BDG-V.2610.37.2018.PC

Załącznik nr 1 do SOPZ
(cz. I zamówienia)

Serwis *.biznes.gov.pl

OPIS SYSTEMU I ARCHITEKTURA

Spis treści

Słownik znaczeń

Nazwy systemów, podsystemów i aplikacji:

Akademia – platforma e-learningowa dla użytkowników wraz z treściami multimedialnymi w postaci kursów i poradników.

eLF - platforma usług elektronicznych służąca konfigurowaniu transakcyjnych publicznych usług online, ich testowaniu i ostatecznie produkcyjnemu udostępnianiu użytkownikom końcowym. Platforma składa się z dwóch podstawowych modułów: Konfigurator eLF i Runtime eLF.

Głos przedsiębiorcy – system do zgłaszania pomysłów przez obywateli dot. uproszczeń w prowadzeniu działalności gospodarczej.

Portal - system informatyczny klasy CMS udostępniający informacje dla użytkowników w ramach Systemu.

SOZ – system zgłaszania zgłoszeń do obsługi błędów, niepoprawności działania i pomysłów na usprawnienie systemu.

Styleguide – zbiór makiet, wzorców graficznych, praktyk użyty do tworzenia stron interfejsów graficznych dla użytkowników systemu we wszystkich aplikacjach systemu.

System – serwis *.biznes.gov.pl.

Ogólna architektura systemu

Moduły serwisu *.biznes.gov.pl

Lp.	Nazwa	Środowiska	Technologia
1.	Portal	Produkcja Testowe Developerskie	PHP, Elixir, Racket, GOLANG, Elastic Search, API
2.	eLF	Produkcja Testowe Developerskie Quality&Assurance	konfigurator PHP, runtime Java, bazy dokumentowe Maria DB, silnik procesowy Camunda, bazy Couchbase, Mongo (runtime) oraz Arrango (dla konfiguratora)
3.	Styleguide	Produkcja	Java, HTML, CSS
4.	Zaplecze	Produkcja Testowe Developerskie Quality&Assurance	PHP, MariaDB, SphinxSearch, json
5.	Akademia	Produkcja Testowe Developerskie	PHP, moodle
6.	API		PHP, MariaDB, SphinxSearch, json
7.	Baza wiedzy	Produkcja Testowe Developerskie Quality&Assurance	PHP, MariaDB
8.	Głos Przedsiębiorcy	Produkcja Testowe Developerskie	frontend PHP, backend Java
9.	System Obsługi Zgłoszeń	Produkcja Testowe Developerskie	PHP
10.	Konto / Logowanie	Produkcja Testowe Developerskie Quality&Assurance	PHP, MariaDB, wykorzystuje SSO

Portal

Główne zadania

Autorski system CMS pozwalający na użycie wyszukiwarki treści wraz z własnymi algorytmami wyszukiwania. Portal jest zintegrowany z Profilem Zaufanym oraz umożliwia własną rejestrację użytkowników w systemie.

Zawiera moduł przekierowań do wniosków, komponent odpytuje API, serwer plików statycznych.

Użyte technologie

Użyte język programowania:

1. **PHP**

   Cel: budowa systemu zarządzania treścią

   a. użyte moduły common, bcmath, xml, json, mbstring, gd, pgsql
   b. użyte frameworki programistyczne: Laravel 5.7+, Symfony 4.1+, Zend 3+

2. Racket

   Cel: automatyzacja publikacji zmian wprowadzonych w głównym środowisku, zbudowany został mikro serwis, który jest wywoływany przy każdym git push'u na gałąź master na wewnętrznym repozytorium GIT. Jego zadaniem jest wydzielenie plików statycznych oraz ułożenie ich do odpowiednich katalogów zachowując ustandaryzowaną strukturę wersji (snapshot, current, release).

3. **GoLang** (wersja 1.10)

   Cel : budowa własnego serwera multimediów to wystawia API do pobierania / zapisywania plików oraz ich wersjonowania. Wrzucane są tutaj wszystkie pliki statyczne (pdf, jpg, mp4, docx) przez redaktorów z poziomu CMS portalu. Następnie całość jest wystawiana w trybie read-only na portal.

   a. Użyte frameworki: Gin – web framework dla GoLang

Serwer www: **Apache** (użyte jako proxy) oraz **NGiNX** (do wystawiania komponentów portalu).

