

Załącznik 4.1 Wymagania techniczne

1. Podstawowe założenia:

- 1.1. Rozwiązania będą zbudowane w taki sposób, aby stały się spójnym elementem oferty CloudFerro (dla klientów różnica między usługami własnymi, a nowymi ma być niezauważalna), w szczególności:
 - 1.1.1. produkty obowiązują wspólny Design System
 - 1.1.2. produkty korzystają z rozwiązań zespołu usług wspólnych (portal klienta, IAM, billing)
 - 1.1.3. produkty muszą zostać zrealizowane i zaimplementowane w ekosystemie technologicznym i organizacyjnym (zarówno rozwojowym oraz produkcyjnym) Zamawiającego
 - 1.1.4. pojedyncza instancja rozwiązania musi umożliwiać świadczenie usług w ramach różnych marek (brandów)
- 1.2. W celu zapewnienia skalowalności, bezpieczeństwa i łatwości utrzymania oczekujemy, że projektowane rozwiązanie będzie zgodne z aktualnymi dobrymi praktykami w zakresie inżynierii oprogramowania i architektury systemowej, w szczególności:
 - 1.2.1. podejście Cloud Native
 - 1.2.2. zgodność z zasadami DevSecOps
 - 1.2.3. przestrzeganie wzorca 12-Factor App
 - 1.2.4. podejście API-First oraz dokumentacja w standardzie OpenAPI.
- 1.3. Docelowo utrzymanie i dalszy rozwój może być realizowane wewnątrz w ramach CloudFerro lub zlecane zewnętrznemu dostawcy. W okresie przejściowym możliwe jest utrzymanie przez dostawcę, który dane rozwiązanie dostarczył, a później przejęcie do utrzymania i rozwoju w ramach CloudFerro. Przygotowane rozwiązanie wraz z dokumentacją musi zapewniać możliwość przekazania utrzymania i dalszego rozwoju w sposób płynny i ekonomiczny.
- 1.4. Pierwsza linia wsparcia dla zastosowania komercyjnego (po udostępnieniu klientom) będzie zawsze realizowana przez CloudFerro
- 1.5. Dostawca prześle Zamawiającemu pełne prawa autorskie do wszystkich utworów oraz do całości dokumentacji

2. Oprogramowanie open source

- 2.1. Zamawiający zezwala (a tam, gdzie to jest powszechne to zachęca) na stosowanie gotowych komponentów opartych o licencje open source, przy czym wszystkie proponowane komponenty open source muszą zostać wymienione w ofercie wraz ze wskazaniem sposobu licencjonowania dla każdego komponentu.
- 2.2. Lista zatwierdzonych modeli licencji open source w CloudFerro:
 - 2.2.1. MIT
 - 2.2.2. Apache 2.0
 - 2.2.3. BSD
- 2.3. Dostawca może proponować komponenty open source oparte o licencję nie znajdującą się na powyższej liście, przy czym decyzja o akceptacji danego rozwiązania i modelu licencyjnego wymaga pisemnej akceptacji Zamawiającego.

3. Realizacja prac w ekosystemie technologicznym i organizacyjnym Zamawiającego

- 3.1. Dostawca zobowiązany jest do prowadzenia prac analitycznych, projektowych, programistycznych, testowych oraz operacyjnych w ekosystemie technologicznym i organizacyjnym Zamawiającego, obejmującym zarówno narzędzia i środowiska (development, test, produkcja), jak i stosowane procesy inżynierskie, standardy jakościowe, praktyki architektoniczne oraz zasady operacyjne.
- 3.2. Budowanie rozwiązań powinno odbywać się z wykorzystaniem udostępnionej przez Zamawiającego infrastruktury – m.in. repozytoriów kodu, platform CI/CD, narzędzi monitorujących i logujących, systemów zarządzania konfiguracją oraz narzędzi do dokumentacji i komunikacji – zgodnie z obowiązującymi procesami i praktykami organizacyjnymi oraz na zbliżonych zasadach dotyczących wewnętrznych zespołów Zamawiającego.

4. Uzasadnienie

- 4.1. Zamawiający prowadzi większość działań rozwojowych i operacyjnych we własnym zakresie, traktując to jako element strategicznej kontroli nad jakością i kierunkiem rozwoju produktów cyfrowych. Wyprodukowane rozwiązania będą docelowo utrzymywane i rozwijane wewnętrznie, dlatego kluczowe jest zapewnienie spójności z istniejącymi praktykami oraz łatwości ich przejęcia i dalszego rozwijania przez zespoły Zamawiającego.

- 4.2. W kolejnych etapach (poza zakresem zamówienie) możliwe jest prowadzenie prac rozwojowych i utrzymaniowych w zespołach mieszanych, złożonych z członków zespołów wewnętrznych i Dostawcy – co tym bardziej wymaga wspólnego środowiska, zrozumienia procesów i jednolitych standardów pracy.

5. Potencjalne zmiany w środowisku i procesach

- 5.1. Zamawiający stale rozwija swój ekosystem narzędziowy i organizacyjny, dostosowując go do rosnących potrzeb technologicznych i operacyjnych. W przypadku planowanych zmian, które mogą mieć wpływ na sposób pracy Dostawcy, zostanie on odpowiednio wcześniej poinformowany i włączony w proces uzgodnień. Celem jest zapewnienie ciągłości i komfortu pracy – zarówno dla Dostawcy, jak i dla zespołów Zamawiającego – przy zachowaniu wspólnego standardu działania.

6. Możliwość proponowania uzasadnionych odstępstw od wymagań

- 6.1. Zamawiający dopuszcza możliwość zgłaszania przez Dostawcę propozycji odstępstw na etapie składania ofert od poszczególnych wymagań pod warunkiem przedstawienia dla każdego z nich odrębnego, merytorycznego uzasadnienia.
- 6.2. Każde odstępstwo powinno zawierać:
- 6.2.1. jasne odniesienie do konkretnego wymagania, którego dotyczy,
 - 6.2.2. opis proponowanego rozwiązania alternatywnego,
 - 6.2.3. uzasadnienie techniczne, organizacyjne lub biznesowe,
 - 6.2.4. potencjalne skutki i konsekwencje wdrożenia odstępstwa (zarówno pozytywne, jak i ryzyka).
- 6.3. Wnioski o odstępstwa zostaną ocenione indywidualnie przez Zamawiającego, który zastrzega sobie prawo do ich akceptacji lub odrzucenia. Intencją Zamawiającego jest zapewnienie elastyczności i otwartości na lepsze lub bardziej adekwatne rozwiązania, przy jednoczesnym utrzymaniu spójności z istniejącym ekosystemem technologicznym i organizacyjnym.

- 6.4. UWAGA: Wycena oferty musi być oparta o wymagania Zamawiającego. Ewentualne wnioski o odstępstwa mogą zawierać informacje o wpływie na wycenę w sytuacji akceptacji odstępstwa.

7. Sposób prowadzenia prac

- 7.1. Prace powinny być prowadzone w sposób zapewniający transparentność postępów realizacji prac dla Zamawiającego.
- 7.2. Cyklicznie będą zorganizowane spotkania statusowe (MS Teams), gdzie Dostawca przedstawi:
- 7.2.1. bieżący postęp prac w stosunku do planu prac
 - 7.2.2. ryzyka i zidentyfikowane problemy wraz z planem naprawczym
 - 7.2.3. prace wykonane od ostatniego spotkania
 - 7.2.4. zaktualizowany plan prac
 - 7.2.5. demonstrację roboczej wersji produktu lub jego komponentów.
- 7.3. Dostawca musi utrzymywać aktualność statusów prac na Jirze Zamawiającego. Zamawiający przeszkoli Dostawcę z zasad stosowania Jiry w CloudFerro oraz udzieli niezbędnych dostępów.

8. Wymagania obszaru: Technology & Development

ID Wymagania	Opis Wymagania
DEV-NOFUNC-00010	Dostawca jest zobowiązany do zaprojektowania rozwiązania jako aplikacji natywnej dla chmury, zgodnie z zasadami 12-factor.
DEV-NOFUNC-00020	Dostarczone rozwiązanie musi być wdrażane na platformie Kubernetes Zamawiającego i w pełni zgodne z jej składnikami, takimi jak: kontrolery ruchu, Prometheus Operator, Cert-Manager, ExternalDNS oraz Logging Operator. Rozwiązanie musi: 1. Mieć zdefiniowane wartości resources.requests i resources.limits dla CPU i pamięci, aby zapewnić przewidywalne zużycie zasobów i prawidłowe działanie w klastrze. 2. Zawierać livenessProbe i readinessProbe, umożliwiające monitorowanie stanu aplikacji oraz kontrolę nad przekierowywaniem ruchu. 3. Być zbudowane jako bezpieczne obrazy kontenerowe, co oznacza uruchamianie kontenerów z użytkownikiem nieuprzywilejowanym (non-root), użycie minimalnego obrazu bazowego oraz brak zbędnych narzędzi w obrazie.

