Taikomieji moksliniai tyrimai / Applied Scientific Research ISSN 2783-6290
2023, 2(2), 140-150. DOI: https://doi.org/10.56131/tmt.2023.2.2.161 Gauta 2023-05-30. Publikuota 2023-10-31

RESTful API žiniatinklio paslaugų diegimas duomenų administravimui skirtinguose serveriuose

Donatas Daugirdas
Šiaulių valstybinė kolegija, Informatikos mokslų katedros lektorius
Šiaulių valstybinė kolegija / Higher Education Institution, Lithuania; Lecturer at the Department of Informatics Sciences
d.daugirdas@svako.lt

Mantas Kryževičius
Šiaulių valstybinė kolegija, Programų sistemų studijų programos absolventas
Šiaulių valstybinė kolegija / Higher Education Institution, Lithuania; Graduate of the Program Systems study program

Anotacija

Straipsnyje analizuojamas INOSTART programos vykdomo projekto metu sukurtas novatoriškas modelis, pagerinantis įmonės internetinių puslapių informacijos kėlimo bei atnaujinimo procesą. Ši sistema - modelis įmonei padės sparčiai pakeisti esamą, dažnu atveju besikartojančią informaciją visose jos buvimo vietose, supaprastins jos valdymo procesą, suteiks prieigą prie jos valdymo IT ir WordPress žinių neturintiems asmenims bei išspręs dažniausiai besikartojančias daugiakalbystės palaikymo problemas.

Reikšminiai žodžiai: duomenų valdymas, administravimas, RESTful API, sistema.

Applying a RESTful API Approach to Data Administration across Different Servers

Summary

The article analyzes the model developed during the INOSTART program project that improves the process of uploading and updating information on the company's web pages. This system will help the company to quickly change existing, often repetitive information in all its locations, simplify its management process, provide access to its management to persons without IT and WordPress knowledge, and solve the most frequent problems of multilingual support.

Keywords: data management, administration, RESTful API, system.

Įvadas

Temos aktualumas. INOSTART programa skirta informacinių technologijų programinių produktų, inžinerinių sprendimų projektams parengti ir plėtoti. Programą įgyvendina Šiaulių miesto teritorijoje esančių aukštųjų mokyklų studentai, bendradarbiaudami su įmonių atstovais ir pasirinktos (informacinių technologijų, inžinerinės) srities mokslininkais pagal įmonės pateiktą poreikį. Šio projekto metu buvo iškelta problema, jog vidutinės ir didelės įmonės, atskyrusios savo veiklos sritis, turi ne vieną internetinę svetainę, įvairiai auditorijai ją pateikia skirtingomis kalbomis. Norint, jog besikartojančios ar vizualiai sunkiai randamos informacijos valdymas skirtingose platformose būtų sklandus, praktikoje plačiai naudojamos centralizuotos duomenų valdymo sistemos, o WEB sistemų valdymo atveju centralizuotos duomenų valdymo sistemos yra pagrįstos RESTful API metodu.

Žiniatinklio programų programavimo sąsajos (API) yra ryšių būdas, leidžiantis sistemoms tarpusavyje sąveikauti internete, suteikiant galimybę pasiekti žiniatinklio paslaugas nuotoliniu būdu

[15]. Naujų technologijų, tokių kaip debesų paslaugos [16], paslaugų sujungimas (angl.“mashup“) [17] ir mikropaslaugos [18], visos remiasi žiniatinklio API ir skatina šių technologijų tobulėjimą ne tik aspektais, susijusiais su dizainu ir našumu, bet ir patogumu administruojant sistemas [19]. RESTful API (Representational State Transfer) yra architektūrinis stilius, kuris nustato taisykles ir apribojimus, kaip klientai ir serveriai turi bendrauti tarpusavyje per HTTP protokolą. Kartu tai yra labai populiarus ir dažnai naudojamas metodas kurti bei integruoti WEB paslaugas. Šis metodas yra populiarus duomenų perdavimui tarp skirtingų serverių bei gali atlikti pagrindinių veiksmų užklausas, pvz., sukurti, skaityti, atnaujinti arba ištrinti įrašą [11]. Naudojant centralizuotą duomenų valdymo sistemą, supaprastėja bei pagreitėja informacijos radimo bei keitimo procesai, o juos gali atlikti asmenys, neturintys profesionalių techninių interneto svetainių turinio valdymo sistemos žinių.

Mokslinė problema straipsnyje keliama šiais probleminiais klausimais: Kaip sprendžiamas duomenų administravimas skirtinguose serveriuose? Kokiomis priemonėmis galima įgyvendinti valdymo sistemų turinio administravimą per vieną centralizuotą duomenų valdymo sistemą?

Tyrimo objektas – centralizuota duomenų valdymo sistema.

Tyrimo tikslas – sukurti ir įdiegti centralizuotą puslapių įrašų valdymo sistemą bei su ja sąveikaujantį įskiepį.

Tyrimo uždaviniai: 1) Išanalizuoti įmonėje naudojamų interneto svetainių struktūrą ir poreikį kurti naują sistemą; 2) Nustatyti problemą, su kuria susiduria įmonė; 3) Išanalizuoti panašaus tipo sistemas; 4) Suprojektuoti bei parengti centralizuotos duomenų valdymo sistemos modelį.

Tyrimo metodai: mokslinės literatūros analizė, įmonės duomenų analizė, sisteminimas, projektavimas, unikalaus prototipo kūrimas.

Užsakovo poreikių analizė

Daugelis įmonių susiduria su duomenų valdymo sunkumais ir viena pagrindinių to priežasčių – jų centralizavimo nebuvimas [3]. Norint, kad verslas sėkmingai vykdytų savo veiklą internetiniuose puslapiuose, svarbu pateikti šiuos elementus bei juos laiku atnaujinti:

1. *Produkto ar paslaugos informacija:* organizacijos puslapyje turėtų būti išsamiai aprašytos verslo siūlomos prekės ar paslaugos. Tai apima produkto aprašymus, technines specifikacijas, naudojimo instrukcijas, nuotraukas ar net vaizdo įrašus.
2. *Kontaktinė informacija:* pateikta aiški informacija apie kontaktus, kurie leistų lankytojams susisiekti su organizacijos atstovais. Tai gali būti telefono numeris, el. pašto adresas, kita kontaktinė forma ar net gyvosios paieškos sistema.