Konfiguracja serwera www:

- ustawione proxy na serwerze Apache,
- FastCGI Process Manager for PHP działające na socketach.

Użyta wyszukiwarka treści

Elixir – użyty jako wyszukiwarka pełno kontekstowa w celu rankingowania treści zwracanych rezultatów z różnych źródeł jak API BW (Poradniki, Usługi) CMS portalu (Strony, Trudne terminy, Sprawy, Ulotki, Strony ulotek, Kategorie spraw).

Eastic służy jako serwer indeksujący treści wewnętrzne na potrzeby wyszukiwarki. Tj. Strony, Trudne terminy, Sprawy, Ulotki, Strony ulotek, Kategorie spraw. Dodatkowo wspomaga API Pozycyjne jako jedna z warstw middleware.

Przechowywanie plików statycznych

Duża część ujednoliconych plików jest przechowywana na static.biznes.gov.pl (są to pliki css, js, img wydzielone ze styleguide). Szablony css oraz pliki statyczne dostarczane są z adresu

static.biznes.gov.pl/[gałąź]/

Pliki statyczne wrzucane przez redaktorów obsługiwane są przez serwer multimedialny stojący pod domeną media.biznes.gov.pl. Nieliczne pliki serwowane są bezpośrednio z katalogu static portalu. Portal nie używa zewnętrznych CDN'ów.

Dynamiczna generacja podstron portalu

W zależności od poszukiwanej streści portal dostarcza treści z innych modułów Systemu:

- szablony e-usług: generowane dynamiczne przez środowisko eLF
- treści bazy wiedzy: treści renderowane są dynamicznie wewnątrz portalu. Używają styli oraz skryptów części frontowej portalu.

Bazy danych

- PostgreSQL 10,
    - Instalacja samodzielna, bez repliki i klastra.
- Redis w wersji 3, bez klastrów.

Udostępnione API

Portal wystawia 3 endpointy na potrzeby szablonów

- wysyłka poradnika na email
- wysłanie opinii
- pobranie poradnika na dyski

Źródła aplikacji portal

Do potrzeb audytu nie udostępniono źródeł aplikacji, skryptów instalacyjnych ani plików konfiguracyjnych.

Zarządzanie użytkownikami portalu

Panel administracyjny posiada tylko możliwość przypisywania ról. Każda z ról posiada określoną liczbę pozwoleń. Same pozwolenia są wbudowane w silnik oraz system szablonów widoków które nie można ich zmieniać dynamicznie.

eLF

Główne zadania

Platforma usług elektronicznych służąca konfigurowaniu transakcyjnych publicznych usług online umożliwiająca złożenie wniosków elektronicznych przez użytkowników (np. przedsiębiorców, osób planujących rozpoczęcie działalności gospodarczej).

Architektura komponentów

System mikro usług (ponad 120 usług działających w produkcji) zabudowywanych w modelu dziedzinowym, działających niezależnie od siebie i posiadające własne repozytoria danych (różne bazy zarówno **SQL** i jaki i **NoSQL**). Usługi do komunikacji używają wzorca architektonicznego **REST**.

Dostęp do usług z zewnątrz aplikacji oraz pomiędzy projektami obywa się poprzez komponenty API Gateway które pozwalają na dostęp do wybranych usług lub ich części. Rolę API Gateway pełni własny mikro usługa działająca na zasadzie nadzorcy filtrującego dostęp do mikro usług.

API mikro usług posiada wygenerowaną i opublikowaną dokumentację automatyczną w **OpenAPI** umożliwiającą testowanie.

Kontakty opublikowanych mikro usług (60 adresów) definiują zakres zachowania usług. Mikro usługi osadzone są w kontenerach Docker które umieszczane są w zarządzanym i monitorowanym środowisku.

Całość działa na platformie **RedHat OpenShift** wraz z oprogramowaniem **Kubernates** które zarządza, kontenerami **Docker** w których działają wytworzone w ramach projektu mikro usługi. Środowisko jest automatycznie optymalizowane i skalowane w zależności od obciążeń. Takie rozwiązanie umożliwia zarządzanie zmianami opublikowanych usług bez potrzeby zatrzymania czy restartu usług.

