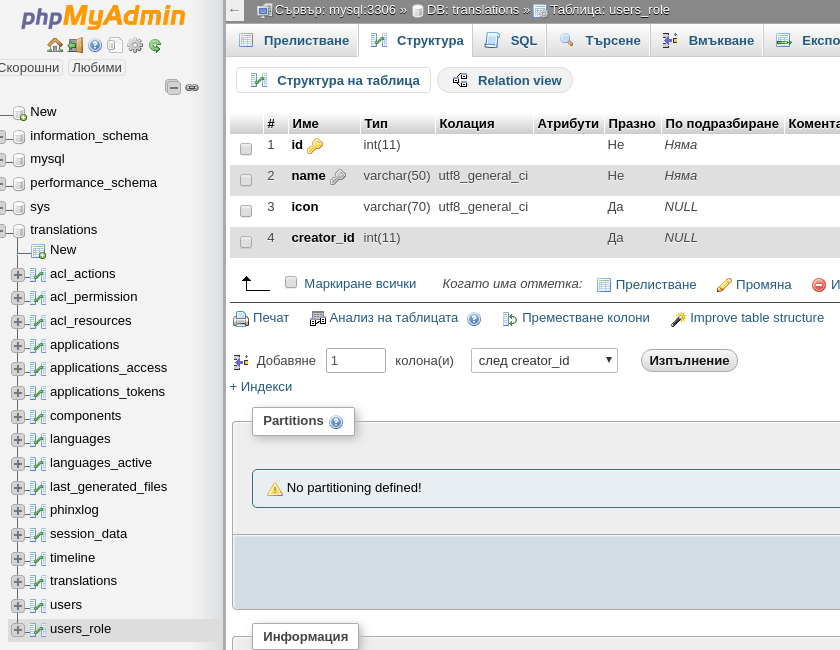


Фигура 53. Таблица потребители

Users Role

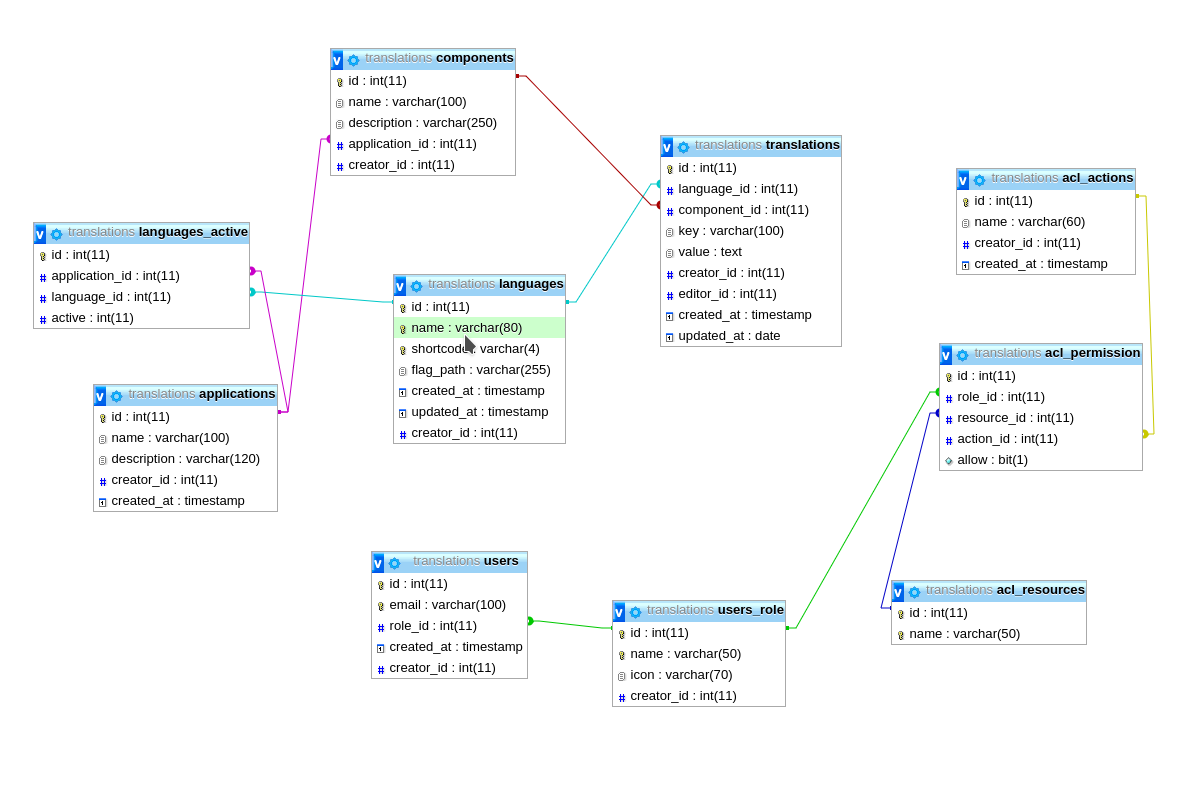
В таблицата с потребителски роли се пазят:

* Индетификационен номер.
* Име на ролята.
* Иконка на ролята
* Индетификационен номер на създателя.