DEV-NOFUNC-00030	<p>Dostawca jest zobowiązany do realizacji pełnego cyklu rozwoju, wdrażania i zarządzania rozwiązaniem w modelu GitOps, z wykorzystaniem środowiska GitLab oraz narzędzi ArgoCD udostępnionych przez Zamawiającego.</p> <p>W szczególności:</p> <ol style="list-style-type: none"> 1. Cały kod źródłowy, manifesty Kubernetes oraz pliki konfiguracyjne muszą być przechowywane i wersjonowane w wyznaczonych repozytoriach GitLab. 2. Do generowania manifestów należy stosować preferowane technologie Zamawiającego: Tanka z Jsonnet, Helm lub ich kombinację. 3. Wszystkie zmiany konfiguracyjne i wdrożeniowe muszą być wprowadzane wyłącznie poprzez repozytoria Git, zgodnie z modelem GitOps. 4. Proces CI/CD musi być realizowany za pomocą pipeline'ów GitLab, zgodnie z wymaganiami Zamawiającego dotyczącymi jakości kodu, testów, bezpieczeństwa i zarządzania artefaktami. 5. Należy tworzyć przejrzyste, krótkie i opisowe komunikaty commitów. 6. Wymagana jest aktywna obserwacja statusu własnych pipeline'ów oraz szybkie reagowanie na błędy, w tym analiza i naprawa błędów wynikających ze zmian wprowadzonych przez Dostawcę. 7. Należy spełniać wymagania dotyczące jakości kodu, zdefiniowane przez Zamawiającego jako tzw. bramki jakości (quality gates), obejmujące m.in. linting, testy jednostkowe, testy integracyjne oraz skanowanie bezpieczeństwa.
DEV-NOFUNC-00040	<p>Dostawca jest zobowiązany do testowania i wdrażania rozwiązania zgodnie z wymaganiami Zamawiającego.</p> <p>W szczególności:</p> <ol style="list-style-type: none"> 1. Wszystkie zmiany muszą być testowane na środowiskach deweloperskich oraz stagingowych przed przekazaniem ich do wdrożenia na środowisku produkcyjnym. 2. Należy wykorzystywać pipeline'y CI/CD do automatycznego wdrażania aplikacji oraz uruchamiania testów jednostkowych na środowiskach testowych. 3. Wszelkie istotne zmiany dotyczące strategii wdrożeniowej, alokacji zasobów oraz architektury muszą być uzgadniane z zespołem DevOps Zamawiającego. 4. Po każdym wdrożeniu należy przeprowadzić weryfikację stanu operacyjnego i funkcjonalnego rozwiązania, obejmującą health check, smoke test oraz walidację funkcjonalną. 5. Każda metoda wdrożenia musi mieć przygotowany plan wycofania zmian (rollback plan) na wypadek błędów występujących po stronie produkcyjnej.
DEV-NOFUNC-00050	<p>Dostawca jest zobowiązany do zapewnienia pełnej zgodności rozwiązania z wymaganiami obserwowalności Zamawiającego.</p> <p>W szczególności:</p> <ol style="list-style-type: none"> 1. Aplikacja musi eksponować metryki w formacie zgodnym z Prometheus. 2. Wszystkie logi aplikacyjne powinny być kierowane na standardowe wyjścia stdout/stderr w celu dalszej agregacji. 3. Rozwiązanie musi generować logi operacyjne, audytowe oraz błędy w sposób umożliwiający ich przekazanie do systemów OpenSearch/VictoriaLogs oraz platforma SIEM (np. dopuszczalne jest użycie agenta Wazuh). 4. Należy zapewnić pełną zgodność z komponentami obserwowalności wdrożonymi w klastrze Kubernetes Zamawiającego (np. Prometheus Operator, Logging Operator).
DEV-NOFUNC-00060	Dostarczony komponent musi generować logi umożliwiające:

	<p>1. Zarządzanie i utrzymanie: identyfikację błędów aplikacyjnych, diagnostykę problemów technicznych, analizę incydentów.</p> <p>2. Bezpieczeństwo: śledzenie prób dostępu do zasobów, wykrywanie nieautoryzowanych działań, identyfikację anomalii i wspieranie analiz incydentów bezpieczeństwa.</p> <p>3. Monitorowanie użycia i wydajności: ocenę obciążenia komponentu, identyfikację wąskich gardeł, analizę zachowań użytkowników oraz optymalizację zasobów systemowych.</p>
DEV-NOFUNC-00070	<p>Wszystkie logi powinny być generowane i przesyłane w postaci ustrukturyzowanej (preferowany format JSON) z wykorzystaniem agentów logujących (np. Fluentd / Logging Operator) oraz udostępnienie do centralnego systemu zarządzania logami (np. OpenSearch, VictoriaLogs) za pomocą protokołów sieciowych zapewniających niezawodność i bezpieczeństwo transmisji.</p> <p>Struktura logów musi umożliwić łatwą integrację z systemami klasy SIEM oraz innymi narzędziami do centralnego monitorowania i analizy zdarzeń, w tym mapowanie pól czasu, poziomu istotności, źródła zdarzenia oraz identyfikatora usługi.</p>
DEV-NOFUNC-00080	<p>Logi generowane przez komponent muszą zawierać co najmniej następujące informacje:</p> <ol style="list-style-type: none"> 1. Znacznik czasu zdarzenia w formacie zgodnym z ISO 8601 (z precyzją do milisekund i uwzględnieniem strefy czasowej), 2. Poziom logowania (np. DEBUG, INFO, WARNING, ERROR), który powinien być możliwy do regulacji, na przykład za pomocą zmiennej systemowej, 3. Identyfikator instancji komponentu (jeśli dotyczy, np. w środowisku skalowalnym lub kontenerowym), 4. Identyfikator użytkownika lub systemu inicjującego zdarzenie (jeśli dotyczy), identyfikator sesji lub żądania (w celu korelacji zdarzeń w ramach jednej operacji), 5. Opis zdarzenia w postaci czytelnej dla człowieka oraz - jeśli to możliwe - również w ustrukturyzowanej postaci (np. JSON), 6. Identyfikator funkcji, systemu lub modułu, w którym zdarzenie zostało wygenerowane, 7. Adres IP źródła żądania (jeśli dotyczy) oraz kod błędu lub status operacji (jeśli dotyczy).
DEV-NOFUNC-00090	<p>Logi powinny być rejestrowane w sposób umożliwiający ich trwałe przechowywanie, centralizację oraz analizę przez systemy monitorujące i narzędzia audytowe. Niedopuszczalne jest zapisywanie danych uwierzytelniających (takich jak hasła, tokeny dostępu itp.) w logach w postaci jawnej.</p>
DEV-NOFUNC-00100	<p>Rozwiązanie musi być zaprojektowane i wdrożone w taki sposób, aby umożliwić przeprowadzanie wdrożeń (deploymentów) bez przerywania dostępności usługi dla użytkownika końcowego (zero downtime deployment).</p>
DEV-NOFUNC-00110	<p>Dostawca jest zobowiązany do zaprojektowania architektury rozwiązania w sposób umożliwiający jego horyzontalne skalowanie.</p>

9. Wymagania obszaru: Testing & Documentation

ID Wymagania	Opis Wymagania
--------------	----------------

DEV-NOFUNC-00120	<p>Dostawca jest zobowiązany do kompleksowego przygotowania testów i ich dokumentacji przed rozpoczęciem fazy UAT.</p> <p>W szczególności:</p> <ol style="list-style-type: none"> 1) przygotowania Strategii testów, 2) dostarczenia matrycy pokrycia wymagań (traceability matrix) - zarówno w zakresie testów manualnych jak i automatycznych, 3) opracowania scenariuszy testowych w systemie Jira udostępnionym przez Zamawiającego, 4) dokumentowania przebiegu testów manualnych i automatycznych w sposób umożliwiający ich weryfikację, 5) testy automatyczne muszą być uruchamiane w pipeline'ach CI/CD zdefiniowanych w GitLabie Zamawiającego.
DEV-NOFUNC-00130	Dostawca jest zobowiązany do przeprowadzenia testów wydajnościowych i testów obciążeniowych (stress testy - określenie limitów) oraz udokumentowania ich przebiegu i wyników.
DEV-NOFUNC-00140	Dostawca jest zobowiązany do przygotowania i przekazania Zamawiającemu raportu gotowości przed rozpoczęciem testów UAT.
DEV-NOFUNC-00150	<p>Dostawca zgłasza gotowość do odbioru w momencie spełnienia wszystkich warunków oraz przekazania Zamawiającemu kompletnego zestawu materiałów obejmującego:</p> <ol style="list-style-type: none"> 1) finalną wersję wytworzonego oprogramowania, 2) repozytorium z kodem źródłowym, 3) kompletną dokumentację użytkową, techniczną oraz utrzymaniową, 4) zestaw testów i scenariuszy, 5) raporty z testów, 6) instrukcje dla środowisk testowych i uruchomieniowych, 7) dokumentacja powinna być dostarcza w formie elektronicznej wraz z możliwością dalszej modyfikacji i wersjonowania.
DEV-NOFUNC-00160	Dostawca jest zobowiązany do przeprowadzenia szkoleń produktowych i operacyjnych dla zespołów Zamawiającego, w zakresie obsługi dostarczonego rozwiązania. Szczegółowy opis wymagań szkoleniowych został opisany w Załączniku 1e.
DEV-NOFUNC-00170	Dostawca jest zobowiązany do prowadzenia i bieżącego utrzymywania pełnej dokumentacji technicznej wszystkich opracowywanych lub zarządzanych komponentów. Szczegóły dotyczące standardu dokumentacji opisane są w Załączniku 1c.