Proces biznesowy i obsługa użytkownika zaczyna się kiedy użytkownik zakończy wprowadzanie informacji do formatek i zdecyduje się wysłać wniosek do wybranego urzędu. Rozpoczyna się proces weryfikacji wniosku, podpisu, uzupełnienia danych pobranych ze słowników, generowanie załączników. Proces biznesowy zawiera w załączeniu wygenerowane dokumenty.

Interfejs użytkownika napisany jest w PHP z komponentami napisanymi w **Angular**. Strony użytkowni są generowane przez silnik generatora stron i podstron wraz mapowaniem (bindowaniem) do kontraktów usług.

Każda usługa składania wniosku udostępniona dla użytkownika składa się konfigurowalnej przez administratora definicji procesu przebiegu i mapowania zbierania danych od użytkownika. Podstrony zbierania danych zawierają wielokrotnie używalne kontrolki wprowadzania danych jak pola danych tekstowych, pola wyboru, listy wyboru, pola wartości wraz z możliwością walidacji wprowadzonych danych, podpowiedzi i autouzupełniania danych już wcześniej zebranych od użytkownika w innym procesie.

Do preparowania wywołania usług (kontaktów) z zebranych od użytkownika danych została stworzona specjalna aplikacja (konfigurator usług). Pozwala na dodanie logiki biznesowej do procesowanego przebiegu przetwarzania wniosku. Logika walidacji zbieranych danych obywa się po stronie użytkownika (wykonywane funkcje walidacyjne JavaScript po stronie front-endu klienta).

W końcowej fazie dane są zapisywane w formacie JSON i poddawane procesów i przetwarzania i przesłania dokumentów do instytucji do której dany wniosek powinien trafić.

Zbierane są metryki działania miko usług oraz kontenerów Docker. Do zbierania metryk użyto rozwiązania open-source **Prometheus**. Do wizualizacji działania usług użyto rozwiązania open-source **Grafana** pobierającego dane z aplikacji Prometheus.

Użyty silnik BPMN (**Camuda Comunity** – licencja open source) służy instrumentacji komunikacji stanowej z usługami zewnętrznymi np. w celu obsługi terminów administracyjnych złożonych wniosków.

Administratorzy monitorują działania mikro usług, repozytoriów danych, zużycie zasobów systemowych na poziomie platformy **OpenShift** (metryki działania platformy). Podejmują aktywne działania w zależności od wykorzystania zasobów reagując na przeciążenia zasobów i skalując poszczególne zasoby.

Usługi składania wniosków utworzone w eLF udostępnione są w aplikacji Baza Wiedzy.

Całość systemu eLF i Baza Wiedzy spięta jest SSO i nie wymaga od użytkownika ponownego logowania.

Logi działania komponentów odkładane są na systemie plików. Do obróbki logów i wizualizacji wykorzystywane jest narzędzie Graylog, Zipkin (np. do śledzenia czasów zapytań w ms).

Użyte technologie

Mikro usługi wykonane w różnych technologiach głównie w

- PHP (wersja 7)
- JavaScript
- Java (jdk1.8.0_172) – wykonano wiele rozdrobionych mikro usług w architekturze SOA

Konfigurator eLF

- PHP (wersja 7) wraz z frameworkiem Lumen

Do wykonania stron interfejsu użytkownika (konfigurator) wykorzystano języki

- PHP (wersja 7)
- Angular 7 (język TypeScript z translatorem automatycznym do JavaScript)
- JavaScript

Serwery www : NGINX, Apache, envoy (proxy), HAproxy

Serwery aplikacyjne: Tomcat, WildFly w różnych mikro serwisach

Bazy danych: ArangoDB (wersja 3.3), Couchbase, MongoDB (wersja 3.6.6), MariaDB (wersja 10.2.15)

Użyte metod dostępu do danych: JCR – SQL 2, SQL

Formatowanie treści: użyta biblioteka markdown-it wersja 8.4.2

Silnik procesów biznesowych: Camunda Community wraz z własnymi pluginami (History Plugin), zamodelowano 9 własnych biznes procesów, dostęp do silnika przez JavaAPI, RestAPI.

Należy zauważyć, iż wykonawca do generowania formularzy napisał własny, dynamiczny silnik korzystający z zapisanych głównie w formacie JSON konfiguracji stąd wynika jego duże użycie w projekcie.