Фигура 54. Таблица роли

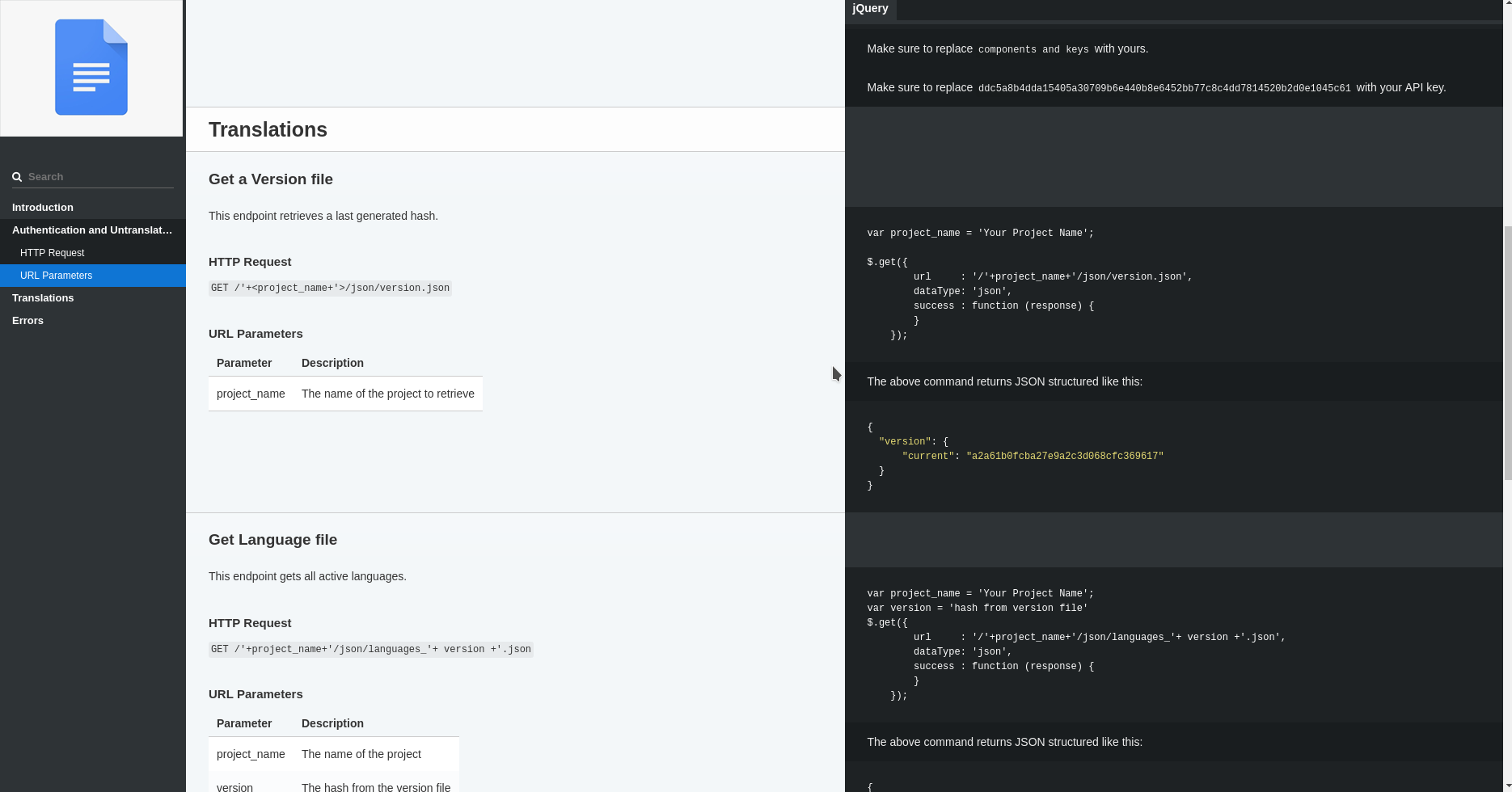
## Релации



Фигура 55. Връзките в базата

## Документация за api

В системата има подробна документация за използване на api. Документацията е създадена с помоща на шаблона Docs, разработен на руби специално за обработка на документации.