Centralizuota duomenų bazė – tai tokia duomenų bazė, kuri yra prižiūrima bei laikoma vienoje centralizuotoje vietoje, t. y. visi duomenys yra saugomi vienoje fizinėje vietoje arba serverio sistemoje. Tai reiškia, kad visi klientai ir programos, kurios naudoja šią duomenų bazę, turi prieigą prie tų pačių duomenų, nepriklausomai nuo to, kokią informaciją jie ieško ar modifikuoja [4]. Centralizuota duomenų bazių praktika dažniausiai taikoma įmonėse bei mokymo įstaigose, kuriose darbo procesai vyksta skirtingose vietose ir todėl gali sukelti duomenų paieškos, įvedimo bei keitimo sunkumų [9]. Centralizuota duomenų bazė turi daug privalumų [3, 8].

Visuose UAB „Vandenvala“ įmonių grupės puslapiuose naudojama WordPress turinio valdymo sistema. Visus įmonių grupės puslapius administruoja tik įmonės IT administratorius, turintis technines žinias IT srityje. Šiuose puslapiuose patalpinta informacija kartojasi daugelyje vietų, kurią pakeisti gali tik WordPress ir programavimo žinių turintis žmogus. Informacija juose keičiama dažnai (ne rečiau kaip kartą per mėnesį), tai užima nemažai laiko, kadangi didžiausia problema – besikartojančios informacijos kiekis bei paieška, kuriose vietose ji patalpinta.

Siekiant išsiaiškinti įmonės poreikį modernizuoti duomenų valdymą, buvo vykdomas susitikimas su įmonės atstove bei atlikta naudojamų sistemų analizė. Pokalbis vyko su rinkodaros vadove, kurio metu buvo nuspręstas sistemos kūrimo poreikis, funkcionalumas bei dizainas: 1) Reikalinga centralizuota duomenų valdymo sistema, kurios pagalba bus galima valdyti įmonių grupės puslapiuose esančią informaciją vienoje vietoje; 2) Reikalingas įrašų daugiakalbystės palaikymas; 3) Reikalingas automatinis įrašų vertimo į kitas kalbas funkcionalumas; 4) Reikalingas blokų

redaktorius; 5) Reikalingas papildomų vartotojų prie sistemos pridėjimas; 6) Informacija turi būti kešuojama kiekvienoje svetainėje; 7) Reikalingas pasirinktų įrašų vertimų adresas vertimų biurams; 8) Dizainas turi būti paprastas ir aiškus. Nuspręsta. jog reikalinga sukurti modelį kuriame bus WordPress įskiepis informacijos sinchronizavimui. Kiekvienoje įmonės svetainėje turi būti įdiegtas įskiepis, kuris galės apdoroti naujai sukurtos sistemos įrašus bei juos išsaugoti duomenų bazėje.

Sistemos techninė analizė

Būsima sistema turi būti pasiekiama tiek kompiuteryje, tiek telefone, todėl nuspręsta, jog bus kuriama WEB aplikacija. Norint sukurti tokio tipo sistemą, reikia tinkamai pasirinkti programavimo kalbą. Remiantis programinės įrangos kokybės vertinimo kompanijos TIOBE 2022 gruodžio mėnesio pateiktais statistikos duomenimis, į populiariausių programavimo kalbų dešimtuką patenka tokios programavimo kalbos kaip "PHP" ir "JavaScript", jos yra tinkamos WEB aplikacijų kūrimui [2, 9].

PHP yra plačiai paplitusi dinaminė interpretuojama programavimo kalba (angl. Hypertext Preprocessor),specialiai pritaikyta interneto svetainių kūrimui [6]. Ji gyvuoja nuo 90-ųjų vidurio ir tapo galinga kalba, sukėlusią revoliuciją žiniatinklio kūrimo pramonėje. [22] PHP kalba iki šiol naudojama daugelyje populiarių aplikacijų, pavyzdžiui Facebook, Yahoo ar Wikipedia [7]. Šia kalba yra nesudėtinga naudotis, o susidūrus su neaiškumais, internete yra daugybė forumų bei oficiali dokumentacija, kur galima rasti reikiamos informacijos. PHP yra viena iš populiariausių skriptų kalbų, naudojamų kuriant žiniatinklius ir buvo naudojama daugelį metų.

Ši programavimo kalba palaiko daugybę duomenų bazių, turi įvairių modulių, yra atvirojo kodo, o kodas vykdomas serverio pusėje.

JavaScript yra programavimo kalba, vykdoma kliento pusėje ir yra viena pagrindinių žiniatinklio technologijų [10]. Atsižvelgiant į projekto reikalavimus, šiuolaikines tendencijas ir programavimo kalbų galimybes, projektu kurti bus naudojamos PHP ir JavaScript programavimo kalbos, su kuriomis bus galima pilnai įvykdyti užsakovo poreikius.

Esamoms ir būsimoms sistemoms būtina, jog jos veiktų stabiliai, nebūtų komplikuota jų architektūra ir būtų paprastas jos palaikymas, modifikavimas bei naujų funkcijų integravimas, todėl reikia pasirinkti tinkamą karkasą [12]. Programavimo kalbos karkasas yra įrankių rinkinys, kurio pagalba galima kurti gerai struktūrizuotą, patikimą programinę įrangą ir sistemas [8]. Karkasų dėka didelė dalis besikartojančių problemų jau yra išspręsta, dėl to aplikacijų kūrimas tampa daug spartesnis.