Do generowania dokumentów w formacie Doc oraz XML użyto silnika szablonów **Freemaker** (wersja 2.3.27, licencja opensource).

Elastic search został użyty do umożliwienia wyświetlania tekstów i komunikatów użytkownikowi na interfejsie użytkownika zgodnie z zapisem i18.

Metryki kodu

Język programowania/skrypt konfiguracyjny	linii kodu	Jako procent całości
Bourne Shell	1420	0,47%
CSS	26	0,01%
Dokerfile	564	0,19%
DOS Batch	2	0,00%
Freemaker Tamplate	16	0,01%
GoLang	626	0,21%
Groovy	41	0,01%
HTML	23047	7,66%
Java	30595	10,17%
JavaScript	9312	3,10%
JSON	52466	17,44%
Markdown	1248	0,41%
Maven	2433	0,81%
PHP	62147	20,66%
Python	347	0,12%
RAML	1463	0,49%
RobotFremework	3247	1,08%
Saas	11863	3,94%
SQL	69	0,02%
TypeScript	69268	23,03%
XML	14780	4,91%
XSD	534	0,18%
XSLT	823	0,27%
YAML	14466	4,81%
Razem	300803	100%

Styleguide

Główne zadania

Określa jak poszczególne komponenty Bazy Wiedzy i eLF mają wyglądać w przeglądarce użytkownika.

Użyte technologie

Język programowania: Java (która wersje JRE, JDK)

Użyte bibiloteki: Servlet, Thymeleaf, Logback

Serwer aplikacji: Tomcat

System Obsługi Zgłoszeń

Główne zadania

Obsługa zgłoszeń o charakterze technicznym związanych z serwisem biznes.gov.pl oraz nadawanie praw dostępu do systemu dla innych podmiotów wskazanych przez Zamawiającego.

Kategorie spraw w SOZ:

a. Wsparcie techniczne serwisu biznes.gov.pl

b. Usterki/awarie serwisu biznes.gov.pl

c. Uwagi/propozycje dot. Serwisu biznes.gov.pl

d. Inne dot. Serwisu biznes.gov.pl

Zadania w formie zapytań są zgłaszane przez zainteresowanych pisemnie, na podstawie wypełnionego formularza internetowego lub w trakcie rozmowy z konsultantem Call Center. Jednym z zadań jest również rekonfiguracja SOZ umożliwiająca zapisywanie i odczytywanie zgłoszeń w formie multikanałowej (możliwość wglądu w zgłoszenia z innych kanałów kontaktu np. live chat – zapis rozmowy, telefon – odsłuch audio rozmowy) oraz udostępnienie ww. funkcjonalności pozostałym podmiotom realizującym zadania w SOZ.

Wykonawca zapewni narzędzia do pomiaru jakości obsługi, generowanie szczegółowych raportów pokazujących m.in realizację kluczowych KPI: np. FCR, SLA, abandon rate, answer time dla wszystkich kanałów komunikacji, zadowolenie Klienta mierzone ankietami we wszystkich kanałach. Dostawca przedstawi szczegółowo system pomiaru jakości pracy Zamawiającemu.

Użyte technologie

Na stronie portalu wklejony jest kod java script Inteliwise.com wywołujący okno formularza zgłoszenia.

Wykorzystanie technologii sztucznej inteligencji lub integracja z technologią AI wykorzystywaną dotychczas przez Zamawiającego do m.in. kategoryzowanie spraw zgłaszanych do SOZ (samouczący się chatbot) poprzez wdrożenie systemu rozpoznawania mowy (pisma) naturalnego w celu kierowania zgłoszeń na podstawie treści zgłoszenia i rozwoju kanałów komunikacji.

Akademia

Główne zadania

Platforma e-learningowa przeznaczona do prowadzenia szkoleń i warsztatów w trybie on-line. Udostępnia wideoporadniki i webinary (treści multimedialne), ścieżki szkoleniowe dla użytkowników oraz testy wiedzy użytkownika.

System zawiera także forum użytkowników. System Akademia zrealizowana jako osobna aplikacja z wykorzystaniem systemu Moodle wraz z integracją logowania przez Profil Zaufany i Węzeł Krajowy.

Oprócz ścieżek szkoleniowych i forum na platformie zamieszczone zostały kursy e-learning w formacie SCORM 1.2.

