INSTYTUT BIOLOGII SSAKÓW PAN,

17-230 BIAŁOWIEŻA, UL. STOCZEK 1, tel. 085 6827750, fax 085 6827752

Przetarg nieograniczony na usługi programistyczne w zakresie stworzenia systemu repozytoryjnego dla danych naukowych – Open Forest Data, ZP IBS PAN 07/2019

Załącznik nr 5.

I. Informacje ogólne

1. Przedmiotem zamówienia jest wykonanie usług programistycznych w zakresie projektu „e-Puszcza. Podlaskie cyfrowe repozytorium przyrodniczych danych naukowych" w ramach Programu Operacyjnego Polska Cyfrowa na lata 2014-2020, polegających na budowie kompleksowej platformy repozytoryjnej Open Forest Data (openforestdata.pl) służącej do gromadzenia, przechowywania i udostępniania danych naukowych w oparciu o oprogramowanie *open source* Dataverse oraz stworzenia dwóch aplikacji mobilnych.

2. W ramach zamówienia będą realizowane prace w następującym zakresie: analizy, projektowania, wytworzenia i modyfikacji aplikacji; instalacji oraz konfiguracji aplikacji i systemów teleinformatycznych, w środowisku wirtualnym będącym w posiadaniu Zamawiającego; wytwarzania i aktualizacji dokumentacji technicznej aplikacji; usuwania błędów w istniejących i modyfikowanych aplikacjach; testów zmodyfikowanych aplikacji i systemów teleinformatycznych; utrzymania systemu i asysty technicznej po dacie odbioru prac.

3. W ramach realizacji prac odbędzie się minimum 3 spotkania Zamawiającego z Wykonawcą, w celu omówienia projektowanych funkcjonalności, opracowania szaty graficznej oraz omówienia dotychczasowych postępów prac oraz zaplanowanie kolejnych etapów prac związanych z przedmiotem zamówienia. Wykonawca będzie pozostawał w regularnej komunikacji niezależnie od fizycznych spotkań.

4. Poniższe zapisy są wymogami minimalnymi, Wykonawca zobowiązuje się wykonać w pełni funkcjonalny system, z uwzględnieniem kwestii niezbędnych do prawidłowego działania systemu repozytoryjnego, włącznie z tymi, które nie zostały zamieszczone w poniższej specyfikacji. Dokładny sposób działania poszczególnych funkcjonalności oraz konkretne rozwiązania zostaną doprecyzowane na etapie projektu funkcjonalnego. Wybrane docelowo rozwiązania mogą się różnić od opisanych poniżej, z zastrzeżeniem, że nie mogą obniżyć funkcjonalności całego systemu ani wpływać na koszt realizacji zamówienia i ostateczną wartość oferty Wykonawcy.

5. Zamawiający w załączniku nr 7 przedstawia typowe rodzaje zasobów, które będą przechowywane w repozytorium.

6. Zamawiający w załączniku nr 8 przedstawia harmonogram prac.

7. Platforma będzie się składała z modularnych systemów opensource'owych: Dataverse (https://dataverse.org), Geonode (http://geonode.org/), Trapper (https://trapper-project.org/) oraz Grafana (https://grafana.com/), które zostaną zintegrowane poprzez protokoły API.

8. Stworzona zostanie interaktywna aplikacja internetowa służąca do agregacji i atrakcyjnej wizualizacji danych (zwana dalej agregatorem) oraz strona internetowa projektu. Zamówienie obejmuje także wytworzenie dwóch aplikacji mobilnych.

9. Stworzony system musi spełniać wymogi Krajowych Ram Interoperacyjności oraz realizować dobre praktyki udostępniania otwartych danych naukowych zgodnych z

zaleceniami FAIR i OpenAire Guidelines.

9.1. System musi:

9.2. zapewniać interfejs przyjazny użytkownikowi, zgodny z zasadami UX/UI;

9.3. zapewniać infrastrukturę umożliwiającą maszynowy odczyt danych;

9.4. nadawać zasobom unikatowe identyfikatory DOI;

9.5. spełniać wymagania WCAG 2.0 z uwzględnieniem poziomu AA;

9.6. zapewniać możliwość komunikacji z aplikacją poprzez protokół API;

9.7. zapewniać możliwość pobierania i gromadzenia metadanych poprzez protokoły OAI-PMH;

9.8. analiza konieczności użycia słownika DCAT i w razie konieczności jego zaimplementowanie;

9.9. Tworzyć i eksportować metadane w plikach RDF, schemacie XML, w formatach Dublin Core i Darwin Core;

9.10. uwzględniać techniki Responsive Web Design;

9.11. oferować przyjazne adresy URL wszystkich stron;

9.12. zapewniać możliwość nadawania licencji Creative Commons oraz możliwość różnych ustawień prywatności zasobów (opcja już istniejąca w Dataverse);

9.13. obsługiwać różne formaty danych, m.in. JSON, XML, RDF, CSV, TSV, ODF, SHP, GeoJSON, GML, KML, WKT, GPKG, PDF, XLS, XLSX, DOC, DOCX, PNG, JPG, TIF, GeoTIFF, STL, PLY, OBJ.

10. W ramach realizacji usługi opracowany zostanie pełen zestaw testów (unit tests) i pełna dokumentacja techniczna systemu. Niedozwolona jest modyfikacja bazowego kodu oprogramowania Dataverse, Geonode oraz Grafana, chyba że modyfikacje te zostaną zgłoszone i zaakceptowane w oficjalnych repozytoriach tych projektów.

II. Opis głównych systemów o otwartym kodzie źródłowym, które są integralną częścią platformy repozytoryjnej

1. DATAVERSE – jest bazo-danową, interaktywną aplikacją sieciową o otwartym kodzie źródłowym wspierającą współdzielenie, przechowywanie, cytowanie, eksplorację i analizę danych naukowych. DATAVERSE jest wykorzystywany w projekcie POPC „e-Puszcza. Podlaskie cyfrowe repozytorium przyrodniczych danych naukowych" do organizacji, zarządzania, przechowywania i współdzielenia danych przyrodniczych i naukowych na podstawie zdigitalizowanych kolekcji będących w posiadaniu instytucji: Instytutu Biologii Ssaków Polskiej Akademii Nauk i Instytutu Nauk Leśnych Politechniki Białostockiej. Kod źródłowy aplikacji DATAVERSE, dostępny na otwartej licencji Apache v.2, znajduje się w otwartym repozytorium github pod adresem: https://github.com/IQSS/dataverse. Dokumentacja aktualnej wersji aplikacji DATAVERSE znajduje się pod adresem: http://guides.dataverse.org/en/latest/.

2. GEONODE – jest dedykowanym w projekcie geoprzestrzennym systemem zarządzania treścią, internetową platformą służącą do zarządzania i publikacji danych przestrzennych. Wspiera dane wektorowe (w formatach shapefile, json, csv, kml i kmz) oraz rastrowe w formacie GeoTIFF. Metadane są dostępne poprzez serwis katalogowy OGC zaimplementowany w pycsw. W skład GEONODE wchodzą takie komponenty jak

GeoServer, GeoWebCache, OpenLayers, baza danych PostgreSQL z PostGIS. System GEONODE jest wykorzystywany do gromadzenia, udostępniania oraz wizualizacji danych o charakterze przestrzennym (GIS) w postaci zarówno wektorowej jak i rastrowej.

3. GRAFANA – jest dedykowanym w projekcie systemem służącym do analizowania oraz wizualizacji metryk i danych pomiarowych na dużą skalę, dostępnym przez przeglądarkę internetową. GRAFANA wykorzystuje dedykowaną bazę danych InfluxDB do gromadzenia metryk i danych pomiarowych, która jest specjalnie zaprojektowana pod kątem przechowywania i serwowania milionów danych metrycznych w postaci wartości powiązanych z czasem. GRAFANA umożliwi wizualizację historycznych wykresów w dowolnym zakresie czasu i analizę pod kątem trendów.
4. TRAPPER – jest bazo-danową, interaktywną aplikacją sieciową open-source wspierającą zarządzanie projektami foto-pułapkowymi, klasyfikację nagrań oraz współdzielenie i ponowne wykorzystanie zebranych danych. TRAPPER będzie wykorzystywany w repoytorium do organizacji, zarządzania i generowania informacji naukowej na podstawie zdjęć i filmów zwierząt zarejestrowanych przez foto-pułapki w zrealizowanych i przyszłych projektach badawczych i monitoringowych z wykorzystaniem foto-pułapek. Kod źródłowy aplikacji TRAPPER, dostępny na otwartej licencji GNU GPL v.3, znajduje się w otwartym repozytorium gitlab pod adresem: https://gitlab.com/oscf/trapper-project. Kod źródłowy, na bazie którego Wykonawca zrealizuje wymienione poniżej zadania znajduje się w branchu *development*. Dokumentacja aktualnej wersji aplikacji TRAPPER znajduje się pod adresem: https://trapper-project.readthedocs.io/en/latest/index.html.