Esamoms ir būsimoms sistemoms būtina, jog jos veiktų stabiliai, nebūtų komplikuota jų architektūra ir būtų paprastas jos palaikymas, modifikavimas bei naujų funkcijų integravimas, todėl reikia pasirinkti tinkamą karkasą [12].

Pasirinkta programavimo kalba – PHP, todėl analizuojami (žr. 2 lent.) Laravel, Symfony ir CodeIgniter naujausių versijų PHP karkasai [2].

2 lentelė

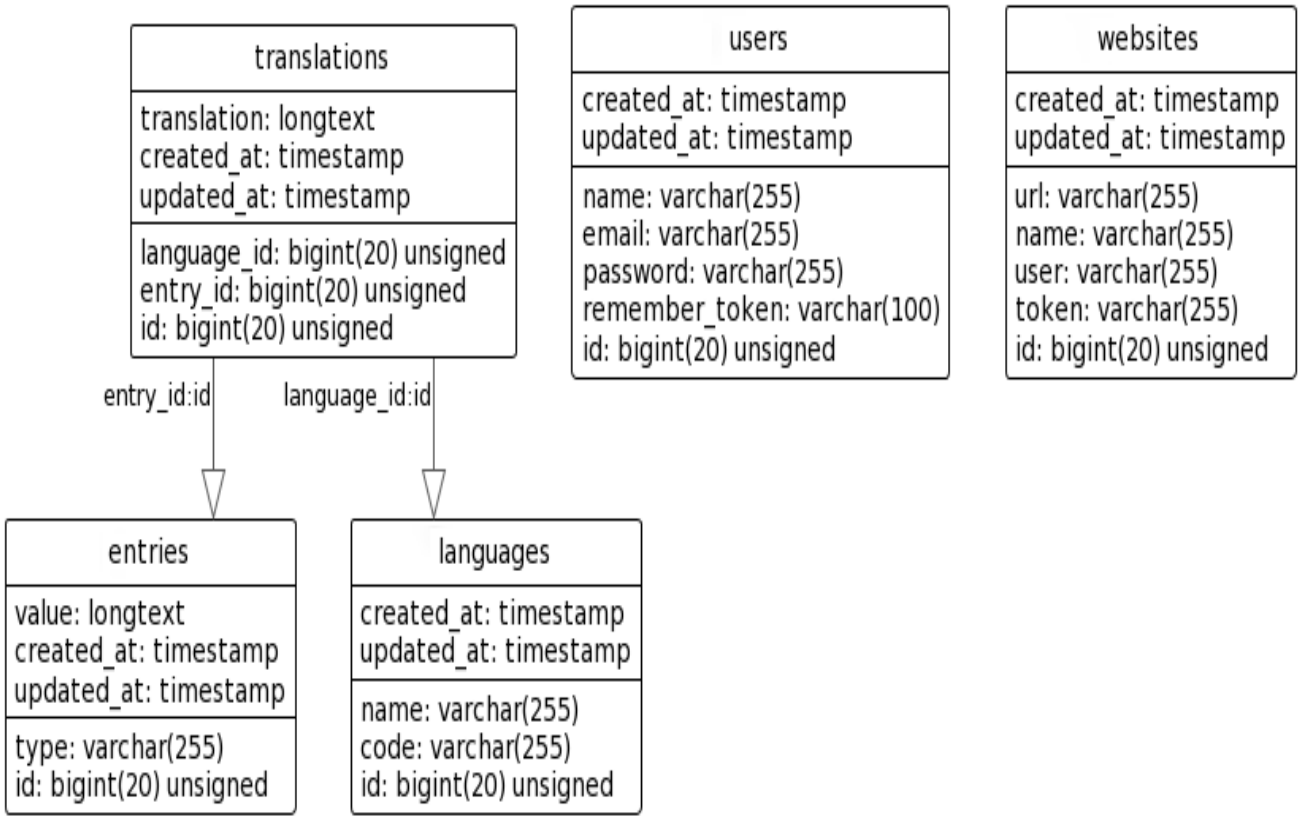
PHP karkasų palyginimas

	Laravel	**Symfony**	**CodeIgniter**
Šablonų variklis	Blade	Twig	Nėra
Panaudojimas	Mažoms ir vidutinėms sistemoms	Didelėms sistemoms	Mažoms ir vidutinėms sistemoms
ORM	Eloquent	Doctrine	Nėra
Architektūra	MVC	MVC	MVC
Priklausomybių įterpimas	Yra, automatinis sujungimas	Yra	Nėra
Testavimo biblioteka	Yra, PHPunit	Yra, PHPunit	Yra, PHPunit
Implementuota autorizacija	Yra	Yra	Nėra
Palaikomas užklausų kiekis per sekundę	102	42	480
Kodo generavimas	Yra, per CLI	Yra, per CLI	Nėra
Licencija	MIT	MIT	BSD-style

Įvertinus karkasų galimybes, tokias kaip architektūra, priklausomybių įterpimas, testavimo biblioteka, implementuoti sprendimai, galimybė automatiškai generuoti kodą ir licenciją (žr. 2 lent.), bei remiantis jau atliktais viešai paskelbtais tyrimais [20, 21, 23, 24]nuspręsta naudoti Laravel karkasą, kuriuo yra pakankamai lengva naudotis, jis turi daug integruotų įrankių, atitinka įmonės poreikius bei turi *ORM* [5], kuris supaprastina duomenų bazės įrašų valdymą bei kurį galima naudoti komerciniais tikslais.

Duomenų bazės projektavimas

Sistemos duomenų bazės struktūros schemoje (žr. 1 pav.) pateikta duomenų tipų informacija. Duomenų bazės lentelėse saugomi sistemoje atlikti pakeitimai. Lentelėje *users* yra saugoma informacija apie sistemos vartotojus, jų vardus, el. pašto adresus, užkoduotus slaptažodžius. Lenelėje *"WEB"sites* saugomos sistemoje pridėtos svetainės, jų adresai, pavadinimai, vartotojai ir API raktai. Lentelėje *entries* saugomi įrašai su nurodytu tipu, ši lentelė yra susieta su lentele *translations*, kurioje saugomi įrašų vertimai. Lentelė *translations* susieta su lentelės *languages* įrašais, kur saugomos sistemoje pridėtos kalbos pavadinimai ir kodai.