10. Wymagania obszaru: Interoperability

ID Wymagania	Opis Wymagania
INT-NOFUNC-00010	<p>Dostawca jest zobowiązany do stosowania Design Systemu Zamawiającego w całym procesie projektowania i implementacji interfejsów użytkownika.</p> <p>W szczególności:</p> <ol style="list-style-type: none"> 1) Wszystkie komponenty UI muszą być zgodne ze stylistyką, hierarchią wizualną i interakcyjną zdefiniowaną w Design Systemie, 2) W przypadku braku danego komponentu w Design Systemie, Dostawca powinien dążyć do wykorzystania istniejących elementów w sposób maksymalnie zbliżony do

	wymaganej funkcjonalności i estetyki, z zachowaniem podstawowych założeń stylistycznych i brandingowych Design Systemu, 3) Rozwiązanie musi wspierać responsywność i dostępność (WCAG 2.1 na poziomie AA) zgodnie z wytycznymi Design Systemu. Na życzenie (po podpisaniu NDA) Zamawiający udzieli dostępu do Figmy.
INT-NOFUNC-00020	Rozwiązanie musi umożliwiać świadczenie usług w ramach różnych marek (brandów). Każdy brand może posiadać: 1) odrębną identyfikację wizualną, 2) odrębną domenę lub subdomenę, 3) odrębne ustawienia konfiguracyjne.
INT-NOFUNC-00030	Dostawca jest zobowiązany do pełnej integracji dostarczanego rozwiązania z istniejącym środowiskiem Zamawiającego, w sposób zapewniający jego spójność funkcjonalną, operacyjną i użytkową z aktualnym ekosystemem produktowym Zamawiającego. W szczególności, nowy produkt musi zostać zintegrowany z Common Core Service, centralnym komponentem wspierającym zarządzanie usługami i rolami, tożsamością użytkowników (IAM), billingiem oraz realizacją zamówień (order fulfilment). Szczegółowy opis Common Core Service oraz wymaganych integracji został zamieszczony w Załączniku 1b.

11. Wymagania obszaru: Security

ID Wymagania	Opis Wymagania
SEC-NOFUNC-00010	Rozwiązanie musi pozwalać na zabezpieczenie wszystkich interfejsów użytkownika (UI) i interfejsów programistycznych (API) przy użyciu protokołu HTTPS.
SEC-NOFUNC-00020	Dostawca jest zobowiązany do zapewnienia, że dostarczone rozwiązanie będzie integrować się z mechanizmami uwierzytelniania i zarządzania tożsamością Zamawiającego z wykorzystaniem standardów OIDC i SAML, z obsługą jednokrotnego logowania (SSO) oraz uwierzytelniania wieloskładnikowego (MFA): <ul style="list-style-type: none"> dla użytkowników zewnętrznych – poprzez federację (OIDC/SAML) lub dedykowany realm w centralnym systemie IAM Zamawiającego, dla użytkowników wewnętrznych – poprzez centralny katalog Zamawiającego, dostępny z użyciem OIDC lub SAML.
SEC-NOFUNC-00030	Dostawca jest zobowiązany do wdrożenia mechanizmu kontroli dostępu opartego na rolach (RBAC), z możliwością ograniczania dostępu na podstawie adresów IP.
SEC-NOFUNC-00040	Dostawca jest zobowiązany do zarządzania danymi wrażliwymi (np. hasła, tokeny, klucze API) wyłącznie z użyciem udostępnionego przez Zamawiającego systemu HashiCorp Vault lub OpenBao. Rozwiązanie musi być zintegrowane z Vault/OpenBao w sposób umożliwiający dynamiczne pobieranie sekretów.
SEC-NOFUNC-00050	Dostawca jest zobowiązany do realizacji zarządzania kluczami szyfrowania wyłącznie przy użyciu systemów KMS (Key Management Systems) stosowanych przez Zamawiającego, tj. HashiCorp Vault / OpenBao.
SEC-NOFUNC-00060	Dostawca jest zobowiązany do zapewnienia, że dane tajne nie są zapisywane w kodzie źródłowym ani plikach konfiguracyjnych w repozytorium oraz że nie

	pojawiają się w logach (stdout, stderr), ani w komunikatach błędów widocznych dla użytkownika lub operatora.
SEC-NOFUNC-00070	<p>Dostawca jest zobowiązany do zapewnienia, że wszystkie dane przetwarzane przez rozwiązanie będą chronione poprzez mechanizmy szyfrowania:</p> <ul style="list-style-type: none"> Dane w spoczynku (at rest): muszą być szyfrowane przy użyciu silnych algorytmów zgodnych z aktualnymi rekomendacjami NIST, ENISA oraz wytycznymi OWASP Cryptographic Storage – w szczególności AES-256 w trybie GCM lub XTS, albo równoważnych mechanizmów akceptowanych w politykach bezpieczeństwa Zamawiającego. Dane w transzycie (in transit): cała komunikacja musi być chroniona przy użyciu protokołów zapewniających poufność i integralność, w szczególności TLS 1.3 lub TLS 1.2 z silnymi zestawami szyfrów (np. ECDHE_RSA lub ECDHE_ECDSA z AES-256-GCM/SHA-384), zgodnie z wytycznymi OWASP Transport Layer Protection. Rozwiązanie musi umożliwiać integrację z systemami zarządzania kluczami (KMS, HSM, Vault/OpenBao lub równoważne)
SEC-NOFUNC-00080	Dostawca musi posiadać wdrożony System Zarządzania Jakością (np. zgodny z ISO 9001) oraz System Zarządzania Bezpieczeństwem Informacji (np. zgodny z ISO 27001); Zamawiający może zażądać potwierdzenia w formie certyfikatów lub dokumentacji.
SEC-NOFUNC-00090	Każdy komponent rozwiązania musi implementować mechanizmy QoS, throttlingu oraz limitowania liczby zapytań, w celu ograniczenia ryzyka przeciążenia systemu niezależnie od tego, czy użytkownikiem końcowym jest osoba zewnętrzna czy pracownik Zamawiającego.
SEC-NOFUNC-00100	Rozwiązanie musi umożliwiać Zamawiającemu realizację regularnych kopii zapasowych danych, w tym bazy danych, zgodnie ze standardami Zamawiającego.
SEC-NOFUNC-00110	Rozwiązanie musi działać poprawnie w środowisku wysokiej dostępności i być odporne na awarię pojedynczego komponentu.
SEC-NOFUNC-00120	Dostawca, realizując prace w ramach przedmiotu zamówienia, może uzyskać dostęp do środowiska Zamawiającego, w tym do sieci wewnętrznej oraz zasobów systemowych, wyłącznie za pośrednictwem dedykowanego połączenia VPN zarządzanego przez Zamawiającego. Warunkiem uzyskania dostępu jest każdorazowe uwierzytelnienie z wykorzystaniem mechanizmu uwierzytelniania dwuskładnikowego (2FA), zgodnego z polityką bezpieczeństwa Zamawiającego. Wszelki dostęp odbywa się z uprzednio zarejestrowanych i zatwierdzonych urządzeń końcowych oraz jest monitorowany i rejestrowany w sposób umożliwiający audyt zgodności z wymaganiami bezpieczeństwa.
SEC-NOFUNC-00130	Dostawca jest zobowiązany do projektowania i implementacji wszystkich interfejsów wystawianych na zewnątrz w sposób zgodny z wymaganiami integracji z zaporą aplikacyjną Web Application Firewall (WAF) Zamawiającego.
SEC-NOFUNC-00140	<p>Dostawca musi zapewnić, że wytworzone lub dostarczone rozwiązanie, w tym wszystkie jego komponenty (np. obrazy kontenerowe, moduły oprogramowania, integracje i rozszerzenia), będzie posiadało stale aktualny SBOM (Software Bill of Materials).</p> <p>SBOM powinien spełniać następujące kryteria:</p>

	<ul style="list-style-type: none"> • Być generowany i utrzymywany dla każdego obrazu kontenerowego oraz wszystkich elementów rozwiązania. • Zawierać kompletną listę zależności (pakiety, biblioteki, moduły), wraz z wersjami i źródłami pochodzenia. • Być dostępny w uznanym standardzie (np. SPDX lub CycloneDX). • Być automatycznie aktualizowany przy każdej zmianie komponentów lub wersji oprogramowania, tak aby pozostawał stale aktualny. • Umożliwiać integrację z procesami zarządzania podatnościami (np. skanowanie CVE, analiza ryzyka).
--	--