III. Wymagania odnośnie technologii używanych podczas implementacji modułów w ramach opisanego przedmiotu zamówienia

1. Systemy operacyjne – wszystkie zaimplementowane moduły i komponenty muszą pracować na systemach operacyjnych z rodziny Linux, na dystrybucji Ubuntu 18.04 lub Centos 8
2. Serwery bazy danych SQL – wymagane jest wykorzystanie serwerów bazy danych działających w architekturze klient-serwer (możliwość współdzielenia przez wielu użytkowników jednocześnie) o otwartym kodzie źródłowym, które są dostępne w oficjalnych repozytoriach pakietów oprogramowania dla dystrybucji Ubuntu 18.04 lub Centos 8. Preferowane jest wykorzystanie serwera MySQL/MariaDB lub PostgreSQL.
3. Serwery WWW – do serwowania implementowanych aplikacji internetowych wymagane jest wykorzystanie serwerów o otwartym kodzie źródłowym, które są dostępne w oficjalnych repozytoriach pakietów oprogramowania dla dystrybucji Ubuntu 18.04 lub Centos 8. Preferowane jest wykorzystanie serwera nginx lub apache.
4. Framework do aplikacji webowych – wymagane jest wykorzystanie platformy programistycznej do wsparcia tworzenia aplikacji internetowych o otwartym kodzie źródłowym, który musi być uruchomiony na dystrybucji Ubuntu 18.04 lub Centos 8. Wymagane jest, aby aplikacja internetowa była pisania we wzorcu MVC (Model-Widok-Kontroler) lub pochodnych wzorcach.
5. Usługa katalogowa LDAP – baza danych przechowująca użytkowników, wymagane jest wykorzystanie serwerów LDAP działających w architekturze klient-serwer (możliwość współdzielenia przez wielu użytkowników jednocześnie) o otwartym kodzie źródłowym, które są dostępne w oficjalnych repozytoriach pakietów oprogramowania dla dystrybucji Ubuntu 18.04 lub Centos 8. Preferowane jest wykorzystanie serwera OpenLDAP.

IV. Opis zadań i potrzeb funkcjonalnych platformy repozytoryjnej Open Forest Data

1. INFORMACJE OGÓLNE

1.1. Realizacja zamówienia zakłada zaprojektowanie, implementację, wdrożenie oraz utrzymanie aplikacji internetowej działającej i kompatybilnej z przeglądarkami internetowymi w wersjach minimum: Google Chrome 78.0.3904.108, Mozilla Firefox 71, Microsoft Edge 44.18362.1.0, Opera 62. Aplikacja musi być dostosowana wizualnie do minimalnej wspieranej rozdzielczości ekranu 480x640 pikseli i kompatybilna dla przeglądarek mobilnych działających na smartfonach i tabletach pracujących na systemach w wersjach minimum Android 4.4 (KitKat) i IOS 7, także w trybie portretowym w wersjach minimum: Google Chrome 78.0.3904.84, Mozilla Firefox 68.3.0, Microsoft Edge 42.0.4.3892.

1.1.1. System nawigacji aplikacji musi zostać zaprojektowany zgodnie z aktualnymi zasadami tworzenia aplikacji internetowych. Nawigacja w aplikacji powinna być intuicyjna oraz ergonomiczna dla użytkownika, zapewniać łatwy dostęp do poszukiwanej treści, stanowić przejrzysty i zrozumiały system komunikacji.

1.1.2. Wszelkie treści tekstowe stanowiące zawartość w języku polskim, angielskim, rosyjskim i białoruskim przygotuje i dostarczy Zamawiający. Zaimplementowane przez Wykonawcę aplikacje internetowe i mobilne muszą wspierać mechanizm tłumaczenia poprzez zastosowanie plików słownikowych (opcja wspierana przez Dataverse i Geonode).

1.1.3. Wykonawca w ramach zaimplementowanych aplikacji stworzy panel/panele administracyjne do zarządzania i konfiguracji.

1.1.4. Rozwiązanie ma posiadać narzędzia do wprowadzania zmian w interfejsie umożliwiających szybką personalizację.

1.1.5. Dostarczone oprogramowanie będzie funkcjonowało w oparciu o dedykowany serwer aplikacji, z możliwością zwiększania wydajności poprzez dodanie kolejnych serwerów aplikacyjnych. Aplikacja internetowa musi wspierać obsługę mechanizmów do load balancingu. Dostarczone oprogramowanie ma działać z wykorzystaniem mechanizmu serwera pośredniczącego (reverse proxy).

1.1.6. Przykładowy wzór funkcjonalności aplikacji agregatora dostępny pod adresem: https://data.nhm.ac.uk/dataset/collection-specimens/resource/05ff2255-c38a-40c9-b657-4ccb55ab2feb

1.1.7. Moduł synchronizacji użytkowników: z uwagi na wykorzystanie trzech niezależnych systemów, które będą wykorzystane w projekcie należy zapewnić synchronizację kont lokalnych systemu Dataverse do serwera LDAP, z którego następnie korzystać będą m.in. system Geonode, Grafana, Trapper i moduł prezentacyjny w celu uwierzytelniania użytkowników.

1.1.8. Implementacja zewnętrznego narzędzia do obsługi oceniania automatycznego zasobów wg standardu „5 star data”. Ocena zasobów odbywa się na podstawie licencji, ustrukturyzowania danych, formatu pliku oraz formatu metadanych. Dalsze informacje znajdują się pod linkami: https://opendatahandbook.org/glossary/en/terms/five-stars-of-open-data/ , https://5stardata.info/en/. Dodatkowo wymagana jest opcja raportu liczebności danych/liczba gwiazdek w module agregatora.

2. Zaprojektowanie i wykonanie zagregowanego modułu prezentacyjnego dla danych

pochodzących z różnych źródeł systemu w postaci interaktywnej aplikacji sieciowej dostępnej przy użyciu przeglądarki, zwanej w skrócie „agregatorem”:

2.1. Aplikacja ma umożliwiać przeglądanie przechowywanych w systemie DATAVERSE obiektów z repozytorium, w tym zamieszczonych w plikach obrazowych (m.in. jpeg, tiff, skany 3D i skany z mikrotomografu) okazów z kolekcji przyrodniczych (obecnie około 180000 rekoródw) z wykorzystaniem widoków:

2.1.1. Widoku listowania rekordów i ich metadanych wraz z miniaturą podglądu zdjęcia i przyciskiem przejścia do widoku szczegółowego opisu obiektu oraz odnośnika do systemu przechowującego obiekt.

2.1.2. Widoku tabelarycznego rekordów i ich metadanych wraz z przyciskami umożliwiającymi podgląd zdjęcia oraz przejście do widoku szczegółowego opisu obiektu oraz odnośnika do systemu przechowującego obiekt.

2.1.3. Widoku galerii zdjęć wraz z przyciskami umożliwiającymi podgląd pełnego zdjęcia oraz przejście do widoku szczegółowego opisu obiektu.

2.1.4. Widoku interaktywnej mapy świata, z zaznaczonymi punktami współrzędnych geograficznych rekordów kolekcji, z uwzględnieniem timeline’u, z przyciskiem przejścia do wyświetlenia szczegółów danego obiektu, z możliwością wyszukiwania i filtrowania zasobów, z możliwością zawężania wyników do określonego obszaru mapy.