1 pav. Duomenų bazės struktūra

3 lentelėje pateikta duomenų bazės struktūros detalizacija.

3 lentelė

Duomenų bazės struktūros detalizavimas

Users			
Pavadinimas	**Tipas**	**Savybės**	**Aprašymas**
id	bigint (20)	Primary key, not null	Identifikavimo numeris
name	varchar (255)	not null	Vartotojo vardas
email	varchar (255)	not null	Vartotojo el. pašto adresas
password	varchar (255)	not null	Vartotojo slaptažodis
remember_token	varchar (100)	nullable	Sesijos ID
created_at	timestamp	not null	Vartotojo sukūrimo laikas
updated_at	timestamp	not null	Vartotojo atnaujinimo laikas

"WEB" sites			
Pavadinimas	**Tipas**	**Savybės**	**Aprašymas**
id	bigint (20)	Primary key, not null	Identifikavimo numeris
url	varchar (255)	not null	Svetaines adresas
name	varchar (255)	not null	Svetaines pavadinimas
user	varchar (255)	nullable	Svetainės administratoriaus prisijungimo vardas
token	varchar (255)	nullable	Svetainės vartotojo API raktas
created_at	timestamp	not null	Svetaines sukūrimo laikas
updated_at	timestamp	not null	Svetaines atnaujinimo laikas

Translations			
Pavadinimas	**Tipas**	**Savybės**	**Aprašymas**
id		Primary key, not null	Identifikavimo numeris
language_id	bigint (20)	unsigned, not null	Kalbos identifikavimo numeris
entry_id	bigint (20)	unsigned, not null	Įrašo identifikavimo numeris
translation	longtext	not null	Vertimas
created_at	timestamp	not null	Vertimo sukūrimo laikas
updated_at	timestamp	not null	Vertimo atnaujinimo laikas

Entries			
Pavadinimas	**Tipas**	**Savybės**	**Aprašymas**
id	bigint (20)	Primary key, not null	Identifikavimo numeris
type	varchar (255)	not null	Įrašo tipas
value	longtext	not null	Įrašas
created_at	timestamp	not null	Įrašo sukūrimo laikas
updated_at	timestamp	not null	Įrašo atnaujinimo laikas

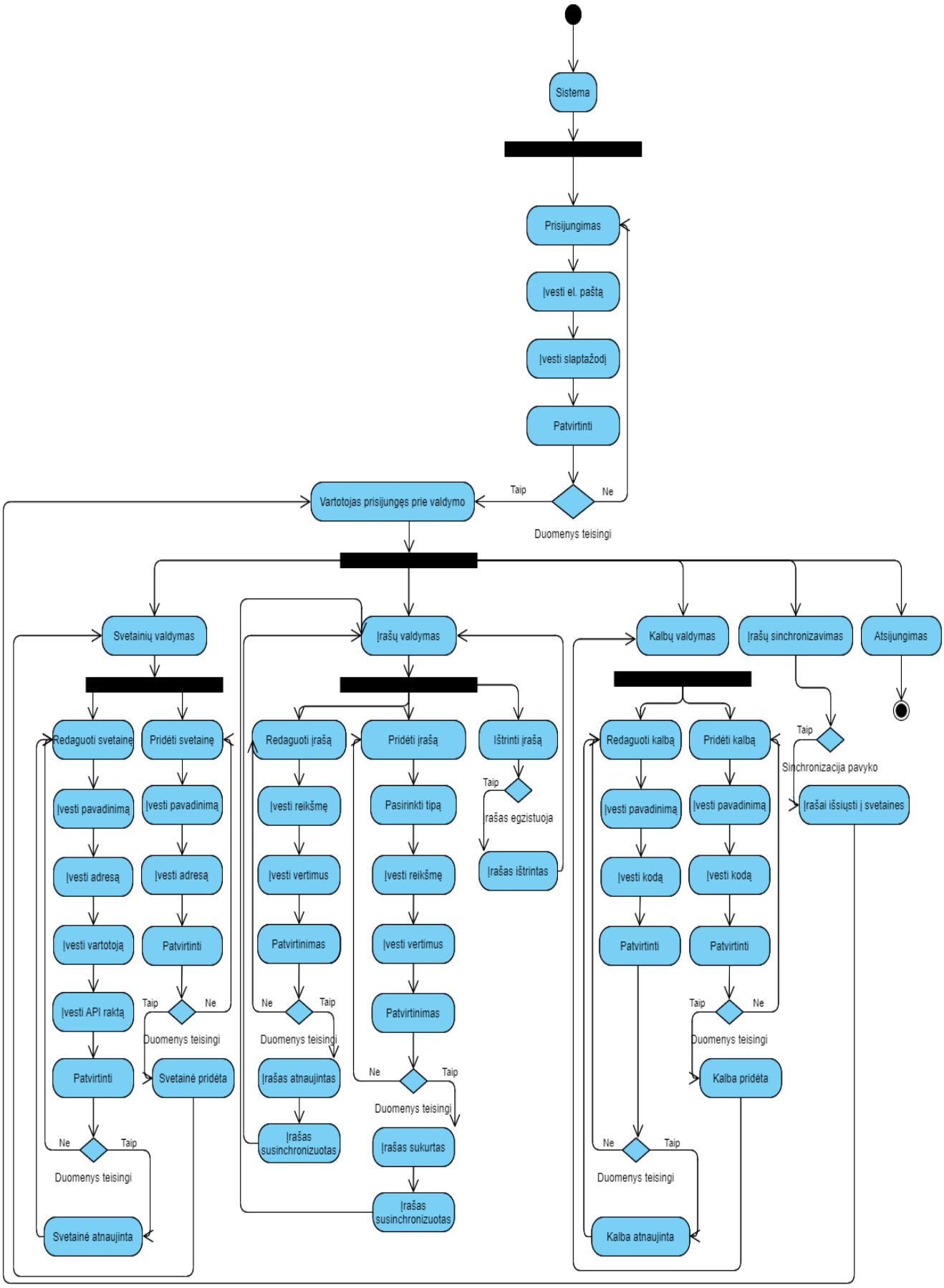
Languages			
Pavadinimas	**Tipas**	**Savybės**	**Aprašymas**
id	bigint (20)	Primary key, not null	Identifikavimo numeris
name	varchar (255)	not null	Kalbos pavadinimas
code	varchar (255)	not null	Kalbos kodas
created_at	timestamp	not null	Kalbos sukūrimo laikas
updated_at	timestamp	not null	Kalbos atnaujinimo laikas