Фигура 56. Документация

Документацията е разписана подробно с примери и изход от системата може да се достъпи с „***/docs“***.

След написването и се генерират папки с изходящ код и стелизирани документи.

Глава 5. Заменяне, SEO, Домей хостинг, Тестване и усъвършенстване



## Заменяне и трансформиране на елементи

Визията и структура на уеб сайта трябва да се променят, без да се премахва съдържание или функционалност за определено устройство или размер на дисплей. Посетител, който преглежда уеб сайта от своя смартфон или таблет, трябва да има достъп до същото съдържание и функционалност, до което би имал и посетител, който използва десктоп компютър. Всички елементи в адаптивния уеб сайт трябва да са гъвкави, което изисква при оформлението на дизайна да сме сигурни, че снимки, съдържание и цялостна структура, на която и да е страница в уеб сайта ще се преконфигурират и пренаредят, съобразявайки ги с различните размери на екрани.

Някои елементи от страницата са скрити и са заменени с по-подходящи такива за съответната разделителна способност. Един такъв елемент е менюто, който е заменен с падащо меню подходящо за работа с таблети и смартфони. Добре е да се знае, че при проектирането на дизайна за по-малки устройства е добре да се запази четливостта и размера на шрифта, така че текстът на страниците на сайта да се вижда добре дори и на малки устройства.

Трябва да се възползваме и от новите удобства на докосването и плъзгането по екрана, които ни предоставят таблетите и смартфоните. Тях можем да прихванем с Javascript и да създадем допълнителни функционалности за улеснение на работата със сайта.

## SEO оптимизация

В превод *SEO / search engine optimization/* означава оптимизиране на сайт за търсачки каквито са Google, Bing, Yahoo и други. Оптимизация за търсачки е процесът на подобряване видимостта на уеб-сайт в търсачките чрез неплатени резултати, който включва оптимизация на html-кода, структурата и текстовете на уеб сайт. Рекламирането на сайта и увеличаването на вътрешните и обратните връзки (backlinks) е още една от стратегиите за SEO оптимизация. Фактически, колкото по-рано и по-често се появява един сайт в резултатите, толкова повече посетители е вероятно да му генерира търсачката. Така сайтът получава присъствие в Интернет пространството. Оптимизацията може да включва различни аспекти за търсене, като търсене в определена сфера или регион, както и търсене на картинки, видео или новини. Оптимизацията за търсачки изследва как работят търсачките и какво търсят хората.

Даден сайт може да съдържа много важна и ценна информация, продукти, полезни онлайн инструменти или материали, но потребителите в интернет да не успяват да го намират, защото не излиза на предно място в листа с резултати от търсенето или съдържанието му въобще не се индексира от Google или други търсачки. За това оптимизацията за търсачки е необходима част от работата по даден уеб сайт, за да бъде лесно откриван по важните ключови думи и фрази свързани със съдържанието му.

## Домейн и хостинг за сайта

След като подготвим проекта за ъплоудване на сървър и публикуването му в мрежата трябва да изберем подходящ домейн за сайта. В случая, тъй като сайта е за система за управление на проекти, то трябва домейнът да има уникално и запомнящо се име.

Щом изберем подходящ домейн можем да се насочим към намирането на хостинг план, подходящ за нуждите на проекта и съобразен с очакваното натоварване и трафик на сайта. Тук предлагащите подобна услуга са изключително много и изборът може би ще се сведе до предпочитания и оферти

## Тестване

Разбира се, преди да бъде промотиран в интернет пространството, сайтът трябва да бъде подложен на щателна проверка от гледна точка на сигурност, скорост, правилно поведение и пр. За целта могат да се използват много от готовите инструменти за тестване и анализ на уеб страници предоставени от водещите компании като Google, Bing и др. Също така е добре да се тества на най-популярните мобилни устройства, водещи на пазара.

Трябва да се провери сигурността на сайта, да се уверим, че всяка страница, която е трябва да бъде защитена, е защитена наистина и напълно безопасна за използване. Също така дали всяка от функционалностите работи както се очаква и изпълнява функциите си.  
 Добре ще е да се интегрират и някои скриптове за анализ, които ще ни предоставят статистическа информация за почти всичко, което се случва на сайта.

## Усъвършенстване

За да може даден проект да е успешен, освен добрата идея, която стои зад него, той има нужда и от непрекъснато усъвършенстване. Поради това, за следващите версии на платформата, са предвидени редица подобрения. Някои от тях са:

* Добавяне на още няколко шаблона за презентационния уеб сайт.
* Добавяне на изкуствен интелект с цел по-бързи преводи.
* Добавяне на нови възможности за експортване на файлове.
* Добавяне на нови разширения за сваляне на файловете.
* Добяване на нови начини за вписване в системата.