2.1.5. W każdym widoku odnośnik do danego rekordu kolekcji ma prowadzić do strony wyświetlającej pełne informacje o danym obiekcie kolekcji, wraz z możliwością wyeksportowania metadanych w formacie.

2.2. Aplikacja ma umożliwiać przeglądanie przechowywanych w systemie GEONODE warstw, map i dokumentów z wykorzystaniem widoków:

2.2.1. Widoku listowania rekordów i ich metadanych wraz z miniaturą podglądu obiektu i przyciskiem odnośnika do widoku szczegółowego opisu obiektu oraz odnośnika do systemu przechowującego obiekt;

2.2.2. Widoku tabelarycznego rekordów i ich metadanych wraz z przyciskami umożliwiającymi podglądu miniatury danego obiektu oraz przejście do widoku szczegółowego opisu obiektu oraz odnośnika do systemu przechowującego obiekt;

2.2.3. Widoku galerii miniatur obiektów wraz z przyciskami umożliwiającymi przejście do widoku szczegółowego opisu obiektu oraz odnośnika do systemu przechowującego obiekt;

2.2.4. W każdym widoku odnośnik do danego rekordu kolekcji ma prowadzić do strony wyświetlającej pełne informację o danym obiekcie kolekcji, wraz z możliwością wyeksportowania danych w formacie ISO 19115:2013.

2.3. Aplikacja ma umożliwiać przeglądanie przechowywanych w systemie GRAFANA wykresów danych metrycznych i pomiarowych z wykorzystaniem widoków:

2.3.1. Widoku listowania rekordów i ich tagów wraz z przyciskiem odnośnika do widoku szczegółowego z osadzonym wykresem poprzez mechanizm iframe oraz odnośnika do systemu przechowującego obiekt

2.3.2. Widoku tabelarycznego rekordów i ich tagów wraz z przyciskami umożliwiającymi podgląd wykresu oraz przejście do widoku szczegółowego opisu obiektu oraz odnośnika do systemu przechowującego obiekt

2.3.3. W każdym widoku odnośnik do danego rekordu kolekcji ma prowadzić strony

wyświetlającej pełne informację o danym obiekcie kolekcji.

2.4. Aplikacja ma umożliwiać zarówno proste wyszukiwanie rekordów z kolekcji przyrodniczej po słowach kluczowych w zakresie opisów bibliograficznych i metadanych rekordów jak i rozszerzone, m.in. na podstawie takich atrybutów jak: posiadanie zdjęcia, posiadanie współrzędnych geograficznych, filtrowania po dowolnym polu metadanych obiektu wraz z podpowiedziami istniejących wartości z całej kolekcji przyrodniczych dla tego pola, filtrowania po tagach obiektu wraz z podpowiedziami istniejących wartości z całej kolekcji przyrodniczych dla tego pola;

2.5. aplikacja ma umożliwiać dynamiczne filtrowanie oraz sortowanie rekordów w widoku tabelarycznym;

2.6. aplikacja ma umożliwiać zmianę układu wyświetlanych zdjęć w widoku galerii zdjęć, tj. układy 2x2, 3x3, 4x4;

2.7. panel administracyjny aplikacji ma umożliwiać konfigurację parametrów, atrybutów, pól metadanych, tagów oraz słów kluczowych, o jakie zbiory aplikacja internetowa ma odpytywać lub wykluczać w wynikach wyszukiwania. Aplikacja ma umożliwiać filtrowanie, jakie kontenery i zbiory danych z systemu Dataverse będą prezentowane module prezentacyjnym

2.8. aplikacja musi być napisana w architekturze klient-serwer i posiadać mechanizm asynchronicznego (AJAX) odpytywania serwera realizowanego przez javascript lub równoważne, przechodzenie do kolejnych wyników paginacji wyświetlanych rekordów jak również widoku podglądu szczegółowego obiektu nie może wymagać ponownego przeładowania strony internetowej i musi zapewniać dynamiczne wczytywanie rekordów;

2.9. aplikacja musi posiadać mechanizm uwierzytelniania się do systemu przy wykorzystaniu zewnętrznego serwera implementującego protokół LDAP. W pliku konfiguracyjnym musi znaleźć się możliwość włączenia tego trybu, jak również adresu serwera LDAP, portu, filtru wyszukiwania i atrybutu po którym aplikacja ma odpytywać serwer LDAP. Po włączeniu tej funkcjonalności, domyślnie aplikacja w pierwszej kolejności będzie uwierzytelniała użytkowników w oparciu o serwer LDAP, następnie w oparciu o lokalną bazę SQL.

2.10. aplikacja musi posiadać zaimplementowaną logikę, zgodnie z którą po poprawnym uwierzytelnieniu użytkownika w bazie LDAP i przy jednoczesnym jego braku w bazie lokalnej SQL, użytkownik zostanie stworzony w tej lokalnej bazie z domyślnymi najmniejszymi uprawnieniami.

2.11. aplikacja musi posiadać mechanizm nadawania uprawnień administracyjnych z poziomu panelu zarządzania aplikacją;

2.12. aplikacja musi posiadać mechanizm zapisywania i odczytywania z pamięci podręcznej powtarzających się oraz najczęstszych zapytań do API systemów Dataverse, Geonode i Grafana na poziomie aplikacji internetowej. Mechanizm ten ma na celu odciążenie tych systemów od wielokrotnego odpytywania API o powtarzające się zapytania. Aplikacja internetowa będzie musiała spełnić wymagania odnośnie wydajności, opisane w dalszej specyfikacji.

2.13. Aplikacja powinna wyświetlać ekran powitalny z regulaminem i checkboxem potwierdzającym akceptację regulaminu użytkowania aplikacji (oraz informacje o RODO) podczas jej pierwszego użycia;

2.14. Specyfikacja dostępnych funkcji API systemów znajdują się w poniższych odnośnikach:

2.14.1. Dataverse: http://guides.dataverse.org/en/latest/api/

2.14.2. Geonode: http://docs.geonode.org/en/2.8.1/reference/api.html

2.14.3. Grafana: https://grafana.com/docs/http_api/

3. Dostosowanie systemu DATAVERSE do potrzeb platformy repozytoryjnej obejmuje minimum:

3.1. Implementacja zewnętrznego narzędzia do obsługi wizualizacji obiektów 3D bezpośrednio w przeglądarce internetowej z wykorzystaniem plików skanu 3D minimum w formatach: 3ds, obj, stl. Integracja z repozytorium musi się odbyć przy wykorzystaniu mechanizmu zewnętrznych narzędzi systemu Dataverse (http://guides.dataverse.org/en/latest/admin/external-tools.html). Wizualizacja skanów 3D w przeglądarce nie może wymagać instalacji dodatkowych wtyczek do przeglądarki i być kompatybilna z przeglądarkami Google Chrome 78, Mozilla Firefox 71, Microsoft Edge 44, Opera 62.

3.2. Implementacja zewnętrznego narzędzia do obsługi wizualizacji plików TIFF, z uwzględnieniem funkcjonalności wyświetlania dużych plików (powyżej 50MB) w formacie TIFF bezpośrednio w przeglądarce bez konieczności pobrania całego pliku, jak również fragmentów plików TIFF poprzez mechanizm zoom.

3.3. Implementacja dodatkowych narzędzi (external tool), jeśli zajdzie taka potrzeba.

3.4. Opracowanie specyficznej dla projektu atrakcyjnej szaty graficznej systemu Dataverse, zaprojektowanej zgodnie z zasadami UX i identyfikacją graficzną Open Forest Data.

3.5. Implementacja i konfiguracja logowania OAuth poprzez konto ORCID (http://guides.dataverse.org/en/latest/installation/oauth2.html).

4. Dostosowanie systemu GEONODE do potrzeb platformy repozytoryjnej obejmuje minimum:

4.1. Personalizacja strony startowej systemu Geonode;

4.2. Opracowanie specyficznej dla projektu atrakcyjnej szaty graficznej systemu Geonode, zaprojektowanej zgodnie z zasadami UX i identyfikacją graficzną repozytorium.

4.3. Opracowanie systemu eksportu metadanych obiektów w Geonode do Dataverse – cykliczne odpytywanie API Geonode poprzez protokół OAI-PMH, tworzenie datasetów na podstawie harwestowanych metadanych, datasety w Dataverse zawierają odnośnik do zasobu w Geonode.