Kuriamos sistemos duomenų bazės struktūra yra nekomplikuota, o parinkti duomenų tipai ir parametrai sumažina galimas netinkamai įvestų duomenų rizikas. Sistemos tobulinimo atveju lentelės bus lengvai modifikuojamos ir plečiamos, kadangi sudėti teisingi sąryšiai ir parametrai.

Centralizuotos duomenų valdymo sistemos veiklos diagramos projektavimas

Sistemų kūrėjai dažnai naudoja Unified Modeling Language (UML) veiklos diagramas, siekdami išsamiai pavaizduoti visus valdymo priemonių srautus, kurie paprastai vadinami naudojimo atvejų scenarijais. Todėl veiklos diagrama laikoma vertingu dizaino artefaktu, leidžiančiu identifikuoti visus potencialius scenarijus ir tikrinti klaidas naudojimo atvejo scenarijų metu. Sistemos veiklos diagramoje (žr. 2 pav.) pateikta sąlyginė sistemos veikla. Veiklos diagrama vaizdžiai pateikia sistemoje seką veiksmų, kurie dažnai naudojami verslo procesų modeliavimui [12, 13].

Sistemos vartotojas pirmiausia turi galimybę tik prisijungti. Vartotojas prijungiamas tik teisingai suvedus prisijungimo duomenis. Sėkmingai prisijungęs vartotojas gali atlikti tokius veiksmus, kaip svetainių valdymas, įrašų valdymas, kalbų valdymas, įrašų sinchronizacija arba atsijungimas.



2 pav. Sistemos veiklos diagrama

4 lentelėje pateikta šios diagramos detalizacija (žr. 4 lent.).