Użyte technologie

Do zarządzania udostępnioną treścią został użyty system Moodle w wersji 3.1.6 wraz z komponentami integracji logowania przez SAML.

Użyte moduły Moodle:

- Certificate module (https://docs.moodle.org/22/en/Certificate_module)
    - Tworzenie certyfikatów poświadczeń w formacie PDF
- Forumqta (https://github.com/cyberlearn/moodle-mod_forumqta)
    - Zadawanie pytań, pisanie odpowiedzi, komentowanie, sortowanie, tagowanie
- Questionnaire (https://moodle.org/plugins/mod_questionnaire)
    - Tworzenie dynamicznych ankiet dla użytkowników

Metryki

Język	linii kodu	procent całości
CSS	5013	49,50%
JavaScript	200	1,97%
PHP	4915	48,53%
Razem	10128	100%

Baza Wiedzy

Główne zadania

- Zarządzanie rejestracją użytkowników w serwisie biznes.gov.pl
- Zarządzanie procesami usług udostępnianych użytkownikom (przedsiębiorcy, urząd – reprezentacje instrukcji, administratorzy).

- Udostępnianie API do usług do komunikacji wewnętrznej zarówno wewnętrznym modułom serwisu biznes.gov.pl (np. portal, platforma e-usług) jak i zewnętrznym serwisom
- Wyszukiwanie treści znajdujących się w bazie wiedzy
- Udostępnianie artykułów jak poradniki, wzorów dokumentów, publikacji, opinii, aktów prawnych, tłumaczeń dokumentów
- Umożliwienie umieszczenie opinii do opublikowanych artykułów przez użytkowników i zarządzanie opiniami
- Generowanie raportów ze stanu informacji znajdujących się w Bazie Wiedzy
- Wysyłka konfigurowalnych powiadomień mailowego do użytkowników
- Zasilanie danych z zewnętrznych źródeł (adresy pocztowe, PKD, Tefryt dane adresowe – podstawowa baza podziału terytorialnego)

Użyte technologie

Back-end

- PHP v5.3.29,
- REST API (JSON)

Front-end

- PHP v5.3.29,
- CakePHP v2.4,
- Bootstrap 3 i 4,
- jQuery 3, CSS v2 i v3, SASS

Autoryzacja usług zewnętrznych

- SAML 2.0, SimpleSAMLphp v1.14

Indeksowanie i wyszukiwanie treści znajdujących się w Bazie Wiedzy

- Sphinx v2.2.10

Wysyłka maili do użytkowników

- Postfix 2.11, Courier

Ad-hoc tworzenie raportów

- jQuery QueryBuilder 2.4.1

Bazy danych

W aplikacji użyto bazy danych MariaDB w wersji 10.1 działającej w trybie samodzielnym bez klastrowania i replikacji. Planowane jest zrobienie replikacji danych w modelu master-slave.

W bazie jest utworzonych ok. 70 tabel. Na produkcji rozmiar bazy danych 6 GB. Największy rozmiar 400.000 rekordów ma tabela zawierającej dane z historii zmian publikowanych artykułów.

Środowisko uruchomieniowe

System w środowisku testowym i produkcyjnym działa na systemie wirtualizacyjnym VMware vSphere wersja 6.5.

API

Baza Wiedzy korzysta w własnego API wykorzystującego wzorzec architektoniczny REST. Dane zapytań przesyłane są metodą GET i POST. Odpowiedzi zwracane są w formacie JSON.

Do zabezpieczenia użycia API wykorzystano zabezpieczenie poprzez wprowadzanie listy dozwolonych hostów klienckich (white-lista) i filtrowanie pakietów http serwera udostępniającego API.

Poszczególne API mogą być udostępniane dla klientów zewnętrznych przez API Geteway o ile zachodzi taka potrzeba.

Liczba usług API: 54

Metryki kodów

Język, konfiguracja	linii kodu	procent całości
Ant	549	0,08%
Bourne Again Shell	57	0,01%
Bourne Shell	113	0,02%
CSS	87074	12,12%
DOS Batch	77	0,01%
HTML	3966	0,55%
JavaScript	136396	18,99%
PHP	480700	66,92%
Sass	257	0,04%
SQL	7686	1,07%
XML	996	0,14%
YAML	410	0,06%
Razem	718281	100%

Testy

Testowanie odbywa się automatyczne poprzez bezpośrednie wywołania wystawionych interfejsów API na środowisku testowym. Testowanie interfejsu użytkownika wspomagane jest przez automat testujący Selenium.