5. Import bieżących danych meteorologicznych do repozytorium.

5.1. Zapewnienie mechanizmu integracji danych eksportowanych na bieżąco ze stacji meteorologicznej z repozytorium – dane wysyłane są na serwer FTP w plikach txt co 10 minut.

5.2. Dane docelowo będą przechowywane w bazie InfluxDB systemu Grafana.

5.3. Opracowanie systemu eksportu metadanych obiektów z Grafany do Dataverse – cykliczne odpytywanie API Grafany, tworzenie datasetów na podstawie harwestowanych metadanych, datasety w Dataverse zawierają odnośnik do zasobu w Grafanie.

5.4. Wizualizacja danych meteorologicznych w Dataverse na osadzonych wykresach Grafany poprzez mechanizm iframe.

6. Stworzenie modułu wewnętrznej bazy danych kolekcyjnych.

6.1. Implementacja modułu – interfejsu graficznego wyświetlającego rekordy z kolekcyjnych baz danych SQL w postaci tabelarycznej, dostępnego dla użytkowników po zalogowaniu i przyznaniu dostępu do modułu. Użytkownicy będą mogli mieć różne poziomy dostępu

– READONLY lub READWRITE. Uwierzytelnianie będzie odbywało się przez konto LDAP projektu. Źródłem danych koniecznym do zaimportowania do bazy danych SQL będą istniejące pliki excel lub csv, z których największy ma 180 tysięcy rekordów i 35 kolumn. Na poziomie dostępu READWRITE użytkownik będzie miał możliwość dodawania i usuwania rekordów oraz ich edycji.

6.2. W module będzie znajdowała się wyszukiwarka, która umożliwi filtrowanie wszystkich wierszy i kolumn wraz z "LUB" oraz "I", a później wyświetlała wybrane rekordy w postaci tabelarycznej. Interfejs graficzny ma zapewniać możliwość sortowania wierszy oraz zmiany liczby rekordów wyświetlanych na stronie. Zaimplementowana zostanie możliwość eksportu wyfiltrowanych wierszy i kolumn do systemu Dataverse.

6.3. Moduł będzie zapisywał historię edycji baz danych wraz z użytkownikiem wprowadzającym zmianę. Zapewniona zostanie możliwość przywrócenia wcześniejszych wersji baz danych.

7. Dostosowanie systemu TRAPPER do potrzeb platformy repozytoryjnej.

7.1. Aktualizacja komponentów bazowych aplikacji TRAPPER.

7.1.1. aktualizacja części kodu źródłowego aplikacji napisanego w języku Python do wersji Python 3.7, Django 2.2 oraz Django Rest Framework 3.9 lub nowszych stabilnych

7.1.2. aktualizacja wszystkich komponentów aplikacji i kodu źródłowego w celu zapewnienia kompatybilności z najnowszymi stabilnymi wersjami nginx, RabbitMQ, PostgreSQL, PostGIS oraz Celery

7.1.3. aktualizacja wszystkich komponentów frontendu, w tym m.in. Angular.js i Boostrap do najnowszych stabilnych wersji; lista wszystkich bibliotek, które zostaną zaktualizowane przez Wykonawcę znajduje się pod tym adresem: https://trapper-project.readthedocs.io/en/latest/technologies.html#frontend

7.1.4. aktualizacja komponentu mapowego aplikacji TRAPPER (aplikacja django geomap) w celu zapewnienia kompatybilności zarówno części backendowej jak i frontendowej z najnowszą wersją aplikacji webGIS'owej UMAP (https://github.com/umap-project/umap)

7.1.5. aktualizacja wszystkich pozostałych dodatkowych pakietów zewnętrznych (3rd-party) Python i Django w celu zapewnienia kompatybilności z wyżej wymienionymi aktualizacjami

7.1.6. po wykonaniu wszystkich wymienionych powyżej aktualizacji aplikacja TRAPPER musi zachować dotychczasową funkcjonalność

7.2. Implementacja uniwersalnego mechanizmu eksportu wyników projektów klasyfikacyjnych z aplikacji Trapper.

7.2.1. implementacja logiki uniwersalnego mechanizmu eksportu wyników projektów klasyfikacyjnych w wielu formatach

7.2.2. zadanie dotyczy modelu Classification w aplikacji django media_classification oraz powiązanych modeli, w tym ClassificationProject oraz Classificator, a także modeli Deployment oraz Location w aplikacji django geomap

7.2.3. w aktualnej wersji aplikacji TRAPPER istnieje mechanizm eksportu wyników projektu klasyfikacyjnego do pliku zip zawierającego tabele z klasyfikacjami i deploymentami oraz dodatkowy plik z metadanymi w formacie EML (*Ecologcial Metadata Language*)

7.2.4. mechanizm ten zaimplementowany jest w widoku: https://$host/media_classification/classification/export/

7.2.5. dodatkowo istnieje API generujące tabele z klasyfikacjami oraz deploymentami: https://$host/media_classification/api/classifications/results/$pk/ https://$host/geomap/api/deployments/export/

7.2.6. Wykonawca uogólni i rozszerzy ten mechanizm aktualizując istniejące i implementując nowe punkty końcowe API oraz aktualizując istniejące widoki.

7.2.7. Metadane będą generowane zarówno na poziomie opisującym sam projekt klasyfikacyjny (ClassificationProject) oraz nadrzędny projekt badawczy (ResearchProject), w ramach którego klasyfikacje zostały wykonane, jak i na poziomie opisującym strukturę samych obiektów klasyfikacji i deploymentów, w tym lista i opis atrybutów klasyfikacji.

7.2.8. Wykonawca zaimplementuje mechanizm eksportu metadanych w następujących formatach:

7.2.8.1. EML (Ecological Metadata language) https://knb.ecoinformatics.org/external//emlparser/docs/index.html (funkcjonalność częściowo zaimplementowana w aktualnej wersji aplikacji TRAPPER)

7.2.8.2. Dublin Core / Darwin Core https://dwc.tdwg.org/terms/

7.2.8.3. Open Camera Trap Metadata Standard https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5267527/

7.2.8.4. Zenodo https://zenodo.org/schemas/records/record-v1.0.0.json

7.2.8.5. DATAVERSE http://guides.dataverse.org/en/latest/api/native-api.html

7.2.9. Wykonawca umożliwi eksport wyników klasyfikacji (model media_classification.Classification) i powiązanych z nimi deploymentów (model geomap.Deployment) i lokalizacji (model geomap.Location) w następujących formatach:

7.2.9.1. Natywnym dla aplikacji TRAPPER (zaimplementowany w aktualnej wersji) https://$host/media_classification/api/classifications/results/$pk/

7.2.9.2. camtrapR https://cran.r-project.org/web/packages/camtrapR/index.html

7.2.9.3. Wildlife Insights https://github.com/ConservationInternational/Wildlife-Insights----Data-Migration https://docs.google.com/spreadsheets/d/1PE5Zl-HUG4Zt0PwSfj-gJRJVbZ__LgH3VuiDW3-BKQg/edit#gid=1190405668

7.2.9.4. DATAVERSE http://guides.dataverse.org/en/latest/admin/metadatacustomization.html http://guides.dataverse.org/en/latest/api/native-api.html

7.2.9.5. Eksport do opisanych powyżej formatów metadanych oraz wyników klasyfikacji i powiązanych z nimi deploymentów i lokalizacji zaimplementowany zostanie w postacie nowych lub zaktualizowanych punktów końcowych API. Dodatkowo zaktualizowany zostanie widok https://$host/media_classification/classification/export/, który będzie

umożliwiać użytkownikowi wybór formatów plików, które znajdą się w wygenerowanej paczce zip z danymi.

7.3. Integracja aplikacji TRAPPER z repozytorium danych DATAVERSE.

7.3.1. Wykonawca zaimplementuje logikę oraz widoki eksportu wyników projektów klasyfikacyjnych do repozytorium DATAVERSE bezpośrednio z poziomu aplikacji TRAPPER.