4 lentelė

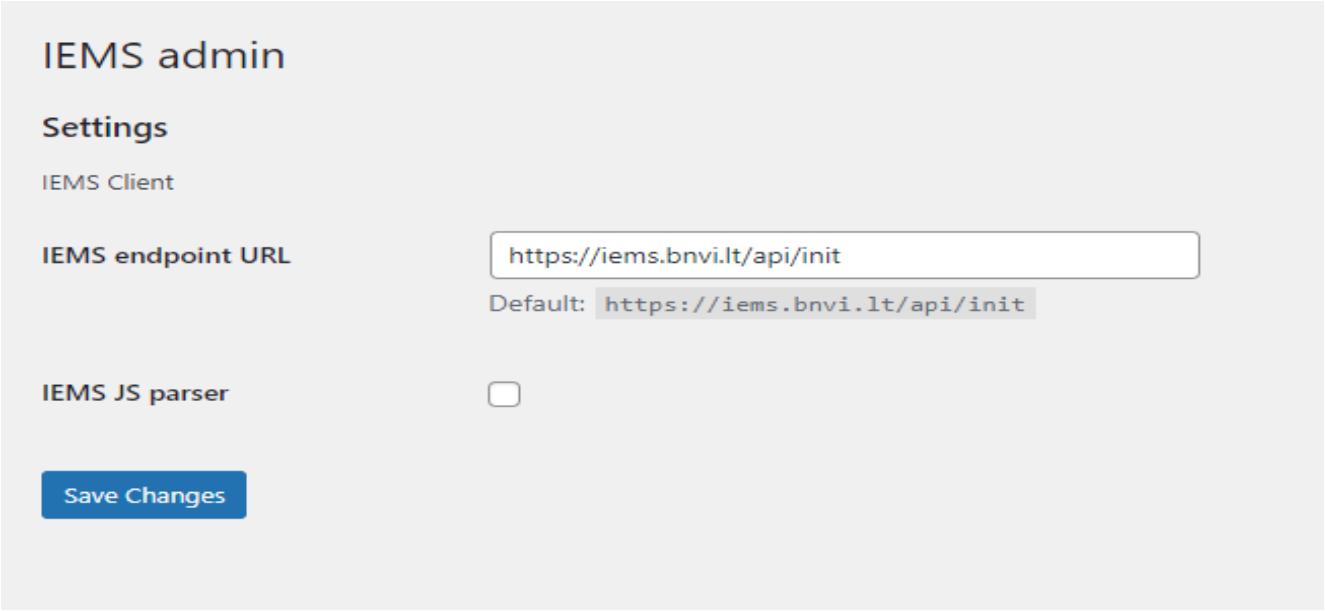
Veiklos diagramos detalizavimas

Įvykis	**Žingsnis (seka)**	**Rezultatas**	**Išvada**
1. Prisijungimas prie valdymo	Norint prisijungti prie valdymo reikia suvesti reikiamus duomenis (el. pašto adresą ir slaptažodį) ir spausti mygtuką „Prisijungti“	Vartotojas prisijungė prie valdymo	Tinkamai suvedus duomenis, vartotojas prijungiamas prie valdymo
2. Pridėti svetainę	Norint pridėti svetainę reikia įvesti svetainės pavadinimą ir adresą	Svetainė pridėta	Tinkamai suvedus svetainės duomenis, svetainė pridedama
3. Redaguoti svetainę	Norint redaguoti svetainės duomenis, reikia paspausti mygtuką “Redaguoti”	Svetainės duomenys atnaujinti	Tinkamai suvedus svetainės duomenis, svetainės duomenys yra atnaujinami
4. Pridėti įrašą	Norint pridėti įrašą reikia pasirinkti įrašo tipą bei įvesti reikšmes ir vertimus	Įrašas sukurtas	Tinkamai suvedus duomenis, įrašas yra pridedamas
5. Redaguoti įrašą	Norint redaguoti įrašą reikia paspausti mygtuką “Redaguoti” ir suvesti naujas įrašo reikšmes ir vertimus	Įrašas atnaujintas	Tinkamai suvedus duomenis, įrašo duomenys yra atnaujinami
6. Ištrinti įrašą	Norint ištrinti įrašą reikia paspausti mygtuką “Ištrinti”	Įrašas ištrintas	Pasirinkus norimą įrašą ir paspaudus mygtuką “Ištrinti”, įrašas yra ištrinamas
7. Pridėti kalbą	Norint pridėti kalbą reikia įvesti kalbos pavadinimą ir kalbos kodą	Kalba pridėta	Tinkamai suvedus duomenis, kalba yra pridedama
8. Redaguoti kalbą	Norint redaguoti kalbos duomenis, reikia įvesti naują kalbos pavadinimą ir kalbos kodą	Kalbos duomenys atnaujinti	Tinkamai suvedus duomenis, kalbos duomenys yra atnaujinami
9. Sinchronizuoti įrašus	Norint sinchronizuoti įrašus, reikia paspausti mygtuką “Sinchronozuoti įrašus”	Įrašai išsiųsti į svetaines	Paspaudus mygtuką “Sinchronizuoti įrašus” įrašai yra išsiunčiami į svetaines
10. Atsijungti	Norint atsijungti nuo valdymo, reikia paspausti mygtuką “Atsijungti”	Vartotojas atsijungia nuo valdymo	Paspaudus mygtuką “Atsijungti”, vartotojas yra atjungiamas nuo valdymo

Iš 4 lentelėje pateiktų duomenų galime matyti, kad prisijungęs vartotojas gali pasirinkti svetainių valdymą, kuriame gali redaguoti arba pridėti svetainę. Norint pridėti svetainę, reikia suvesti tik jos pavadinimą ir adresą, o suvedus šiuos duomenis reikia patvirtinti. Jei duomenys teisingai suvedami, vartotojui vėl rodomas svetainių valdymas. Svetainės redagavimas beveik nesiskiria nuo pridėjimo, tik jame reikia papildomai įvesti svetainės vartotoja ir API raktą. Taip pat prisijungęs vartotojas gali pasirinkti įrašų valdymą, kuriame gali pridėti, redaguoti arba ištrinti įrašą. Įrašo pridėjimui reikia pasirinkti jo tipą, įvesti reikšmę, vertimus ir patvirtinti. Tinkamai suvedus duomenis, vartotojui rodomas įrašų valdymas. Prisijungęs vartotojas gali valdyti kalbos vertimą, t. y. sukurti ir redaguoti jas. Abiem atvejais reikia įvesti kalbos pavadinimą ir kalbos kodą, o suvedus duomenis ir patvirtinus, kalba atnaujinama arba sukuriama ir rodomas kalbų valdymas. Vartotojas taip pat turi galimybe rankiniu būdu sinchronizuoti įrašus, o po sėkmingos sinchronizacijos vartotojas gali pasirinkti norimą veiksmą.

Su sistema sąveikaujančio įskiepio kūrimas

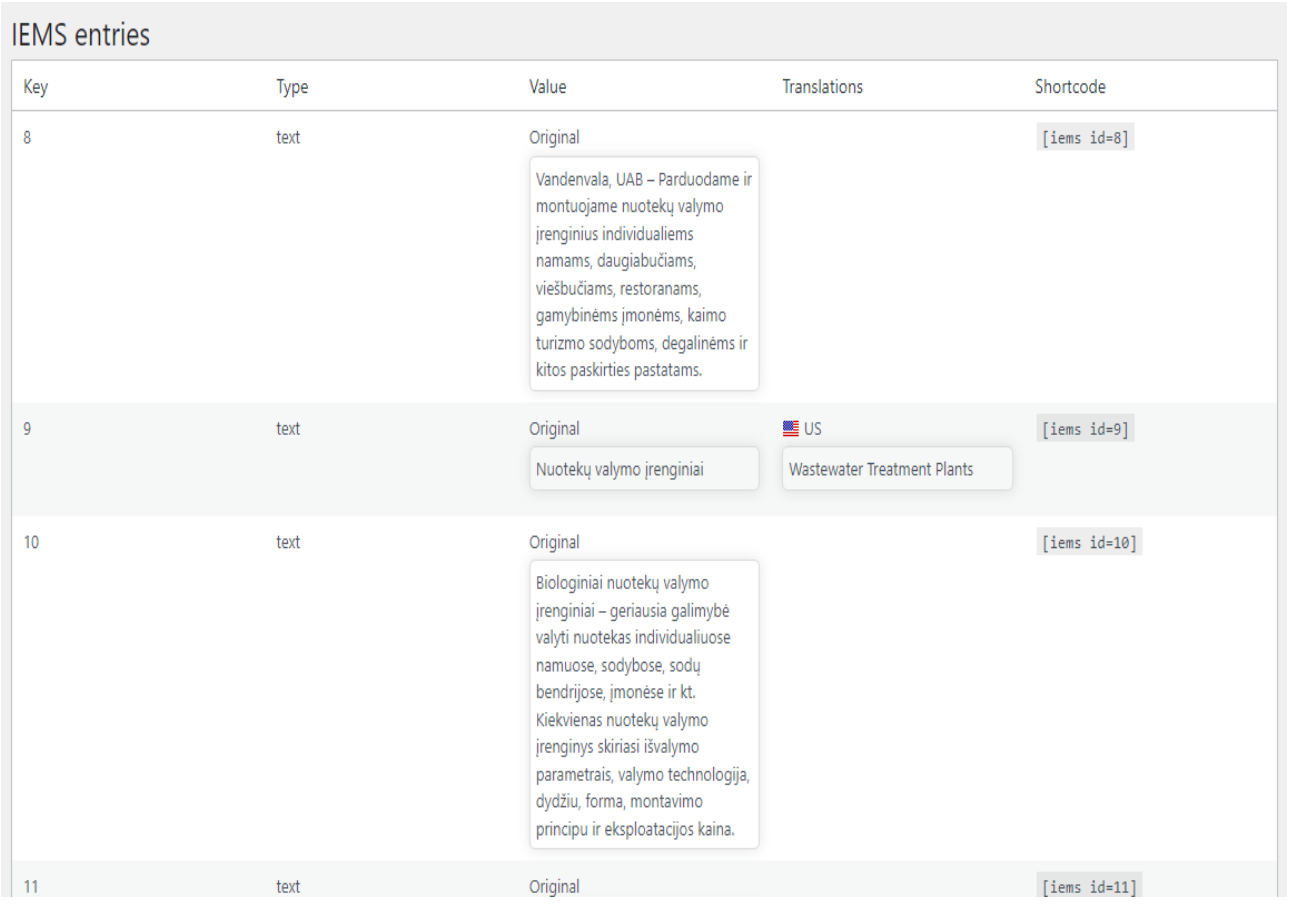
Kad sistema veiktų pilnaverčiai, jai buvo sukurtas su šia sistema suderinamas Wordpress įskiepis komunikavimui tarp svetainės ir sukurtos centralizuotos duomenų valdymo sistemos (žr. 3 pav.). Šis papildinys praplečia Wordpress turinio valdymo sistemos galimybes, sukurdamas papildomą svetainės sąsajos galutinį tašką bei galimybę naudoti sistemos automatiškai sugeneruotą kodą, kuris svetainės vykdymo metu yra konvertuojamas į reikšmę, nurodytą centralizuotoje duomenų valdymo sistemoje.