12. Wymagania obszaru: Maintainability & Operation

ID Wymagania	Opis Wymagania
MNT-NOFUNC-00005	Rozwiązanie musi wspierać mechanizmy aktualizacji z minimalnym wpływem na dostępność usługi (rolling/zero-downtime upgrades), realizowane z wykorzystaniem architektury typu master-replica (primary-secondary). Aktualizacje i zmiany konfiguracyjne powinny być wykonywane najpierw na replikach podrzędnych, a następnie na instancji głównej, z zastosowaniem kontrolowanego switchover pomiędzy repliką a instancją główną.
MNT-NOFUNC-00010	Dostawca jest zobowiązany do przestrzegania tabeli zawierającej czasy reakcji i czasy rozwiązania dla poszczególnych typów zgłoszeń incydentów - Załącznik 1a.
MNT-NOFUNC-00020	Dostawca jest zobowiązany do zapewnienia wsparcia technicznego w następujących godzinach: a) dla zgłoszeń typu „request” - od 8:00 do 20:00, w dniach roboczych w Polsce, b) dla zgłoszeń typu „incydent” - całodobowo, przez 365 dni w roku.
MNT-NOFUNC-00030	Dostawca jest zobowiązany do uzyskania uprzedniej akceptacji Zamawiającego dla wszystkich planowanych prac serwisowych realizowanych po stronie Dostawcy.
MNT-NOFUNC-00040	Wszelkie procesy operacyjne, serwisowe, dokumentacyjne oraz działania wykonawcy (w tym w zakresie zarządzania usługami, zmianami i operacjami serwisowymi) muszą być zgodne z ITIL v4 oraz procedurami i zasadami obowiązującymi w CloudFerro,

UWAGA: wymagania obszaru Maintainability obowiązują Dostawcę w okresie utrzymywania rozwiązania przez Dostawcę

13. Integralną częścią Załącznika 4.1 są poniższe dokumenty:

- 13.1. Załącznik 4.1a - Tabela z oczekiwanymi czasami reakcji
- 13.2. Załącznik 4.1b - Opis Common Core Service
- 13.3. Załącznik 4.1c - Standard dokumentacji technicznej
- 13.4. Załącznik 4.1d - Lista zatwierdzonych technologii
- 13.5. Załącznik 4.1e – Wymagania szkoleniowe

Załącznik 4.1a – Tabela z oczekiwanymi czasami reakcji

1. Klasyfikacja incydentów według priorytetów oraz oczekiwany poziom Service Level

Opis	P1 – Critical	P2 – High	P3 – Moderate	P4 – Low
Definicja	Incydenty mające poważny wpływ na działalność klientów, takie jak: utrata dostępności usług dla większości użytkowników, poważne naruszenia bezpieczeństwa, awarie infrastruktury krytyczne dla działania usług. Wymaga natychmiastowej reakcji i mobilizacji zespołów.	Incydenty mające istotny wpływ na działalność klientów, takie jak: utrata dostępności usług dla części użytkowników, ograniczone naruszenia bezpieczeństwa, problemy z wydajnością. Wymaga szybkiej reakcji i zaangażowania odpowiednich zespołów.	Incydenty mające umiarkowany wpływ na działalność klientów, takie jak: drobne błędy oprogramowania, problemy z konfiguracją, opóźnienia w dostarczaniu danych. Wymaga reakcji, ale niekoniecznie natychmiastowej.	Incydenty mające niewielki wpływ na działalność klientów, takie jak: błędy nie wpływające na funkcjonalność usług, drobne kwestie techniczne, żądania informacyjne. Reakcja na te incydenty może być odłożona na później i nie wymaga natychmiastowej mobilizacji zespołów.
Czas reakcji	20 min	1h	1 dzień roboczy	1 dzień roboczy
Czas przywrócenia	4h	2 dni	10 dni	30 dni
RCA – Root Cause Analysis	obowiązkowe	obowiązkowe	opcjonalne	opcjonalne
Czas dostarczenia RCA	7 dni	7 dni	-	-

Załącznik 4.1b - Opis Common Core Service

1. Definicje

Service - Produkt cyfrowy CloudFerro lub partnera zewnętrznego umożliwiający organizacjom klientów wykonywanie zadań online przy użyciu infrastruktury cyfrowej i aplikacji.

Common Core Service - Aplikacja biznesowa CloudFerro wspierająca zarządzanie usługami i rolami, IAM, Billing, realizację zamówień.

Tenant Manager - Centralny system zarządzania organizacjami i użytkownikami obsługujący provisioning, billing i kontrolę dostępu dla wielotenantowej platformy chmurowej CloudFerro.

Registration flow - Proces onboarding użytkowników przez Tenant Manager z integracją Keycloak do tworzenia kont i konfiguracji organizacji.

IAM user and organization management - Identity and Access Management przez integrację Tenant Manager z Keycloak do uwierzytelniania i autoryzacji.

Active billed services - Monitorowanie usług w czasie rzeczywistym i śledzenie rozliczeń przez integrację Service Manager i Tenant Manager.

Usage reports - Kompleksowy system raportowania do konsumpcji zasobów, podsumowań rozliczeń i analityki finansowej.

Product Catalogue - Scentralizowany system zarządzania produktami i cenami z konfiguracjami specyficznymi dla marek i regułami rozliczeniowymi.

Virtual Wallets management - System prepaid billing zarządzający saldami kredytowymi klientów przez modele PPU i PAYG.

Email and InApp Notifications - Wielokanałowy system powiadomień dla alertów rozliczeniowych, aktualizacji usług i komunikacji z użytkownikami.

Tickets processing - System obsługi klienta zarządzający żądaniami serwisowymi przez integrację JIRA.

My Requests processing - Zarządzanie żądaniami serwisowymi inicjowanymi przez użytkowników z workflow zatwierdzania i integracją provisioningu.

Billing processing - Ekosystem rozliczeniowy składający się z usług Mediation, Accounting, Charging, Billing Collector, Billing Processor i Contracting.

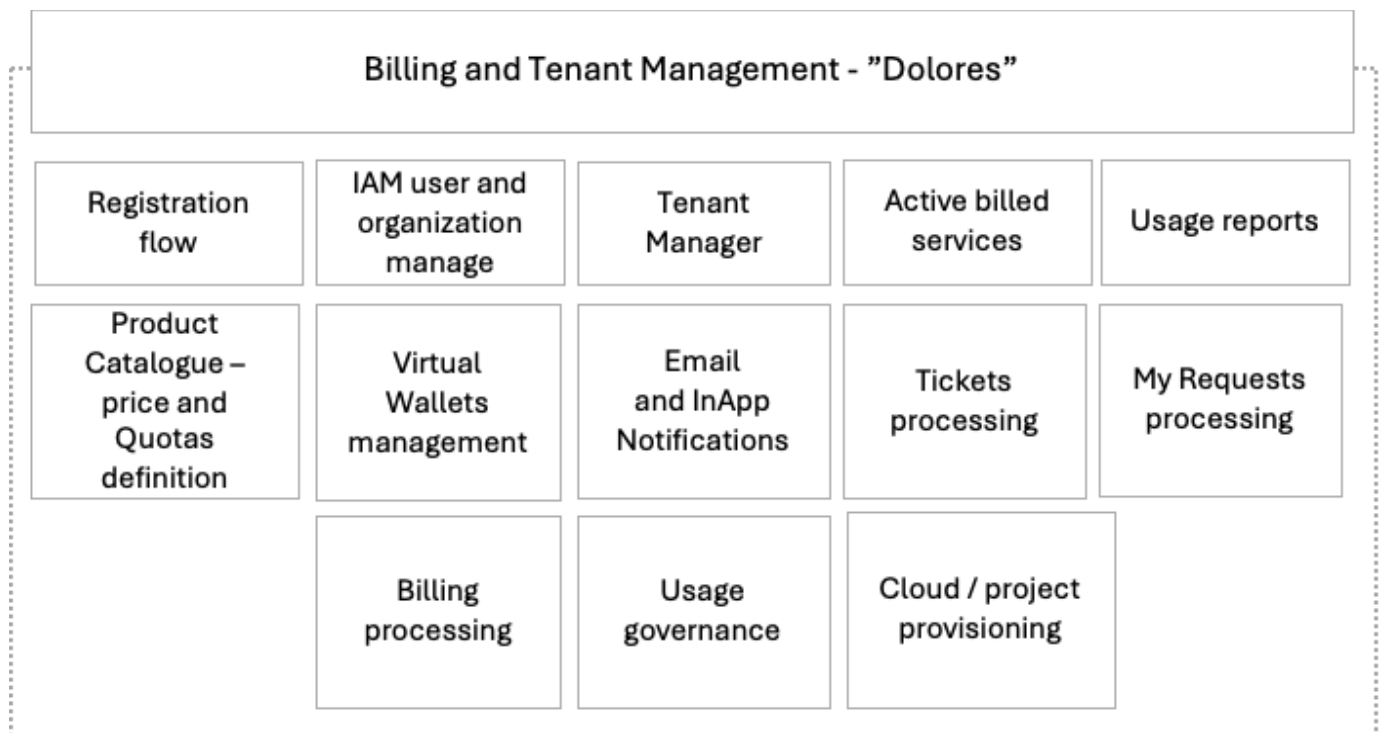
Usage governance - Kontrola alokacji zasobów przez Decision Service egzekwujący quotas i reguły biznesowe.