Авторска справка

В настоящата дипломна работа е представена разработка на уеб-базирано приложение за управление на преводи. Описани са основните технологии, използвани за реализацията на функционалността и дизайна на уеб приложението.

Разработен е авторски модел за управление на проекти, който има за цел да осигури бърз и лесен достъп на потребителите до платформата. Целта е постигната чрез интегрирането на приложението и създаването на база от данни.

Основните приноси на дипломната работа са:

* Създадено е работещо уеб приложение за управление на преводи.
* С помощта на *приложението* са разработени редица функционалности и дизайн, на уеб приложението.
* Лесен и опростен дизайн за употреба (параграф 3.2 от дипломната работа).
* Функционалност при търсенето на конкретен проект (параграф 3.4 от дипломната работа).
* Форма за контакт, чрез която посетителите на уеб приложението ще могат да отправят запитвания и предложения за бъдещото развитие на проекта( параграф 3.4 от дипломната работа).
* Описание на приложението, представено в Глава 3.
* Поле за коментари от потребители (параграф 3.2 от дипломната работа).
* Използвани разширения за функционалността на приложението (параграф 3.4 от дипломната работа).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Принос** | **Вид принос** | **Цел** | **Задачи** | **Параграф** |
| **1.1** | **Приложен** | **2.1, 2.2** | **1.1** | **3.2, 3.4** |
| **1.2** | **Приложен** | **2.1, 2.2** | **1.1** | **3.2, 3.4** |
| **2.1** | **Приложен** | **2.1, 3.2, 3.3** | **1.2, 1.3** | **3.2, 3.4** |
| **2.2** | **Приложен** | **2.1, 3.2, 3.3** | **1.2, 1.3** | **3.2, 3.4** |
| **3.1** | **Приложен** | **2.1, 3.1, 3.3** | **1.1, 1.2, 1.3** | **3.2, 3.4** |
| **4.1** | **Приложен** | **2.1, 2.3, 3.3** | **2.1** | **3.2, 3.4** |

Заключение

Налагането на технологиите във всички сфери на дейности показва нуждата от системи за управление на преводи. Представената системата за управление на преводи би била от полза, както за големи компании, така и за малки. Тя ще улесни работата им чрез редицата разработени функции и удобен дизайн.

Целта на тази дипломна работа е успешно постигнатa и програмният продукт е в работещо състояние. Технологиите, които са използвани за разработката му, позволяват лесно по-нататъчно развитие и разширяване на функционалности. Основната посока може да бъде добавяне на повече функции и модули.

Под внимание са взети тенденциите в уеб дизайна, като е обърнато особено внимание на цветовата гама, използвана за изграждането на уеб приложението. Нейният дизайн е изграден така, че да бъде модерен и уникален, и да не затруднява използването на системата от потребителя.

Системата притежава редица предимства, изброени в отделните части от описания проект, които ѝ помагат да изпъква пред останалите конкуренти.

Използвана литература

1. Стоянов, Ст., А. Стоянова-Дойчева, М. Трендафилова, Е. Дойчев, *Софтуерни технологии*, Пловдивско университетско издателство, Пловдив, 2006.
2. Денис Колесниченко, *HTML 5 & CSS 3*. *Практическо програмиране за начинаещи*, трето преработено издание.
3. Денис Колесниченко, *PHP & MySQL. Практическо програмиране,* второ преработено издание.
4. Денис Колесниченко, *JavaScript & jQuery. Практическо програмиране.*
5. Wordbee, https://www.wordbee.com/, *последно посетен на 26.09.2018 г.*
6. Lingotek https://www.lingotek.com/, *последно посетен на 26.09.2018 г.*
7. Ubrta https://www.csoftintl.com/technology/ubrta/, *последно посетен на 26.09.2018 г.*
8. Джейсън Гернър, Елизабет Нарамор, „Програмиране и Web дизайн с PHP5, MySQL, Apache” първи и втори том, изд. АлексСофт
9. MySQL, http://dev.mysql.com/doc/refman/5.7/en/, *последно посетен на 26.09.2018 г.*
10. HTML, CSS, JavaScript, http://www.w3schools.com, *последно посетен на 26.09.2018 г.*
11. Best PHP IDE in 2014 – Survey Results https://www.sitepoint.com/best-php-ide-2014-survey-results/, *последно посетен на 26.09.2018 г.*
12. MySQL, http://dev.mysql.com/doc/refman/5.7/en/, *последно посетен на 26.09.2018 г.*