3 pav. Su sistema sąveikaujantis įskiepis

Wordpress turinio valdymo sistemoje įjungus šį įskiepį, jis kreipiasi į numatytą sistemos galutinį taško adresą, kartu perduodant tos svetainės pavadinimą ir jos adresą. Kai įskiepis yra įjungtas, jis gali apdoroti iš sistemos siunčiamą informaciją. Įskiepis apdoroja JSON formatu gautą informaciją ir ją išsaugo svetainės duomenų bazėje tolimesniam naudojimui.

Sugeneruotas įrašų trumpasis kodas, kurį galima rasti centralizuotoje duomenų valdymo sistemoje bei įskiepio valdyme (žr. 4 pav.), talpinamas bet kokioje svetainės vietoje, kurioje yra galimybė įdėti tekstą (žr. 5 pav.). Svetainės vykdymo metu įskiepis apdoroja įterptus trumpuosius kodus ir rodo to įrašo reikšmę pagal esamą kliento lokalę. Įrašui yra nurodytas jo identifikacijos raktas, pagal kurį yra surandama reikšmė duomenų bazėje.



4 pav. Įrašų sąrašas įskiepio valdyme

5 pav. Sistemos sugeneruoto trumpojo kodo įterpimas į pasirinktą svetainės vietą

Sukurtas centralizuotos duomenų valdymo sistemos įskiepis (modelis) leis administratoriui reikiamose puslapio vietose panaudoti sistemos sugeneruotus trumpuosius kodus, o įrašų vertimams nebebus reikalinga juos valdyti atskirai, kadangi visa tai apdoroja įskiepis pagal sistemoje suvestus duomenis ir kliento pasirinktą kalbą. Šis įskiepis yra pritaikytas tik WordPress turinio valdymo sistemai, tačiau, atsiradus poreikiui, bus galimybė sukurti ir kitoms turinio valdymo sistemoms, kadangi iš sistemos siunčiamos užklausos su įrašų duomenimis yra universalios (siunčiamos JSON formatu), o jų duomenis galima realizuoti bet kokioje programavimo kalboje.

Išvados

Išanalizavus įmonėje naudojamų interneto svetainių struktūrą ir įmonės poreikį, nuspręsta sukurti centralizuotą duomenų valdymo modelį. Šios sistemos pagalba turi būti galima valdyti įmonės puslapiuose esančią informaciją, suvesti įrašų norimos kalbos vertimus su automatinio vertimo funkcionalumu, reikalingas blokų redaktorius tam kad būtų galima pridėti paveikslėlius, lenteles, teksto formatavimą. Taip pat sistema turi turėti funkcionalumą pridėti papildomus vartotojus, jos siunčiama informacija į įskiepį turi būti kešuojama, o dizainas turi būti paprastas ir aiškus. Aprašytos priemonės projekto įgyvendinimui.

Išanalizavus įmonės svetainių struktūrą nustatyta problema, kad įmonės internetines svetaines šiuo metu administruoja tik įmonės IT administratorius, o dėl puslapių kiekio, informacijos besikartojimo bei skirtingų kalbų šis procesas užtrunka daug laiko ir ne visada ji būna pakeičiama visose vietose. To priežastis – naudojama turinio valdymo sistema, reikalaujanti techninių žinių, informacijos reikia ieškoti vizualiai rankiniu būdu daugybėje vietų, todėl ieškota sprendimo, kaip išspręsti šią problemą.

Atlikta panašių sistemų analizė, jai buvo parinktos 3 atsitiktinės panašaus tipo WP valdymo sistemos. Išanalizavus sistemas buvo pastebėta, kiekviena iš jų turi pagrindines funkcijas, tokias kaip atnaujinimų valdymas, įskiepių valdymas, saugumo analizės ir kitos, tačiau nei viena iš jų neturi pagrindinių įmonės poreikių atitinkančių funkcijų – valdyti atskirus įrašus, jų vertimus bei integruoti sistemą su skirtingomis turinio valdymo sistemomis.