Cloud / project provisioning - Automatyczny provisioning infrastruktury przez workflow oparty na saga w Tenant Manager i usługa Provisioner.

2. Opis wysokopoziomowy aplikacji zarządzania usługami

CloudFerro zapewni dedykowany portal zarządzania tenantami i administracji biznesowej "Dolores" - wewnętrznie opracowaną aplikację skoncentrowaną na użytkowniku, kluczową dla spójnego doświadczenia użytkownika, zgodną z istniejącą ofertą chmury publicznej CloudFerro. Ta sama usługa jest dziś używana przez kilka marek, w szczególności Creodias i CloudFerro.

Ogólny model funkcjonalny Dolores



System Dolores rozszerza funkcje chmury OpenStack o możliwości rozliczania, raportowania, governance usług i provisioning zasobów w zarejestrowanej organizacji.

2.1 Komponenty funkcjonalne platformy Dolores

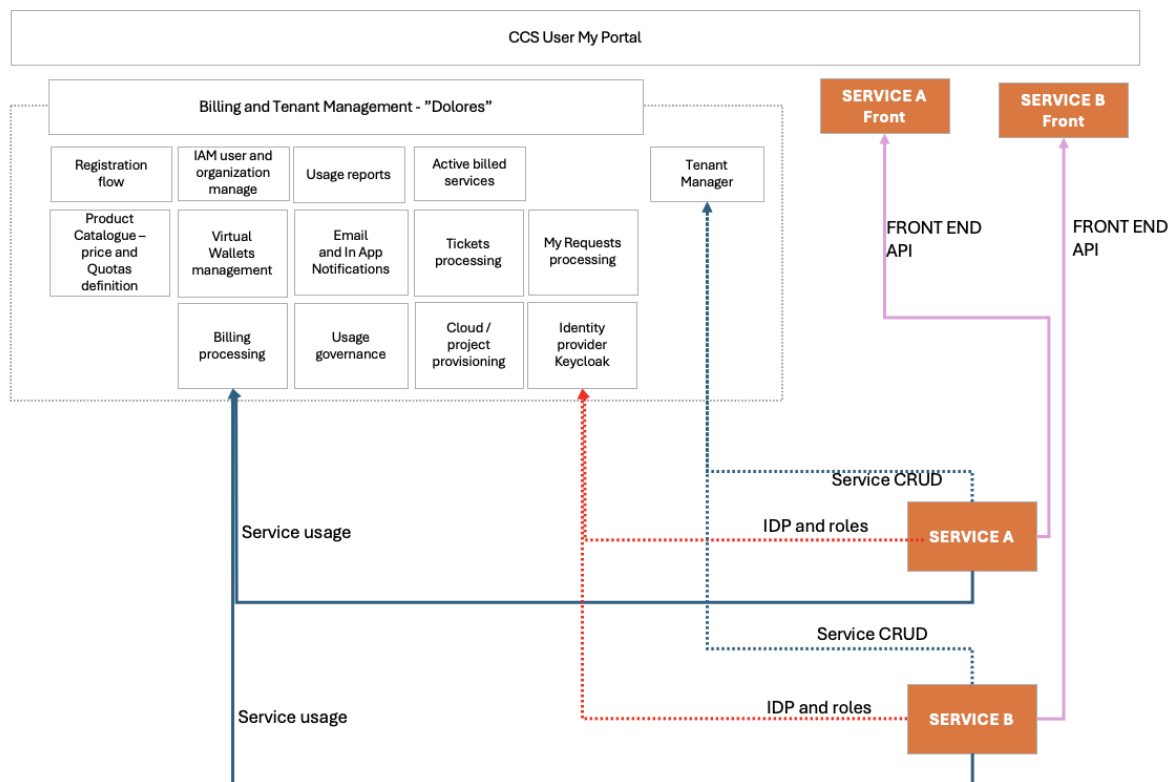
Platforma Dolores składa się z połączonych komponentów, które współpracują, aby zapewnić kompleksowe możliwości zarządzania usługami chmurowymi i rozliczeniami:

- **Registration flow** - Zaimplementowany przez system zarządzania użytkownikami Tenant Manager z integracją Keycloak, zarządzający procesem onboardingu nowych użytkowników, w tym tworzeniem kont użytkowników w Keycloak, weryfikacją tożsamości, konfiguracją organizacji i początkowym provisioningiem dostępu. Zapewnia domyślne przypisanie ról (admin, member, member+) i umożliwia użytkownikom rejestrację organizacji lub dołączanie do istniejących za pomocą kodów zaproszenia z odpowiednią kontrolą dostępu opartą na grupach.
- **IAM user and organization management** - Zaimplementowany przez kompleksowy system identity and access management Tenant Manager integrujący się z Keycloak do uwierzytelniania użytkowników, autoryzacji opartej na rolach (admin, member, member+), zarządzania hierarchią organizacji, zarządzania grupami, Single Sign-On we wszystkich usługach CloudFerro i obsługi wielu realmów dla separacji marek.
- **Tenant Manager** - Centralny system zarządzania organizacjami i użytkownikami zapewniający kompleksowe zarządzanie cyklem życia tenantów, w tym rejestrację klientów, przypisanie użytkowników, operacje rozliczeniowe, provisioning zasobów chmurowych przez workflow oparte na saga, zarządzanie portfelami (PPU/PAYG), raportowanie finansowe i integrację z wieloma usługami zewnętrznymi (Service Manager, Product Catalogue, Provisioner, Keycloak) dla kompletnej administracji tenantów.
- **Active billed services** - Monitorowanie usług w czasie rzeczywistym i śledzenie rozliczeń zaimplementowane przez system zarządzania kontraktami Service Manager i monitor rozliczeń Tenant Manager. Śledzi aktywowane usługi chmurowe, ich status rozliczeniowy, monitorowanie konsumpcji zasobów, zgodność kontraktów, zarządzanie cyklem życia usług i integrację z systemami rozliczeniowymi dla kompleksowego nadzoru rozliczeniowego i alokacji zasobów opartej na kontraktach.
- **Usage reports** - Kompleksowy silnik raportowania zaimplementowany przez usługi raportowania Tenant Manager zapewniający szczegółową analitykę konsumpcji zasobów, podsumowania rozliczeń, zestawienia kosztów, trendy użycia i raporty finansowe z obsługą wielu formatów wyjściowych (JSON, CSV) i szczegółowym podziałem według projektów, kontraktów, użycia zasobów i agregacji logów opłat.
- **Product Catalogue** - Scentralizowany system zarządzania produktami i cenami obsługujący definicje usług chmurowych, złożone struktury cenowe, konfiguracje specyficzne dla marek, zarządzanie pakietami, konfiguracje workflow, zarządzanie usługami rozliczeniowymi, zarządzanie zasobami multi-cloud i umożliwiający wdrożenia white-label z dostosowanymi

katalogami produktów i zaawansowanymi regułami rozliczeniowymi przez konfiguracje specyficzne dla marek.