Zgłaszanie i obsługa błędów

Błędy od strony Zamawiającego zgłaszane są do systemu Redmine.

Metryki błędów

				Priorytet zgłoszenia			
miesiąc/obszar	liczba zgłoszeń	niski	średni	wysoki	pilny	natychmiastowy	średni czas obsługi (dni)
październik 2018							
BW	16	1	6	6	2	1	23
eLF	17	0	4	5	8	0	8
listopad 2018							
BW	18	0	10	7	1	0	12
eLF	15	0	7	1	4	3	11

Repozytorium kodów

Do zarządzania repozytoriami użyto systemu GitLab w wersji Community (bezpłatnej, open-source https://gitlab.com/gitlab-org/gitlab-ce). Repozytoria kodów projektów są podzielone logicznie z możliwością przypisywania użytkowników do poszczególnych repozytoriów.

Testowanie kodu obywa się na osobnym serwerze przez wydzielony zespół sprawdzania jakosci (ang. QA - Quality Assurance).

Minimalizowane są wielkości plików JavaScipt i szablonów CSS za pomocą automatycznych narzędzi optymalizujących (usuwanie spacji, tabulatorów, wcięć etc).

W celu prowadzenia projektu powstały 4 środowiska:

- deweloperskie
- QA testy wewnętrzne, testowanie regresji i sprawdzanie jakości wykonanych zmian (w jednej chmurze ze środowiskiem deweloperskim)
- testowe dla potrzeb testów końcowych zamawiającego
- produkcyjne

Testowane są także interfejsy użytkownika w przypadku komponentów wykonanych w technologii Angular są przeprowadzane automatyczne testy komponentów.

We wdrażaniu zmian w kodzie wykosztowane są praktyki ciągłej integracji (ang. Continuous Integration) i ciągłego dostarczana (ang. Continuous Delivery) do której wykorzystano narzędzia open-souce **Git (GitLab)**, **Jenkins**.

Infrastruktura

Infrastruktury fizycznej z podziałem na środowiska i aplikacje.

Systemy produkcyjne

LP	HOSTNAME	RAM	CPU (ilość core)	Dysk	OS	Przeznaczenie/ co jest zainstalowane
1	prod-knowledge-www-centos7	32 GB	16	100 GB	Centos 7	Serwer WWW: httpd.x86_64 2.4.6
2	prod-knowledge-db-centos7	32 GB	16	90 GB	Centos 7	Serwer baz danych: MariaDB-server 10.1.24
3	prod-elearning-www	4 GB	2	70 GB	Centos 7	Serwer WWW: httpd.x86_64 2.4.6, moodle 3.1
4	prod-elearning-db-centos7	4 GB	4	50GB	Centos 7	Serwer baz danych: MariaDB-server 10.1.24
5	prod-elf-master.prod-elf.epk.local	64 GB	16	210 GB	Centos 7	Serwer Openshift Master: Openshift v3.9.0 Kubernetes v1.9.1
6	prod-elf-n1.prod-elf.epk.local	64 GB	16	180 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
7	prod-elf-n2.prod-elf.epk.local	64 GB	16	180 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
8	prod-elf-n3.prod-elf.epk.local	64 GB	16	180 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
9	prod-elf-n4.prod-elf.epk.local	64 GB	16	180 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
10	prod-elf-n5.prod-elf.epk.local	64 GB	16	120 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
11	prod-elf-n6.prod-elf.epk.local	64 GB	16	120 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
12	prod-elf-n7.prod-elf.epk.local	64 GB	16	120 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
13	prod-elf-konfigurator-db1	24 GB	8	110 GB	Centos 7	Serwer baz danych konfiguratora: Docker: CassandraDB
14	prod-elf-konfigurator-db2	24 GB	8	110 GB	Centos 7	Serwer baz danych konfiguratora: Docker: CassandraDB
15	prod-elf-konfigurator-db3	24 GB	8	110 GB	Centos 7	Serwer baz danych konfiguratora: Docker: CassandraDB