7.3.2. w szczególności Wykonawca zaimplementuje logikę tworzenia z poziomu aplikacji TRAPPER takich obiektów DATAVERSE jak dataverse, dataset oraz datafile wraz z powiązanymi z nimi metadanymi opisującymi eksportowany projekt badawczy i klasyfikacyjny oraz tabele z danymi (klasyfikacje, deploymenty i lokalizacje).

7.3.3. rozszerzona zostanie aplikacja django accounts w celu umożliwienia użytkownikowi aplikacji TRAPPER, w widoku profilu (https://$host/accounts/profile/), konfiguracji korespondującego konta w DATAVERSE tj. takich parametrów jak host, token, username oraz innych niezbędnych do prawidłowego działania implementowanej funkcjonalności.

7.3.4. Wykonawca zaimplementuje mechanizm referencji do fizycznych zasobów (tj. filmów i zdjęć z foto-pułapek) przechowywanych w systemie TRAPPER bezpośrednio z poziomu DATAVERSE, który umożliwiać będzie autoryzowany dostęp do kolekcji multimediów powiązanych z danymi i metadanymi zarchiwizowanymi w repozytorium DATAVERSE.

7.3.5. w celu realizacji powyższej funkcjonalności Wykonawca zaproponuje i zaimplementuje mechanizm synchronizacji systemów autoryzacji w aplikacjach TRAPPER i DATAVERSE.

7.4. Wdrożenie systemu wielojęzyczności w aplikacji TRAPPER:

7.4.1. wykorzystanie mechanizmów dostępnych w django: https://docs.djangoproject.com/en/2.2/topics/i18n/translation/.

7.4.2. aktualizacja wszystkich fragmentów kodu źródłowego generującego informacje wyświetlane i wysyłane użytkownikom.

7.5. Integracja aplikacji TRAPPER z logowaniem po protokole LDAP

7.5.1. Wykonawca zaimplementuje mechanizm uwierzytelniania się do systemu TRAPPER przy wykorzystaniu zewnętrznego serwera implementującego protokół LDAP. W pliku konfiguracyjnym musi znaleźć się możliwość włączenia tego trybu, jak również adresu serwera LDAP, portu, filtru wyszukiwania i atrybutu po którym aplikacja ma odpytywać serwer LDAP. Po włączeniu tej funkcjonalności, domyślnie aplikacja TRAPPER w pierwszej kolejności będzie uwierzytelniała użytkowników w oparciu o serwer LDAP, następnie w oparciu o lokalną bazę SQL.

7.5.2. Wykonawca zaimplementuje możliwość wyłączenia panelu rejestracji użytkowników w systemie TRAPPER.

7.5.3. Wykonawca wykona testy funkcjonalne zaimplementowanej funkcjonalności uwierzytelniania minimum z serwerami OpenLDAP i ActiveDirectory wskazanymi przez Zamawiającego.

7.5.4. Wykonawca zaimplementuje logikę, zgodnie z którą po poprawnym uwierzytelnieniu użytkownika w bazie LDAP i przy jednoczesnym jego braku w bazie lokalnej SQL, użytkownik zostanie stworzony w tej lokalnej bazie z domyślnymi najmniejszymi uprawnieniami bez zapisania jego hasła.

7.5.5. Mechanizm integracji z serwerami LDAP służy jedynie do możliwości uwierzytelniania użytkowników, zarządzanie uprawnieniami i pozostałe funkcjonalności pozostają jak dotychczas bez zmian i będą wykonywane bezpośrednio w aplikacji TRAPPER.

7.6. Integracja aplikacji TRAPPER z mechanizmem SSO zgodnej ze standardem SAML 2.0

7.6.1. Wykonawca zaimplementuje mechanizm uwierzytelniania się do systemu TRAPPER przy wykorzystaniu zewnętrznego serwera dostawcy tożsamości Identity Provider (IdP). W pliku konfiguracyjnym musi znaleźć się możliwość włączenia tego trybu, jak również zdefiniowania Entity ID oraz wymaganych atrybutów. Po włączeniu tej funkcjonalności, na stronie logowania aplikacji TRAPPER powinna pojawić się możliwość wyboru jednego ze zdefiniowanych dostawców tożsamości IdP, po jego wyborze ma nastąpić przekierowanie na stronę dostawcy tożsamości, który po poprawnym uwierzytelnieniu i zatwierdzeniu przez użytkownika chęci przekazania wymaganych atrybutów zwróci je do aplikacji TRAPPER

7.6.2. Wykonawca zaimplementuje możliwość wyłączenia panelu rejestracji użytkowników w systemie TRAPPER.

7.6.3. Wykonawca wykona testy funkcjonalne zaimplementowanej funkcjonalności uwierzytelniania minimum z dostawcami tożsamości opartymi o oprogramowanie Shibboleth oraz SimpleSAMLPhp wskazanymi przez Zamawiającego.

7.6.4. Wykonawca zaimplementuje logikę, zgodnie z którą po poprawnym uwierzytelnieniu użytkownika przez dostawcę tożsamości IdP i przy jednoczesnym jego braku w bazie lokalnej SQL, użytkownik zostanie stworzony w tej lokalnej bazie z domyślnymi najmniejszymi uprawnieniami oraz z przekazanymi atrybutami wymaganymi oraz opcjonalnymi (zdefiniowanymi wcześniej w pliku konfiguracyjnym), jak imię, nazwisko, adres email itp. Dodatkowo po każdym logowaniu poprzez IdP, w przypadku zmiany tych atrybutów należy zaktualizować lokalną bazę SQL.

7.6.5. Mechanizm integracji z mechanizmem SSO zgodnym ze standardem SAML 2.0 służy jedynie do możliwości delegowania uwierzytelniania innym instytucjom, zarządzanie uprawnieniami i pozostałe funkcjonalności pozostają jak dotychczas bez zmian i będą wykonywane bezpośrednio w aplikacji TRAPPER.

8. Stworzenie strony internetowej projektu, zgodnie z poniższymi wymaganiami minimalnymi:

8.1. Strona internetowa projektu musi być oparta o system zarządzania treścią (CMS);

8.2. Wykorzystany system zarządzania treścią musi być o otwartym kodzie źródłowym;

8.3. Edycja strony internetowej przez użytkowników musi opierać się na mechanizmie WYSIWYG;

8.4. Strona internetowa musi implementować funkcjonalności związane z prezentacją informacji z repozytorium Dataverse, którą trzeba zaimplementować poprzez odpytywanie API tego systemu, w tym m.in. funkcje:

8.4.1. prezentacji statystyki ogólnej liczby zgromadzonych jak i pobranych w systemie zbiorów danych w podziale na dni, miesiące oraz lata pobieranego z API metryk;

8.4.2. prezentacji statystyki ogólnej liczby zgromadzonych jak i pobranych w systemie warstw, map i dokumentów w podziale na dni, miesiące oraz lata;

8.4.3. paska wyszukiwania, które będzie jednocześnie odnośnikiem do systemu Dataverse z wyszukanymi zbiorami danych na podstawie wprowadzonych słów

kluczowych;

8.4.4. prezentacji ostatnio dodanych zbiorów danych;

8.4.5. prezentacji kategorii danych zgromadzonych w systemie.

8.5. Strona internetowa musi implementować funkcjonalności związane z prezentacją informacji z repozytorium Geonode, którą trzeba zaimplementować poprzez odpytywanie API tego systemu, w tym m.in. funkcje:

8.5.1. paska wyszukiwania, który będzie jednocześnie odnośnikiem do systemu Geonode z wyszukanymi zbiorami danymi geoprzestrzennymi na podstawie wprowadzonych słów kluczowych, możliwość filtrowania wszystkich kategorii metadanych i zastosowanie filtrów przestrzennych;

8.5.2. prezentacji ostatnio dodanych warstw, map i dokumentów;

8.5.3. prezentacji kategorii danych zgromadzonych w systemie.