- Virtual Wallets management - System prepaid billing zaimplementowany przez Tenant Manager zarządzający saldami kredytowymi klientów przez modele PPU (Pay-Per-Use) i PAYG (Pay-As-You-Go), przetwarzający transakcje płatnicze, śledzący logi opłat, zarządzanie kontraktami, automatyczne operacje rozliczeniowe i integrację generowania faktur z usługami zewnętrznymi jak InvoiceOcean.
- Email and InApp Notifications - Wielokanałowy system powiadomień zaimplementowany przez Tenant Manager dostarczający alerty rozliczeniowe, aktualizacje usług, powiadomienia systemowe i komunikację z użytkownikami przez usługi email SMTP i wiadomości w aplikacji z odpowiednim zarządzaniem szablonami, śledzeniem dostarczania i integracją z procesami worker dla niezawodnego dostarczania.
- Tickets processing - System obsługi klienta zaimplementowany przez Tenant Manager zarządzający zadaniami serwisowymi, problemami technicznymi, zapytaniami rozliczeniowymi przez integrację JIRA, automatyczne kierowanie zgłoszeń, śledzenie statusu, kompleksowe zarządzanie workflow wsparcia z odpowiednimi procedurami eskalacji i automatyzacją konfiguracji użytkowników.
- My Requests processing - Zarządzanie zadaniami serwisowymi inicjowanymi przez użytkowników zaimplementowane przez Tenant Manager obsługujące aktywacje usług, modyfikacje zasobów, ządania wsparcia, workflow zatwierdzania z odpowiednim śledzeniem statusu i integracją z systemami provisioningu dla automatycznej realizacji ządań przez orkiestrację opartą na saga.
- Billing processing - Zaawansowany ekosystem rozliczeniowy składający się z wielu wyspecjalizowanych usług: Mediation Service do transformacji zdarzeń OpenStack, Accounting Service do agregacji użycia w sesje rozliczeniowe, Charging Service do kalkulacji cen i logiki rozliczeniowej, Billing Collector do ingestion danych użycia zewnętrznych, Billing Processor do walidacji rozliczeń zagregowanych i Contracting Service do alokacji pojemności opartej na bucketach, obsługujący wiele modeli rozliczeniowych, w tym rozliczenia oparte na kontraktach i użyciu.
- Usage governance - System kontroli alokacji zasobów zaimplementowany przez Decision Service, który egzekwuje quotas przez walidację ządań zasobów względem quotaas kont pobieranych z Tenant Manager, stosowanie reguł biznesowych z Product Catalogue, zapewnienie zgodności z ograniczeniami organizacyjnymi i warunkami kontraktów z Service Manager przed umożliwieniem provisioningu zasobów w OpenStack.
- Cloud / project provisioning - Automatyczny system provisioningu infrastruktury zaimplementowany przez workflow oparte na saga w Tenant Manager, które koordynują z usługą Provisioner tworzenie projektów OpenStack, alokację zasobów, konfigurację sieci, zarządzanie cyklem życia zasobów chmurowych z odpowiednim uwierzytelnianiem, egzekwowaniem quotas i integracją z Dynamic Vendor Endpoint do dystrybucji konfiguracji VM.

3. Ogólne wymagania integracyjne



3.1 Rejestracja użytkowników

Rejestracja użytkowników powinna następować zgodnie ze wspólnym procesem rejestracji usług podstawowych CloudFerro, zapewniając użytkownikowi wspólne środki tożsamości do współdzielenia między wszystkimi usługami CloudFerro.

Rejestracja i zarządzanie użytkownikami

System CloudFerro obsługuje przepływ rejestracji użytkowników zapewniający zbieranie wymaganych zgód (obowiązkowych i opcjonalnych) i danych. Proces zapewnia rejestrację konta użytkownika w IDP (Keycloak), aktywację konta, a także konfigurację uwierzytelniania dwuskładnikowego. Proces rejestracji z opcjami definiowania wymaganych, opcjonalnych atrybutów (pola niestandardowe organizacji i użytkowników), a także zgód.

CloudFerro będzie działać jako dostawca IDP

Proces umożliwi użytkownikowi rozpoczęcie rejestracji z dostępem do Tenant Managera zapewniając domyślny zestaw ról.

Możliwe domyślne zestawy ról:

- Dostęp do aktywacji usług
- Dostęp do usług bezpłatnych
- Dostęp do określonych funkcji tenant managera: moja organizacja, mój profil, zgłoszenia serwisowe, powiadomienia

Nowy użytkownik rejestruje się rolami ADMIN.

Faktycznie jako następny użytkownik powinien zarejestrować organizację i zostać jej administratorem lub dołączyć do organizacji, jeśli użytkownicy mają kod zaproszenia dostarczony przez jej administratora.

ADMIN będzie mógł przypisywać role użytkownikom oraz usuwać użytkownika z organizacji.

3.2 Funkcje zarządzania usługami – CRUD

Każda nowa usługa integrowana z zarządzaniem usługami CloudFerro powinna zapewnić operację wsparcia:

1. Service Create – Endpoint dostarczony przez CloudFerro,
2. Suspend soft,
3. Suspend hard,
4. Update,
5. Case,
6. Active services – pokaż aktywne zasoby.

3.2.1 Tworzenie usług

Endpoint dostarczony przez CloudFerro do realizacji kroku aktywacji usługi, a także aktywacji elementu usługi.

Service item - zasób lub element generujący dane użycia.

Proces biznesowy umożliwiający użytkownikowi aktywację usługi.

Powinny być obsługiwane następujące możliwości:

1. Self service – proces umożliwiający dostęp do usług automatycznie,
2. Usługa nie powinna zezwalać na użycie do momentu otrzymania informacji przez endpoint.

Domyślne warunki biznesowe dla aktywacji usług:

- Użytkownik jest zarejestrowany,
- Konto użytkownika zostało aktywowane przez użytkownika,
- Użytkownik zaakceptował warunki i zasady usługi – może być spełnione przez ogólne T&C podpisane podczas rejestracji lub dodatkowe zebrane w osobnej transakcji,
- Organizacja została utworzona – w przyszłości przepuścimy dane organizacji przez proces kontroli sankcji,
- Użytkownik ma aktywne środki płatnicze:
 - Aktywny kontrakt PAYG, lub:
 - Aktywny portfel PPU z wystarczającą ilością kredytów – minimum powinno być zdefiniowane przez definicję usługi lub proces, lub:
 - Aktywny kontrakt terminowy pokrywający żądany produkt,
- Minimalna kwota kredytów – usługa może zdefiniować minimalną wartość, którą organizacja powinna posiadać przed aktywacją usługi lub elementu usługi.

3.2.2 Zawieszenie usług

Billing jest odpowiedzialny za monitorowanie użycia usług i warunków biznesowych. W przypadku określonych warunków, Billing powinien dysponować środkami w celu wykonania sankcji.

Usługa powinna zapewnić Common Core Services akcję umożliwiającą proces biznesowy zawieszenie usługi w wyniku:

- Niewystarczających środków płatniczych,
- Przeterminowanych środków płatniczych,
- Decyzji biznesowej operatora opartej na T&C (naruszenie usługi).

Oczekiwane zachowanie: metoda wykonania jest funkcją specyficzną dla usługi, która powinna prowadzić do ograniczeń konsumpcji usługi.

Zastosowanie akcji powinno skutkować:

- zawieszeniami komponentów usługi,
- zawieszeniami praw dostępu,
- innymi metodami realizującymi cel akcji.

Oczekiwane flow:

- Billing będzie emitować zdarzenia domenowe, na które może reagować każda indywidualna usługa, np. zawieszać poszczególne zasoby. Usługa zewnętrzna jest odpowiedzialna za decydowanie kiedy i jak zawieszać zasoby,
- Zdarzenia rozliczeniowe będą wysyłane przez magistralę zdarzeń, zapewniając, że billing pozostaje oddzielony i nieświadomy lokalizacji usługi zewnętrznej,
- Usługa zrealizuje sankcję,
- Akcja będzie powtarzana przez billing tak długo, jak konsumpcja usługi jest rejestrowana przez przepływ przetwarzania rozliczeń.

Usługa powinna zapewnić 2 kategorie zawiesznień:

- Suspend soft – skutkujące:
 - Zawieszeniami komponentów usługi,
 - Odmową w nowych żądaniach elementów usługi.
- Suspend hard:
 - Suspend soft,
 - Cofnięcie praw do akcji *Write* – z wyjątkiem – tych, które prowadzą do ograniczeń konsumpcji usług,
 - Przełączenie usługi na okres karencji.

3.2.3. Okres karencji i zakończenie

Okres karencji

Faza włączona automatycznie przez proces cyfrowy lub przez operatora.

W fazie w zależności od możliwości usługi:

- Usługa powinna być technicznie zawieszona (np. VM odłączone od stanu obciążenia)
- Role dostępu ograniczone do odczytu, z wyjątkiem akcji prowadzących do ograniczeń konsumpcji
- Aktywacja zasobów powinna być wyłączona
- Dezaktywacja zasobów / projektów powinna być włączona
- Odczyt i transfer danych powinny być włączone, jeśli oferowane bezpłatnie zgodnie z cennikiem.
- Użytkownik powinien mieć dostęp do wsparcia cyfrowego.

Użytkownik może cofnąć proces spełniając określone wymagania biznesowe:

- należne płatności pokryte
- środki płatnicze są aktywne

Zakończenie

Proces prowadzący do:

- zwolnienia zasobów usługi – odpowiedzialność usługi,
- archiwizacji danych rozliczeniowych (konta, dane użycia). - odpowiedzialność CCS,

Użytkownik powinien mieć dostęp do danych użycia rozliczeniowego przez okres X miesięcy (zalecane 12 miesięcy).

Konta użytkowników nie powinny być usuwane, chyba że użytkownik żąda usunięcia danych osobowych zgodnie z polityką prywatności i/lub użytkownik wykonuje akcje usunięcia.

Usługa powinna zapewnić endpoint do wykonania zakończenia usługi i elementów usługi.

Rezultat:

- zasoby są zwolnione,
- konsumpcja usługi zatrzymana,
- nie generowane są dane użycia.

3.2.4. Aktywne usługi – Pokaż aktywne usługi

Usługa powinna zapewnić CloudFerro endpoint umożliwiający odczyt listy aktywnych usług / elementów usług.

Endpoint powinien zapewnić:

- nazwę zasobu,
- id zasobu,
- wartość - faktyczne użycie mierzone na usługę w momencie żądania,
- ilość – opcjonalne,
- id usługi.