16	prod-elf-mariadb-01	32 GB	8	150 GB	Centos 7	Serwer baz danych Elf: MariaDB-server 10.2.17
17	prod-elf-mariadb-02	32 GB	8	150 GB	Centos 7	Serwer baz danych Elf: MariaDB-server 10.2.17
18	prod-elf-mariadb-03	32 GB	8	150 GB	Centos 7	Serwer baz danych Elf: MariaDB-server 10.2.17
19	prod-elf-db	16 GB	8	150 GB	Centos 7	Serwer baz danych elf: Docker: MongoDB, Couchbase
20	prod-elf-db-n1	16 GB	1	60 GB	Centos 7	Serwer baz danych elf: Docker: MongoDB, Couchbase
21	prod-elf-db-n2	16 GB	1	60 GB	Centos 7	Serwer baz danych elf: Docker: MongoDB, Couchbase
22	prod-elf-db-n3	16 GB	1	60 GB	Centos 7	Serwer baz danych elf: Docker: MongoDB, Couchbase
23	DB	6 GB	2	30	Centos7	Baza danych PostgreSQL
24	Elastic	3 GB	2	26	Centos 7	Elastic i API Pozycjonujące
25	Biznes	6 GB	1	50	Centos 7	Portal
26	Media	2 GB	1	10	Centos 7	Media i static
27	Log	3 GB	1	30	Centos 7	Logstash

Środowiska testowe

LP	HOSTNAME	RAM	CPU (ilość core)	Dysk	OS	Przeznaczenie/ co jest zainstalowane
1	test-knowledge-www-centos7	32 GB	8	80 GB	Centos 7	Serwer WWW: httpd.x86_64 2.4.6
2	test-knowledge-db-centos7	16 GB	4	45GB	Centos 7	Serwer baz danych: MariaDB-server 10.1.24
3	test-elearning-www	4 GB	2	70 GB	Centos 7	Serwer WWW: httpd.x86_64 2.4.6, moodle 3.1
4	test-elearning-db	4 GB	2	50 GB	Centos 7	Serwer baz danych: MariaDB-server 10.1.24
5	test-elf-master.test-elf.epk.local	32 GB	8	220 GB	Centos 7	Serwer Openshift Master: Openshift v3.9.0 Kubernetes v1.9.1
6	test-elf-n1.test-elf.epk.local	32 GB	8	210 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
7	test-elf-n2.test-elf.epk.local	32 GB	8	210 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
8	test-elf-n3.test-elf.epk.local	32 GB	8	210 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
9	test-elf-n4.test-elf.epk.local	32 GB	8	210 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
10	test-elf-n5.test-elf.epk.local	32 GB	8	150 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
11	test-elf-n6.test-elf.epk.local	32 GB	8	150 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
12	test-elf-n7.test-elf.epk.local	32 GB	8	150 GB	Centos 7	Serwer Openshift Node: Openshift v3.9.0 Kubernetes v1.9.1
13	test-elf-konfigurator-db1	24 GB	8	200 GB	Centos 7	Serwer baz danych konfiguratora: Docker: CassandraDB
14	test-elf-konfigurator-db2	24 GB	8	200 GB	Centos 7	Serwer baz danych konfiguratora: Docker: CassandraDB
15	test-elf-konfigurator-db3	24 GB	8	200 GB	Centos 7	Serwer baz danych konfiguratora: Docker: CassandraDB
16	test-elf-mariadb-01	32 GB	8	30 GB	Centos 7	Serwer baz danych Elf: MariaDB-server 10.2.17
17	test-elf-mariadb-02	32 GB	8	30 GB	Centos 7	Serwer baz danych Elf: MariaDB-server 10.2.17

18	test-elf-mariadb-03	32 GB	8	30 GB	Centos 7	Serwer baz danych Elf: MariaDB-server 10.2.17
19	test-elf-db	16 GB	8	70 GB	Centos 7	Serwer baz danych elf: Docker: MongoDB, Couchbase
20	test-elf-db-n1	8 GB	1	60 GB	Centos 7	Serwer baz danych elf: Docker: MongoDB, Couchbase
21	test-elf-db-n2	8 GB	1	60 GB	Centos 7	Serwer baz danych elf: Docker: MongoDB, Couchbase
22	test-elf-db-n3	8 GB	1	60 GB	Centos 7	Serwer baz danych elf: Docker: MongoDB, Couchbase
23	Biznes	6 GB	1	50 GB	Centos 7	Portal test