8.6. Strona internetowa musi implementować funkcjonalności związane z prezentacją informacji z systemu Grafana, którą trzeba zaimplementować poprzez odpytywanie API tego systemu, w tym m.in. funkcje:

8.6.1. paska wyszukiwania, które będzie jednocześnie odnośnikiem do systemu Grafana z wyszukanymi zbiorami danych na podstawie wprowadzonych słów kluczowych

8.7. Moduł blogu.

8.8. Integracja strony internetowej z Google Analytics.

8.9. Funkcjonalność wymiany informacji z użytkownikami w zakresie opinii i zgłaszania uwag dotyczących interfejsów jak i zakresu udostępnianych zbiorów cyfrowych – Formularz kontaktowy, kontakt z opiekunem/właścicielem obiektu/ wysłanie maila do opiekuna, zgłoszenie błędu/problemu.

8.10. Integracja z mediami społecznościowymi w zakresie funkcjonalności „Lubię to” i udostępniania aktualności.

8.11. Aktualności – Administrator serwera strony będzie miał możliwość utworzenia dowolnej liczby kategorii aktualności i wprowadzania treści artykułów z zaznaczeniem kategorii, w której ma być wyświetlana. Dla treści wprowadzonych przez administratora będzie istnieć możliwość ustalania kolejności wyświetlania wiadomości – standardowo, wg daty wprowadzenia – im nowsza tym wyżej oraz ręcznie oraz określenia statusu (opublikowany, nieopublikowany, archiwalny).

8.12. Moduł obiekt dnia – Będzie codziennie prezentować wybrany losowo obiekt z naukowej kolekcji przyrodniczej wraz z jego metadanymi.

V. Aplikacje mobilne

1. Aplikacja mobilna 1 – do zgłaszania obserwacji martwych zwierząt.

1.1. Celem aplikacji mobilnej jest promocja bezpieczeństwa, dostarczanie poradników postępowania w razie kolizji ze zwierzęciem lub znalezienia martwego zwierzęcia oraz promocja nauki obywatelskiej („citizen science”).

1.2. Aplikacja mobilna będzie służyła do zbierania obserwacji martwych zwierząt oraz będzie dostarczała różne instrukcje działania, w zależność od typu wprowadzonej obserwacji.

1.3. Teksty wyświetlane w aplikacji oraz identyfikacja wizualna projektu zostaną

dostarczone przez klienta.

1.4. Wymagania techniczne:

1.4.1. Aplikacja zostanie utworzona na platformę Android 4.4 – 9.X.

1.4.2. Aplikacja przeznaczona do użytku na smartfonach w orientacji portretowej.

1.4.3. Aplikacja będzie działała podczas połączenia z Internetem i offline.

1.4.4. Aplikacja zostanie opublikowana w sklepie Google Play na koncie klienta.

1.5. Funkcjonalności aplikacji:

1.5.1. Ekrany dostępne dla niezalogowanych użytkowników – wybór z menu:

1.5.1.1. Strona powitalna - informacje o aplikacji, opis projektu.

1.5.1.2. Poradnik, co zrobić, gdy znajdzie się martwe zwierzę - tekst dostarczony przez klienta.

1.5.1.3. Poradnik, co zrobić w razie kolizji (prawny) - tekst dostarczony przez klienta.

1.5.1.4. Podgląd mapy z obserwacjami zgłoszonymi w aplikacji – na mapie widoczne są pinezki odpowiadające miejscom obserwacji, po wyborze pinezki pojawia się informacja o czasie i rodzaju obserwacji oraz gatunku. Mapę można przesuwać, zwiększać, zmniejszać, filtrować obserwacje według zasięgu mapy, gatunku, rodzaju obserwacji.

1.5.1.5. Galeria-porównywarka wybranych gatunków z wizualizacją skanów czaszek (pliki stl dostarczone przez klienta) – widok galerii gatunków z możliwością przejścia do podstrony zawierającej opis gatunku oraz wizualizacje skanów 3D czaszek. Skany 3D można obracać, przybliżać, oddalać itp. Co więcej, zapewniona będzie opcja porównania skanu 3D ze zdjęciem zrobionym przez użytkownika, na przykład poprzez nałożenie zdjęcia i skanu w różnych stopniach przezroczystości.

1.5.1.6. Pomoc – instrukcja korzystania z aplikacji.

1.5.2. Rejestracja i logowanie użytkowników:

1.5.2.1. Możliwość logowania poprzez konta społecznościowe (Facebook, Google), konto LDAP systemu repozytoryjnego, jak również zarejestrowania nowego konta w aplikacji. W tym przypadku użytkownik przy rejestracji podaje adres e-mail (który służy też jako login), hasło (oraz potwierdzenia hasła), imię i nazwisko.

1.5.2.2. Na adres e-mail wysyłany jest mail potwierdzający rejestrację, z linkiem aktywacyjnym.

1.5.2.3. Możliwość zmiany hasła poprzez wysłanie na adres podany przy rejestracji linku do zmiany hasła – po otwarciu linku uruchamia się aplikacja z ekranem do wpisania nowego hasła.

1.5.3. Dodawanie obserwacji – tylko dla zalogowanych użytkowników:

1.5.3.1. Po wybraniu opcji dodania obserwacji, otwiera się aparat fotograficzny lub wybór zdjęcia z galerii.

1.5.3.2. Po dodaniu zdjęcia przejście do dodania opisu obserwacji. Kategorie opisu:

1.5.3.3. Gatunek – pytanie „czy wiesz, co to za gatunek?” – możliwość wyboru

„tak" lub „nie" – przy wyborze „tak" pojawia się formularz, gdzie można wpisać gatunek oraz pewność swojego oznaczenia.

1.5.3.4. Sposób śmierci/zdarzenie – zamknięta lista kategorii do wyboru.

1.5.3.5. Położenie obserwacji – automatyczne sczytywanie współrzędnych z położenia telefonu oraz możliwość przesunięcia tej lokalizacji na mapie.

1.5.3.6. Data i czas – automatycznie ustawia się aktualna data i godzina, możliwość zmiany tych wartości.

1.5.3.7. Po dodaniu opisu obserwacji pojawia się ekran instrukcją działania zależną od wprowadzonego gatunku lub/i zdarzenia (loga i krótkie komunikaty).

1.5.3.8. Użytkownik ma możliwość przejrzenia listy swoich obserwacji, wyboru i wyświetlenia obserwacji z podglądem zdjęcia i opisem oraz lokalizacji na mapie.

1.5.4. Uprawnienia administratora:

1.5.4.1. Możliwość weryfikacji obserwacji wprowadzonych przez użytkowników – różne poziomy weryfikacji.

1.5.4.2. Możliwość dodania alertu, który wysyła mail na podany adres po pojawieniu się w bazie danych obserwacji spełniającej zadane warunki (np. gatunek lub miejsce).

1.5.4.3. Możliwość usuwania obserwacji, użytkowników.

1.5.4.4. Możliwość eksportu bazy danych.

1.6. Strona internetowa aplikacji:

1.6.1. Strona internetowa umożliwia zalogowanie się na swoje konto z aplikacji mobilnej tym samym loginem i hasłem.

1.6.2. Podstrony:

1.6.2.1. Strona startowa – opis aplikacji.

1.6.2.2. Własne obserwacje: mapa z pinezkami, odpowiadającymi lokalizacjom obserwacji oraz lista własnych obserwacji z podglądem zdjęcia i opisami, możliwość filtrowania obserwacji według lokalizacji, zasięgu mapy, gatunku, rodzaju zdarzenia.

1.6.2.3. Wszystkie obserwacje: mapa z pinezkami, odpowiadającymi lokalizacjom obserwacji oraz lista własnych obserwacji z podglądem zdjęcia i opisami, możliwość filtrowania obserwacji według lokalizacji, zasięgu mapy, gatunku, rodzaju zdarzenia, użytkownika.

1.6.2.4. Konto użytkownika – możliwość zmiany danych i ich widoczności.

2. Aplikacja mobilna 2 – gatunki inwazyjne

2.1. Celem aplikacji mobilnej jest edukacja społeczeństwa na temat gatunków inwazyjnych oraz promocja nauki obywatelskiej („citizen science").

2.2. .Aplikacja mobilna będzie służyła do zbierania obserwacji gatunków inwazyjnych.