Įvertinus panašias sistemas ir įmonės iškeltą unikalią problemą, buvo suprojektuotas bei sukurtas centralizuota duomenų valdymo sistemos modelis. Su šia sistema galima pridėti ir redaguoti įrašus, kalbas, svetaines, galima valdyti įrašų vertimus, sinchronizuoti juos su kiekviena pridėta ir aktyvuota svetaine.

Sukurtam sistemos modeliui buvo sukurtas su šia sistema suderinamas Wordpress įskiepis komunikavimui tarp svetainės ir sukurtos centralizuotos duomenų valdymo sistemos, kurio pagalba turinio valdymo sistemoje atsirado galutinis taškas, į kurį gali kreiptis sistema duomenų perdavimui.

Šis įskiepis leidžia administratoriui reikiamose puslapio vietose panaudoti sistemos sugeneruotus trumpuosius kodus, o įrašų vertimams nebebus reikalinga juos valdyti atskirai.

Literatūra

1. Chua, B. B., Dyson, L. E. Applying the ISO 9126 model to the evaluation of an e-learning system. *Proc. of ASCILITE*, 5(8), 2004, 184–190. https://ascilite.org/conferences/perth04/procs/pdf/chua.pdf
2. *PHP frameworks comparison*. 2023. Social Compare. https://socialcompare.com/en/comparison/php-frameworks-comparison
3. *5 Key Benefits of Centralized Data Management*. 2018. ZE Perspective. https://blog.ze.com/data-management-solution/5-key-benefits-of-centralized-data-management/
4. Apers, P. M. G. Data allocation in distributed database systems. *ACM Transactions on Database Systems (TODS)*, 13(3), 1988, 263–304. https://doi.org/10.1145/44498.45063
5. Babu, Ch., Gunasingh, G. DESH: Database evaluation system with hibernate ORM framework. In *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2016. p. 2549–2556. https://ieeexplore.ieee.org/abstract/document/7732441
6. Lerdorf, R., Tatroe, K., MacIntyre, P. *Programming PHP*. 2nd Edition. O'Reilly Media, Inc., 2006.
7. Brotherton, C. *The Most Popular PHP Frameworks to Use in 2023*. Kinsta, 2023. https://kinsta.com/blog/php-frameworks/
8. *What is Centralized Database? Functions, Advantages & Disadvantages*. 2021. Database Town. https://databasetown.com/centralized-database-functions-advantages/
9. Wilton, P. *Beginning JavaScript*. Second Edition. John Wiley & Sons, 2004.
10. Patni, S. *Pro RESTful APIs*. Apress, 2017.
11. *The MIT License*. 2006. Open Source Initiative. https://opensource.org/license/mit/
12. Shariff, K. A., et al. Generating UML Diagram Using Natural Language Processing and Use Case Diagram. *International Journal of Research in Engineering, Science and Management*, 2(1), 2019, 32–35. https://www.ijresm.com/Vol.2_2019/Vol2_Iss1_January19/IJRESM_V2_I1_8.pdf
13. Daugirdas, D., Vileikis, D. Projection and implementation of a model for car rental system. *Applied Scientific Research*, 1(1), 2022, 123–131. https://doi.org/10.56131/tmt.2022.1.1.47
14. Nayak, A., Samanta, D. Synthesis of test scenarios using UML activity diagrams. *Software & Systems Modeling*, 10, 2011, 63–89. https://doi.org/10.1007/s10270-009-0133-4
15. Ruby, S., Amundsen, M., Richardson, L. RESTful Web APIs. 2013. Sebastopol, CA: O'Reilly Media.
16. Mell, P., Grance, T. *The NIST Definition of Cloud Computing*. Recommendations of the National Institute of Standards and Technology. 2011. Special Publication 800–145. https://faculty.winthrop.edu/domanm/csci411/Handouts/NIST.pdf
17. Cheng, B., Zhao, S., Qian, J., Zhai, Z., & Chen, J. Lightweight service mashup middleware with REST style architecture for IoT applications. *IEEE Transactions on Network and Service Management*, 15(3), 2018, 1063–1075.
18. Newman, S. *Building microservices*. *Designing Fine-Grained Systems*. Second Edition. (2021). Sebastopol, CA: O'Reilly Media.
19. Ivanchikj, A., Pautasso, C., & Schreier, S. Visual modeling of RESTful conversations with RESTalk. *Software & Systems Modeling*, 17, 2018, 1031–1051.
20. Laaziri, M., Benmoussa, K., Khoulji, S., Larbi, K. M., & El Yamami, A. A comparative study of laravel and symfony PHP frameworks. *International Journal of Electrical and Computer Engineering*, 9(4), 2019, 704–712. http://doi.org/10.11591/ijece.v9i1.pp704-712

21. Kuflewski, K., & Dzieńkowski, M. Symfony and Laravel – a comparative analysis of PHP programming frameworks. *Journal of Computer Sciences Institute*, 21, 2021, 367–372. https://doi.org/10.35784/jcsi.2749
22. Bankov, B. Software Evaluation of PHP MVC Web Applications. In *International Multidisciplinary Scientific GeoConference-SGEM*, 2019, p. 603–610. DOI: 10.5593/sgem2019/2.1/S07.079
23. Wardana, A. K., Rianto, R., & Fauzi, E. R. Applying CodeIgniter framework on JHS website development. In *AIP Conference Proceedings*, 2023, June, 2491(1). AIP Publishing. https://doi.org/10.1063/5.0105694
24. Kirsan, A. S., Arisa, N. N., & Insanittaqwa, V. F. Improved access speed of the Codeigniter framework and REST APIs for the implementation of SIAKAD: Academic information system in Balikpapan schools. In *AIP Conference Proceedings*, 2023, March, 2508(1). AIP Publishing. https://doi.org/10.1063/5.0116859