3.3 Integracja z IDP i systemem zarządzania rolami

CloudFerro będzie działać jako dostawca IDP.

Usługa powinna integrować się z systemem identity and access management CloudFerro dla ujednoliconego uwierzytelniania użytkowników we wszystkich usługach CloudFerro.

Usługa powinna umożliwiać Single Sign-On (SSO) działający z IDP CloudFerro, pozwalając użytkownikom na dostęp do usługi przy użyciu istniejących danych uwierzytelniających CloudFerro bez dodatkowego logowania.

3.4. Integracja zarządzania quotas

Usługa powinna integrować się z CloudFerro Decision Service zapewniając governance alokacji zasobów, zapewniając zgodność żądań z przypisanymi quotas.

3.4.1. Integracja Decision Service

Obowiązkowy proces walidacji

Proces zapewniający walidację alokacji zasobów przez CloudFerro Decision Service przed provisioningiem lub modyfikacją zasobów.

Przed tworzeniem lub modyfikacją jakichkolwiek zasobów, usługa powinna walidować żądanie przez CloudFerro Decision Service:

- Wszystkie żądania alokacji zasobów powinny być przesłane do Decision Service przed przetwarzaniem,
- Przesłać wymagania dotyczące zasobów (vCPU, RAM, storage, itp.) wraz z kontekstem organizacji,
- Uszanować odpowiedź Decision Service - kontynuować tylko gdy otrzymana zostanie decyzja "allow",
- Przesłać kompletne wymagania dotyczące zasobów w jednym żądaniu dla dokładnej walidacji.

3.4.2. Egzekwowanie quota

Proces zapewniający zgodność alokacji zasobów opartą na odpowiedziach Decision Service.

Usługa powinna implementować ścisłe egzekwowanie quota oparte na odpowiedziach Decision Service:

- Gdy Decision Service odrzuca żądanie z powodu quota, odmówić alokacji zasobów i zapewnić jasny komunikat błędu użytkownikowi,
- Usługa nigdy nie powinna omijać walidacji Decision Service lub alokować zasobów gdy quotas są przekroczone,
- Zastosować walidację quota w momencie żądania zasobu, nie po alokacji.

3.5. Integracja rozliczeniowa

Usługa musi integrować się z CloudFerro Billing Collector Service dla automatycznego raportowania danych użycia i przetwarzania rozliczeń.

3.5.1. Integracja Billing Collector

Obowiązkowy proces przesyłania danych użycia

Proces zapewniający, że konsumpcja zasobów usługi zewnętrznej jest automatycznie przechwytywana i raportowana do CloudFerro Billing Collector Service dla dokładnego rozliczania klientów.

Usługa musi przysyłać dane użycia do Billing Collector Service:

- Pobierać konfigurację reportera z Product Catalogue przed przesłaniem danych rozliczeniowych,
- Przesyłać elementy rozliczeniowe z odpowiednimi danymi agregatora pasującymi do skonfigurowanego schematu,

- Dołączać kontekst organizacji i szczegóły konsumpcji zasobów dla dokładnego rozliczania,
- Uwierzytelniać się odpowiednio i obsługiwać błędy walidacji właściwie.

3.5.2. Wymagania dotyczące danych użycia

Standardowy format danych rozliczeniowych

Usługa musi przestrzegać standardów danych rozliczeniowych CloudFerro dla automatycznego przetwarzania:

- Zapewniać kompletne informacje rozliczeniowe z unikalnymi identyfikatorami zdarzeń i danymi konsumpcji zasobów
- Zapewniać spójność danych z konfiguracją reportera i walidacją organizacji
- Obsługiwać przetwarzanie wsadowe i kompleksową obsługę błędów dla niepowodzeń walidacji

3.6. Obsługa multi-brand i multi-tenant

Usługa musi obsługiwać architekturę CloudFerro skoncentrowaną na markach z kompletną izolacją tenantów i możliwościami white-label:

- Działać w kontekstach specyficznych dla marek, gdzie każda marka ma dedykowany realm uwierzytelniania, ustawienia konfiguracyjne i reguły biznesowe dla kompletnej izolacji operacyjnej
- Zapewniać kompletne rozdzielenie danych i zasobów między organizacjami klientów w ramach każdej marki przez dedykowane zarządzanie tożsamością i kontrolę dostępu

Obsługiwać dostosowanie specyficzne dla marki, w tym dedykowane domeny, lokalizację i niezależną logikę biznesową na współdzielonej infrastrukturze.

Załącznik 4.1c - Standard dokumentacji technicznej

1. Dokumentacja techniczna przeznaczona na potrzeby rozwoju i utrzymania po stronie CloudFerro musi zawierać:

1.1. Opis systemu i celów

- 1.1.1. Zakres i przeznaczenie rozwiązania
- 1.1.2. Diagram architektury – komponenty, zależności i integracje
 - 1.1.2.1. W modelu c4, kontekst, kontenery i komponenty
 - 1.1.2.2. Diagramy przepływów
 - 1.1.2.3. Wzorce architektoniczne
 - 1.1.2.4. Ograniczenia systemu
 - 1.1.2.4.1. Aspekt CAP (Consistency, Availability, Partition Tolerance)
 - 1.1.2.4.2. Kluczowe kompromisy projektowe (tradeoff) i decyzje architektoniczne wraz z opisem i wpływem na działanie aplikacji

1.2. Interfejsy i API – dotyczy wszystkich interfejsów z diagramu kontenerów

- 1.2.1. Dokumentacja interfejsów HTTP w formacie OpenApi wersja > 3 wraz z przykładami zapytań / odpowiedzi i opisem
- 1.2.2. Dokumentacja innych interfejsów za pomocą JSON schema, grpc/Protobuf, event streaming lub innych popularnych standardów
- 1.2.3. Autoryzacja, limity

1.3. Szczegóły implementacyjne

- 1.3.1. Może być dostarczona jako część repozytorium z kodem kontenera, kodem deploymentu lub kodem testów w formacie markdown
- 1.3.2. Opis kluczowych algorytmów, struktur danych, workflow i tego jak aplikacja działa
- 1.3.3. Dokumentacja struktury bazy danych (jeśli występuje) powinna być automatycznie generowana z aktualnego schematu oraz zawierać instrukcję umożliwiającą samodzielne jej odtworzenie
- 1.3.4. Opis użytych konwencji kodowania i standardów w repozytorium

1.4. Testy jednostkowe, integracyjne i end-to-end

- 1.4.1. Instrukcja jak uruchomić i jak rozwijać (szczególnie ważne w przypadku testów integracyjnych i end to end)
- 1.4.2. Testy traktujemy tak jak komponent i dokumentacja powinna zawierać wszystko z sekcji 1.3 Szczegóły implementacyjne
- 1.4.3. Dokumentacja opisująca metodologię i rozwiązania techniczne przyjęte w przypadku testów end-to-end i integracyjnych

1.5. Wdrażanie i budowanie

- 1.5.1. Opis struktury, modułów i użytych narzędzi
- 1.5.2. Instrukcja uruchomienia
- 1.5.3. Instrukcje restartu komponentów i reakcji na awarie lub nieudane wdrożenia - rollback, skalowanie, ponowne wdrożenia
- 1.5.4. Opis CI/CD - co zawiera, integracja z wdrożeniami

1.6. Utrzymanie

- 1.6.1. Jak wykrywać i rozwiązywać błędy i restartować komponenty. Wraz z opisem które działania mogą spowodować downtime aplikacji
- 1.6.2. Procedura aktualizacji i zmiany konfiguracji i ich wpływu na downtime aplikacji
- 1.6.3. Opis monitoringu, logów i alertów
- 1.6.4. Backup jak tworzyć i przywracać wraz z ograniczeniami

1.7. Zarządzanie zmianami i wersjonowanie

- 1.7.1. Sposób wersjonowania API i bazy danych
- 1.7.2. Historia zmian pomiędzy kolejnymi wersjami

1.8. Niefunkcjonalne

- 1.8.1. Wydajność: oczekiwana liczba użytkowników, liczba operacji na sekundę, transfer wraz z opisem uwzględniającym zmianę tych parametrów wraz ze skalowaniem
- 1.8.2. Skalowanie i wymagania sprzętowe (RAM, cpu, dyski)
- 1.8.3. Ograniczenia środowiskowe i systemowe

Załącznik 4.1d - Lista zatwierdzonych technologii

1. Poniższa lista zawiera zbiór zatwierdzonych technologii w CloudFerro:

1.1. Programming languages

Lp.	Name	Description
1.	Python	Default for all projects
2.	Go	When justified by high performance or parallelism requirements
3.	Java	When required for compatibility with an external system
4.	C++	
5.	TypeScript	default for all fronted projects

1.2. Databases

Lp.	Name	Type
1.	PostgreSQL	Open Source, Relational Database Management System
2.	Redis	Open Source, NoSQL, In-memory Key-Value Store
3.	Cassandra	Open Source, NoSQL
6	SQLite	Embedded Relation Database