2.3. Teksty wyświetlane w aplikacji oraz identyfikacja wizualna projektu zostaną dostarczone przez klienta.

2.4. Wymagania techniczne:

2.4.1. Aplikacja zostanie utworzona na platformę Android 4.4 – 9.X.

2.4.2. Aplikacja przeznaczona do użytku na smartfonach w orientacji portretowej.

2.4.3. Aplikacja będzie działała podczas połączenia z Internetem i offline.

2.4.4. Aplikacja zostanie opublikowana w sklepie Google Play na koncie klienta.

2.5. Funkcjonalności aplikacji.

2.5.1. Ekrany dostępne dla niezalogowanych użytkowników – wybór z menu:

2.5.1.1. Galeria gatunków inwazyjnych – galeria zdjęć z podpisami – po kliknięciu w zdjęcie pojawia się ekran ze szczegółowym opisem gatunku.

2.5.1.2. Strona powitalna - informacje o aplikacji, opis projektu.

2.5.1.3. Pomoc – instrukcja korzystania z aplikacji.

2.5.1.4. Podgląd mapy z obserwacjami zgłoszonymi w aplikacji – na mapie widoczne są pinezki odpowiadające miejscom obserwacji, po wyborze pinezki pojawia się informacja o czasie i rodzaju obserwacji oraz gatunku. Mapa można przesuwać, zwiększać, zmniejszać, filtrować obserwacje według zasięgu mapy, gatunku, rodzaju obserwacji.

2.5.2. Rejestracja i logowanie użytkowników:

2.5.2.1. Możliwość logowania poprzez konta społecznościowe (Facebook, Google), konto LDAP systemu repozytoryjnego E-puszcza, jak również zarejestrowania nowego konta w aplikacji. W tym przypadku użytkownik przy rejestracji podaje adres e-mail (który służy też jako login), hasło (oraz potwierdzenia hasła), imię i nazwisko.

2.5.2.2. Na adres e-mail wysyłany jest mail potwierdzający rejestrację, z linkiem aktywacyjnym.

2.5.2.3. Możliwość zmiany hasła poprzez wysłanie na adres podany przy rejestracji linku do zmiany hasła – po otwarciu linku uruchamia się aplikacja z ekranem do wpisania nowego hasła.

2.5.3. Dodawanie obserwacji – tylko dla zalogowanych użytkowników:

2.5.3.1. Po wybraniu opcji dodania obserwacji, otwiera się aparat fotograficzny lub wybór zdjęcia z galerii.

2.5.3.2. Po dodaniu zdjęcia przejście do dodania opisu obserwacji. Kategorie opisu:

2.5.3.3. Gatunek – zamknięta lista gatunków do wyboru.

2.5.3.4. Położenie obserwacji – automatyczne sczytywanie współrzędnych z położenia telefonu oraz możliwość przesunięcia tej lokalizacji na mapie.

2.5.3.5. Data i czas – automatycznie ustawia się aktualna data i godzina, możliwość zmiany tych wartości.

2.5.3.6. Użytkownik ma możliwość przejrzenia listy swoich obserwacji, wyboru i wyświetlenia obserwacji z podglądem zdjęcia i opisem oraz lokalizacji na mapie.

2.5.4. Uprawnienia administratora:

2.5.4.1. Możliwość weryfikacji obserwacji wprowadzonych przez użytkowników – różne poziomy weryfikacji.

2.5.4.2. Możliwość dodania alertu, który wysyła mail na podany adres po

pojawieniu się w bazie danych obserwacji spełniającej zadane warunki (np. gatunek lub miejsce).

2.5.4.3. Możliwość usuwania obserwacji, użytkowników.

2.5.4.4. Możliwość eksportu bazy danych.

2.6. Strona internetowa aplikacji:

2.6.1. Strona internetowa umożliwia zalogowanie się na swoje konto z aplikacji mobilnej tym samym loginem i hasłem.

2.6.2. Podstrony:

2.6.2.1. Strona startowa – opis aplikacji.

2.6.2.2. Własne obserwacje: mapa z pinezkami, odpowiadającymi lokalizacjom obserwacji oraz lista własnych obserwacji z podglądem zdjęcia i opisami, możliwość filtrowania obserwacji według lokalizacji, zasięgu mapy, gatunku, rodzaju zdarzenia.

2.6.2.3. Wszystkie obserwacje: mapa z pinezkami, odpowiadającymi lokalizacjom obserwacji oraz lista własnych obserwacji z podglądem zdjęcia i opisami, możliwość filtrowania obserwacji według lokalizacji, zasięgu mapy, gatunku, rodzaju zdarzenia, użytkownika.

2.6.2.4. Konto użytkownika – możliwość zmiany danych i ich widoczności.

VI. Testy i optymalizacja wydajności aplikacji

1. Testy i optymalizacja wydajności modułu prezentacyjnego, zwanego w skrócie „agregatorem”

1.1. wykonanie testów obciążeniowych tj. zestawienie środowiska testowego oraz wykonanie testów,

1.2. przygotowanie raportu z analiz wydajności aplikacji,

1.3. optymalizacja jakości i organizacji kodu oraz optymalizacja wydajności aplikacji

1.4. Testy obciążeniowe wykonane zostaną przy uwzględnieniu następujących parametrów:

1.5. Min. 10 tysięcy zasobów rekordów kolekcji (mapowanych jako datasets) przechowywanych na min. 1 tysiącu katalogów (dataverses),

1.6. Min. 1 tysiąc zasobów rekordów kolekcji w systemie Geonode (mapowanych jako warstwy, dokumenty i mapy),

1.7. Min. 100 zasobów rekordów kolekcji w systemie Grafana (mapowanych jako wykresy)

1.8. Min. 100 użytkowników, przy czym każdy użytkownik ma dostęp do min. 100 zasobów,

1.9. Konfiguracja serwera do testów: 8 GB RAM Intel(R) Xeon(R) CPU E5-2430 v2 @ 2.50GHz (4 cores),

1.10. Oczekiwane średnie czasy reakcji dla wywołania wyświetlenia losowych 100 rekordów z modułu prezentacyjnego przy zadanym filtrze wyszukiwania odpytujących API systemów Dataverse, Geonode i Grafana, na poziomie testów wywołania API to max 200ms bez uwzględnienia czasu odpowiedzi API tych zewnętrznych systemów.

VII. Analiza kodu źródłowego TRAPPERa po aktualizacjach i

refaktoryzacja w celu optymalizacji wydajności

1. wykonanie testów obciążeniowych tj. zestawienie środowiska testowego oraz wykonanie testów

1.1. przygotowanie raportu z analiz wydajności aplikacji

1.2. optymalizacja jakości i organizacji kodu oraz optymalizacja wydajności aplikacji

1.3. analizie i refaktoryzacji poddany zostanie kod źródłowy związany z filtrowaniem obiektów w bazie danych (querysets) i systemem kontroli dostępu do zasobów (user permissions & roles)

1.4. gdziekolwiek będzie to możliwe preferowane jest upraszczanie kodu i usuwanie jego zbędnych fragmentów

1.5. Testy obciążeniowe wykonane zostaną przy uwzględnieniu następujących parametrów:

1.5.1. min. 20 milionów zasobów (model geomap.Resource) zarejestrowanych na min. 100 tysiącach lokalizacji (model storage.Location) podczas min. 200 tys. deploymentów (model geomap.Deployment)

1.5.2. max. ilość zasobów zarejestrowana podczas jednego deploymentu to 200

1.5.3. max. ilość deploymentów na jednej lokalizacji to 4

1.5.4. zasoby zorganizowane w min. 10 tysięcy kolekcji (model storage.Collection), przy czym jeden zasób może należeć do wielu kolekcji

1.5.5. min. 10 tysięcy użytkowników, przy czym każdy użytkownik ma dostęp do min. 100 tysięcy zasobów i min. 500 kolekcji

1.5.6. min. 100 zdefiniowanych projektów badawczych (model research.ResearchProject), przy czym każdy projekt badawczy zawiera min. 1 milion zasobów, min. 10 użytkowników w roli Admin, min. 1 tysiąc użytkowników w roli Expert oraz min. 1 tysiąc użytkowników w roli Collaborator

1.5.7. 2 zdefiniowane projekty klasyfikacyjne (model media_classification.ClassificationProject) na każdy projekt badawczy, po pół miliona zasobów przypisanych do każdego projektu klasyfikacyjnego; min. 10 użytkowników w roli Admin; pozostali użytkownicy powiązanego projektu badawczego mają rolę Collaborator w obu projektach klasyfikacyjnych oraz min. 2 miliony klasyfikacji (modele media_classification.UserClassification oraz media_classification.Classification) w ramach każdego z projektów

1.5.8. konfiguracja serwera do testów: 8 GB RAM Intel(R) Xeon(R) CPU E5-2430 v2 @ 2.50GHz (4 cores)

1.5.9. testy zostaną przeprowadzone dla 1000 użytkowników i dla każdej z 4 kategorii uprawnień: Superuser, Admin, Expert i Collaborator (w sumie cztery tysiące wywołań bazy danych dla danego testu)

1.5.10. Oczekiwane średnie czasy reakcji dla poszczególnych wywołań API podczas testów z wymienionymi powyżej parametrami:

1.5.10.1. storage/api/resources - max. 500ms

1.5.10.2. geomap/api/locations - max. 200ms

1.5.10.3. geomap/api/deployments - max. 200ms

1.5.10.4. media_classification/api/classifications?project=X - max. 300ms

VIII. Aktualizacja istniejących i implementacja brakujących testów jednostkowych kodu źródłowego aplikacji TRAPPER

1. aktualizacja istniejących i implementacja brakujących testów jednostkowych (*unit tests*) funkcjonalności backendu (*django*).

2. testami jednostkowymi pokryta zostanie pełna funkcjonalność następujących modułów *django aplikacji TRAPPER*: *accounts, storage, research, geomap, media_clasification, sendfile, messaging, common*.

IX. Opracowanie pełnej technicznej dokumentacji aplikacji

1. Opracowana zostanie techniczna dokumentacja wszystkich klas i funkcji w kodzie źródłowym, a kody źródłowe aplikacji internetowej będzie składował w repozytorium kodu platformie https://github.com. Cała dokumentacja techniczna powinna być opracowywana w języku angielskim. Zamawiający wymaga, aby dokumenty tworzone i modyfikowane w ramach realizacji Zlecenia charakteryzowały się wysoką jakością, właściwą dla profesjonalnego charakteru świadczonych usług, na którą będą miały wpływ, takie czynniki jak

1.1. struktura dokumentu, rozumiana jako podział danego dokumentu na rozdziały, podrozdziały i sekcje w czytelny i zrozumiały sposób;

1.2. sposób pisania rozumiany jako zachowanie spójnej struktury, formy i sposobu pisania dla poszczególnych dokumentów oraz fragmentów tego samego dokumentu;

1.3. poprawność ortograficzna, gramatyczna i stylistyczna dokumentów;

1.4. dokumentacja kodu źródłowego w języku Python wykonana zostanie w formacie reStructuredText (reST);

1.5. opracowanie pełnej dokumentacji technicznej w środowisku Sphinx (http://www.sphinx-doc.org/en/master/);

1.6. kompletność dokumentu – pełne przedstawienie omawianego problemu obejmujące całość z danego zakresu rozpatrywanego zagadnienia;

1.7. spójność i niesprzeczność dokumentu – zapewnienie wzajemnej zgodności pomiędzy wszystkimi rodzajami informacji umieszczonymi w dokumencie, brak logicznych sprzeczności pomiędzy informacjami zawartymi we wszystkich przekazanych dokumentach oraz fragmentach tego samego dokumentu;

1.8. aktualność – uwzględnienie w dokumencie bieżących czynników i uwarunkowań, w tym aktualnie istniejącej dokumentacji; zachowanie ogólnie przyjętych norm, standardów i kryteriów jakości;

2. Zamawiający wymaga, aby do wytworzonej w ramach realizacji przedmiotu zamówienia dokumentacji, stanowiącej utwór w rozumieniu ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz. U. z 2017r. poz. 880, z późn. zm.), Wykonawca przeniósł na Zamawiającego wszelkie autorskie prawa majątkowe oraz prawa do zezwalania na wykonywanie zależnych praw autorskich oraz przenoszenia praw na inne osoby wraz z prawem do dokonywania w nich zmian i wykonywania praw zależnych.

X. Pozostałe wymagania i gwarancje

1. Przedmiot zamówienia zostanie testowo uruchomiony na platformie sprzętowej (serwerach) Zamawiającego.

2. Wykonawca zagwarantuje, iż przedmiot zamówienia będzie:

2.1. wolny od wad prawnych i roszczeń osób trzecich;

2.2. wolny od mechanizmów blokujących jego funkcjonowanie.

3. Zamawiający wymaga udzielenia gwarancji na oferowany przedmiot zamówienia na warunkach jak niżej:

3.1. gwarancja min. 12 miesięcy,

3.2. gwarancja liczona jest od daty podpisania końcowego protokołu odbioru przez zamawiającego.

4. Wymagane warunki świadczenia serwisu gwarancyjnego:

4.1. czas reakcji serwisu gwarancyjnego w ciągu max. 1 dnia roboczego od momentu zgłoszenia,

4.2. czas naprawy od momentu zgłoszenia w ciągu max. 5 dni roboczych

4.3. W ramach gwarancji Wykonawca zobowiązuje się do:

4.3.1. nieodpłatnego usuwania wad

4.3.2. pomocy w analizie kodu źródłowego oraz dokumentacji systemu

4.3.3. udzielania administratorom wyjaśnień dotyczących użytkowania i eksploatacji wykonanego systemu

4.3.4. pomocy w optymalizacji pracy systemu, bezpośredniej diagnostyki w docelowej lokalizacji Zamawiającego, z wykorzystaniem metody zdalnego dostępu

5. Wymagania w zakresie sposobu wykonywania Zlecenia:

5.1. Zamawiający wymaga, aby Wykonawca realizował powierzone zadania z uwzględnieniem zasad zwinnych metodyk wytwórczych:

5.1.1. Zadania określone w Zleceniu są realizowane w sposób iteracyjny i przyrostowy w ramach Sprintów o wskazanej w postępowaniu długości (Sprint może trwać od 1 do 4 tygodni).

5.1.2. Każdy Sprint rozpoczyna się od ustalonego zakresu funkcjonalności lub prac do realizacji w danym Sprincie.

5.1.3. W przypadkach uzgodnionych z Zamawiającym i za jego zgodą określona część prac może przejść na kolejny Sprint.

5.1.4. Każdy Sprint kończy się dostarczeniem Zamawiającemu przez Wykonawcę produktu: funkcjonalności lub innego wyniku prac, którym może być gotowa konfiguracja, dokument, wynik testów itp., w zależności od przypisanego Zadania.

5.1.5. Zadania realizowane w ramach Zlecenia będą podlegały określonym w Umowie procedurom odbiorczym.

5.2. Wykonawca będzie korzystał z repozytorium projektu E-Puszcza znajdującego się na platformie https://github.com (dokładny adres zostanie podany po zawarciu umowy), w którym będzie realizował kolejne Sprinty związane z implementacją poszczególnych zadań. Wykonawca będzie systematycznie wysyłał zmiany w kodzie do powyższego repozytorium w celu archiwizacji prac.

6. Wymagania - styl kodowania.

6.1. styl kodowania obowiązujący Wykonawcę musi być zgodny ze standardem pep8 (https://www.python.org/dev/peps/pep-0008/) dla wszystkich fragmentów kodu

źródłowego w języku Python;

6.2. oryginalne fragmenty kodu źródłowego aplikacji TRAPPER w języku Python, które nie spełniają standardu pep8 zostaną poprawione przez Wykonawcę w celu uzyskania zgodności z tym standardem;

6.3. jeśli dotychczasową funkcjonalność systemu można po aktualizacji komponentów bazowych zrealizować bez pomocy zewnętrznych, dodatkowych bibliotek i/lub pakietów tj. funkcjonalność taka jest zaimplementowana w nowych wersjach komponentów bazowych, Wykonawca będzie wykorzystywał takie możliwości; przykładem może być typ pola w bazie danych Postgresql Hstore, który jest dostępny natywnie w wersji Django 2.2, nie ma więc potrzeby wykorzystania zewnętrznej aplikacji django django-hstore (https://github.com/djangonauts/django-hstore), która zapewnia taką funkcjonalność w aktualnej wersji systemu TRAPPER.