1.3. Message brokers

Lp.	Name	Type
1.	RabbitMQ	Open Source

1.4. Logs and monitoring

Lp.	Name	Type
1.	OpenSearch	Open Source, Distributed Search Engine
2.	Grafana	Open Source, Data Visualization and Monitoring Tool
3.	Sentry	Open Source, Error Tracking & Performance Monitoring
4.	Prometheus	Open Source, Monitoring & Alerting System
5.	Thanos	Open Source, Distributed Storage and Querying System, Prometheus extension
6.	Opsgenie	SaaS, Incident Management Platform
7.	Zabbix	OpenSource, Monitoring Platform
8.	Pingdom	SaaS, Website Monitoring Service
9.	Rundeck	Open Source, Job Scheduling and Automation Platform
10.	Netbox	Open Source, Network Documentation and IPAM

1.5. Frameworks

Lp.	Name	Type
1.	Angular	Open Source, Frontend Web Framework
2.	Django	OpenSource, Web Framework
3..	FastAPI	Open Source, Web Framework for APIs
4..	pytest	Open Source, Testing Framework
5..	TailwindCSS	OpenSource, Utility-First CSS Framework
6..	Gin	OpenSource, Go Web Framework
7..	SQLAlchemy	Open Source, Python SQL Toolkit and ORM
8..	GORM	Open Source, Go ORM Library
9..	Alembic	Open Source, Database Migration Tool

1.6. IAM and Secrets Management

Lp.	Name	Type
1.	Keycloak	Open Source
2.	OPA	Open Source, Policy Engine
3.	OpenStack Keystone	Open Source, Identity Service
4.	HashiCorp Vault	Open Source, Secrets Management Platform
5.	Fernet	Open Source, Symmetric Encryption Library
6.	OpenBao	OpenSource, Secrets Management Platform

1.7. Networking

Lp.	Name	Type
1.	Envoy	Open Source, Service Proxy
2.	OpenStack Neutron	Open Source, Network Service
3.	Nginx	Open Source, Web Server/Reverse Proxy

1.8. Deployment

Lp.	Name	Type
1.	Docker*	Open Source, Containerization Platform
2.	Kubernetes	Open Source, Container Orchestration Platform
3.	Helm	Open Source, Package Manager
4.	Rancher	Open Source, Kubernetes Management Platform
5.	Ansible	Open Source, Configuration Management and Automation Platform
6.	Terraform	An IAC tool, used to automate various infrastructure tasks
7.	Kaniko	Open Source, Container Image Builder
8.	Jsonnet	Open Source, Data Templating Language

9.	PyPI (Python Package Index)	Open Source, Package Repository
----	-----------------------------	---------------------------------

***when Kubernetes cannot be used**

1.9. Cloud

Lp.	Name	Type
1.	OpenStack	Open Source, Cloud Computing Platform
2.	OpenStack Cinder	OpenSource, Block Storage Service
3.	OpenStack Nova	Open Source, Compute Service
4.	OpenStack Glance	Open Source, Image Service
5.	OpenStack Barbican	Open Source, Key Management Service
6.	OpenStack Octavia	Open Source, Load Balancing Service
7.	OpenStack Heat	Open Source, Orchestration Service

1.10. Storage

Lp.	Name	Type
1.	Ceph	Open Source, Distributed Storage System
2.	Proxmox	Open Source, Virtualization Platform

1.11. CI/CD

Lp.	Name	Type
1.	GitLab CI	Open Source, Continuous Integration
2.	ArgoCD	Open Source, GitOps Deployment Tool

1.12. Observability & Tracing

Lp.	Name	Type
1.	OpenTelemetry	Open Source, Telemetry Collection
2.	Elastic APM	Open Source/Commercial, APM Tool
3.	Struktlog	Open Source, Structured Logging Library
4.	ZeroLog	Open Source, Zero Allocation JSON Logger
5.	Proc-logger	CloudFerro Custom, Process-Level Logging
6.	Cf-logging	CloudFerro Custom, Standardized Logging Framework

1.13. Testing & Quality

Lp.	Name	Type
1.	SonarQube	Open Source / Commercial, Static Code Analysis Tool
2.	Python + Pytest	
3.	Python + Robot Framework	
4.	Python + BDD	

5.	Python + Playwright	
6.	Postman	
7.	K6 – for performance	
8.	Testify	Open Source, Go Testing Framework

2. Dostawca może proponować komponenty oparte o technologie nie znajdującą się na powyższej liście, przy czym decyzja o akceptacji danego rozwiązania wymaga pisemnej akceptacji Zamawiającego.

Załącznik 4.1e – Wymagania szkoleniowe

1. Zakres merytoryczny szkolenia

Szkolenie powinno kompleksowo przygotować zespół Zamawiającego do:

- codziennej obsługi systemu,
- jego utrzymania i zarządzania,
- rozwiązywania problemów,
- ewentualnego rozwijania (jeśli dopuszczalne).

Wymagane są **dwa główne moduły**:

a) Szkolenie funkcjonalne (użytkowe)

Skierowane do operatorów Systemu / Portalu (głównie Customer Support):

- omówienie interfejsu i logiki działania aplikacji,
- konfiguracja podstawowa (np. tworzenie użytkowników, reguł, polityk (jeśli dotyczy)),
- najczęstsze przypadki użycia i scenariusze biznesowe,
- obsługa błędów i typowych sytuacji wyjątkowych,
- przykłady runbooków;
- rejestrowanie i śledzenie zdarzeń/systemów.

b) Szkolenie techniczne (administracyjne)

Dla administratorów, Zespołu DevOps i osób odpowiedzialnych za utrzymanie:

- szczegółowy przegląd architektury i komponentów,
- konfiguracja zaawansowana i deployment,
- mechanizmy backupu i odtwarzania danych,
- integracje z innymi systemami (LDAP, SSO, SIEM, monitoring, itp.),
- analiza logów i debugowanie,
- aktualizacje, migracje, zarządzanie wersjami.

2. Czas trwania i struktura szkolenia

a) Minimalny czas trwania

Minimalne wymogi:

- szkolenie użytkowe: min. **4 godziny (przeprowadzone dla min 3 grup użytkowników, w tym w języku angielskim)**
- szkolenie administracyjne: min. **8 godzin**, z czego min. 50% powinno być przeznaczone na **praktyczne ćwiczenia**

b) Podział tematyczny

Szkolenie musi być zorganizowane w **moduły tematyczne**, z jasno określonymi celami.

Przykładowo:

- dzień 1: instalacja i konfiguracja,
- dzień 2: bezpieczeństwo i monitoring,
- dzień 3: backup i recovery + testy.

c) Forma prowadzenia

- zdalnie lub stacjonarnie – wg ustaleń z Zamawiającym,
- obowiązkowo **nagrywane** i udostępnione,
- **interaktywne** – uczestnicy muszą mieć możliwość wykonywania ćwiczeń praktycznych,
- możliwość zadawania pytań w trakcie i po.

3. Materiały szkoleniowe

Dostawca jest zobowiązany do przekazania:

- prezentacji ze szkolenia (w formacie edytowalnym i PDF),
- dokumentacji wdrożeniowej i administracyjnej,
- przewodników „krok po kroku” (how-to, quick start, checklisty),
- materiałów ćwiczeniowych do warsztatów,
- przykładów konfiguracji / skryptów / gotowych szablonów.

Materiały muszą być przygotowane:

- w języku polskim **lub** angielskim (wg decyzji Zamawiającego),
- w wersji **elektronicznej**, z prawem do wewnętrznego użytku i edycji.

4. Weryfikacja skuteczności szkolenia

a) Test wiedzy / zadanie końcowe

Zaleca się, by uczestnicy wykonali na koniec:

- test wiedzy (online lub pisemny),
- zadanie praktyczne sprawdzające umiejętność samodzielnej konfiguracji/systemowego działania.

b) Checklista szkoleniowa

- Dostawca przedstawi listę zagadnień objętych szkoleniem,
- Zamawiający potwierdzi ich faktyczną realizację w formie pisemnej.

5. Kwalifikacje prowadzącego

Prowadzący szkolenie musi:

- mieć udokumentowane doświadczenie z daną technologią,
- biegle posługiwać się językiem szkoleniowym (PL lub EN),
- być dostępny podczas szkolenia na pytania, dyskusje i wyjaśnienia.

6. Wsparcie poszkoleniowe

Wymagane jest zapewnienie przez Dostawcę:

- **konsultacji poszkoleniowych** przez minimum 10 godzin w okresie 2 tygodni od szkolenia,
- możliwość zadawania pytań mailowo / przez komunikator (np. Teams, Slack),
- opcjonalnie: 1 spotkanie Q&A online po tygodniu od szkolenia.

7. Potwierdzenie realizacji i dokumentacja

Po zakończeniu szkolenia Dostawca zobowiązany jest dostarczyć:

- listę obecności uczestników,
- raport z realizacji programu,
- dostęp do nagrania szkolenia i materiałów.