|  |
| --- |
| Projekt studencki |

Miejsce, data i czas spotkania: **MS Teams,** dnia: **08.03.2024 (piątek)** od: 8:15 do: 10:30

**Uczestnicy:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **LP** | **Rola / Stanowisko** | **email** | **Zadanie** | **Firma** |
|  | Student – **Project Manager** (Zespół Zapp) | Marcel Hinc, 164382@student.uwm.edu.pl | 1 | Uniwersytet Warmińsko-Mazurski w Olsztynie |
|  | Student (Zespół Zapp) | Markowska Kinga, 164407@student.uwm.edu.pl | 1 | Uniwersytet Warmińsko-Mazurski w Olsztynie |
|  | Student (Zespół Zapp) | Drząszcz Krystian, 164361@student.uwm.edu.pl | 1 | Uniwersytet Warmińsko-Mazurski w Olsztynie |
|  | Student (Zespół Zapp) | Franczak Dominik, 164367@student.uwm.edu.pl | 1 | Uniwersytet Warmińsko-Mazurski w Olsztynie |
|  | Student - **Project Manager** (Siemsony Mielsony) | Jakub Pisarski, 164419@student.uwm.edu.pl | 2 | Uniwersytet Warmińsko-Mazurski w Olsztynie |
|  | Student (Zespół Siemsony Mielsony) | Minkowski Sebastian 164412@student.uwm.edu.pl | 2 | Uniwersytet Warmińsko-Mazurski w Olsztynie |
|  | Student(Zespół Siemsony Mielsony) | Mróz Kamil 164462@student.uwm.edu.pl | 2 | Uniwersytet Warmińsko-Mazurski w Olsztynie |
|  | Student(Zespół Siemsony Mielsony) | Pietrzak Michał 164418@student.uwm.edu.pl | 2 | Uniwersytet Warmińsko-Mazurski w Olsztynie |
|  | Student(Zespół Siemsony Mielsony) | Powirski Mateusz 164421@student.uwm.edu.pl | 2 | Uniwersytet Warmińsko-Mazurski w Olsztynie |
|  | Koordynator UWM |  | 1,2 | Uniwersytet Warmińsko-Mazurski w Olsztynie |
|  | Koordynator PKPIK | pawel.jastrzebski@it-pkp.pl | 1,2 | PKP Informatyka sp. z o.o. |
|  | Architekt | ? | 1,2 | PKP Informatyka sp. z o.o. |
|  | Deweloper | ? | 1,2 | PKP Informatyka sp. z o.o. |
|  | Prelegent PKPIK | adam.lozowski@it-pkp.pl | 1,2 | PKP Informatyka sp. z o.o. |

|  |  |
| --- | --- |
| **I** | **Zadania** |
|  | **Zadanie 1 (Aplikacyjne) :**  **Monitoring utylizacji oprogramowania w rozliczeniu miesięcznym – modyfikacja agenta, dodanie pluginu**  Opis stanu obecnego : system umożliwia inwentaryzację zainstalowanego oprogramowania na stacji roboczej i przeprowadzenie bilansu przydzielonych licencji    Opis stanu docelowego: wymagane jest zapewnienie rozliczalności w czasie użycia oprogramowania na stacji roboczej. |

|  |  |
| --- | --- |
|  | **Zadanie 2 (Infrastrukturalne) :**  **[Podstawowe] Inwentaryzacja i detekcja sprzętu z użyciem komputera poprzez Agenta, SSH, WMI, WinRM, SNMP**  Opis stanu obecnego : system umożliwia inwentaryzację i detekcję sprzętu z użyciem komputera stacjonarnego  Opis stanu docelowego: wymagane jest :  - konfiguracja agenta  - wykrywanie i inwentaryzacja sprzętu, komponentów, oprogramowania z użyciem SNMP, SSH, WMI, WinRM  - skanowanie sieci  - sformatowanie kolekcji danych  - przedmiot skanowania : telefon komórkowy, domowy router  **[Zaawansowane] Inwentaryzacja i detekcja sprzętu z użyciem pośrednika (Broker) poprzez Agenta, SSH, WMI, WinRM, SNMP**  Opis stanu obecnego : system umożliwia inwentaryzację i detekcję sprzętu z użyciem brokera PC  Opis stanu docelowego: wymagane jest :  - dobór funkcjonalny Brokera sprzętowego  - konfiguracja Brokera sprzętowego  - wykrywanie i inwentaryzacja sprzętu, komponentów, oprogramowania z użyciem SNMP, SSH, WMI, WinRM  - skanowanie sieci  - sformatowanie kolekcji danych  - przedmiot skanowania : router przemysłowy, switch, drukarka sieciowa, kamera IP  Przykładowa architektura :    Przykładowa inwentaryzacja komputera |

|  |  |
| --- | --- |
| **II** | **Opis produktu** |
|  | GLPI to otwarte oprogramowanie służące do zarządzania zasobami informatycznymi w organizacjach. Jest to narzędzie, które umożliwia skuteczne monitorowanie, organizowanie i zarządzanie infrastrukturą IT. Pozwala w centralny sposób, efektywnie zarządzać zasobami IT, zgłoszeniami, Service Desk, projektami, budżetami, zmianami, statystykami, licencjami, oprogramowaniem, jednostkami organizacyjnymi, grupami użytkowników oraz pojedynczymi użytkownikami. |

|  |  |
| --- | --- |
| **III** | **Architektura** |
|  | **Linux – Ubuntu 22.0.4**  **Opis:** kompletna dystrybucja systemu operacyjnego GNU/Linux  **Rola:** System operacyjny |
|  | **Apache 2**  **Opis:** otwarte i darmowe oprogramowanie serwera HTTP  **Rola:** Serwer aplikacyjny |
|  | **MySQL lub MariaDB**  **Opis:** Baza danych MySQL lub MariaDB jest używana do przechowywania danych systemu GLPI.  **Rola:** Baza danych przechowuje informacje dotyczące urządzeń, incydentów, użytkowników i innych aspektów zarządzania  zasobami IT. |
|  | **PHP 8.1**  **Opis:** GLPI jest napisane głównie w języku PHP.  **Rola:** PHP pełni rolę głównego języka programowania do implementacji logiki biznesowej, obsługi zapytań HTTP oraz  zarządzania danymi w systemie GLPI. |

|  |  |
| --- | --- |
| **IV** | **Technologie** |
|  | **HTML, CSS, JavaScript**  Opis: GLPI wykorzystuje technologie front-endowe, takie jak HTML, CSS i JavaScript, do renderowania interfejsu użytkownika.  Rola: Zapewniają one interaktywność, responsywność i atrakcyjność interfejsu użytkownika. |
|  | **XML**  **Opis:** Rozszerzalny język znaczników, prosty format tekstowy przeznaczony do wymiany danych  **Rola:** Do generowania raportów i zbierania danych inventory |
|  | **Perl**  **Opis:** Interpretowany, wysokopoziomowy język skryptowy programowania.  **Rola:** Integracje, zadania automatyzujące. |
|  | **jQuery**  **Opis:** GLPI korzysta z biblioteki jQuery do ułatwienia manipulacji i obsługi zdarzeń w warstwie klienta.  **Rola:** Umożliwia szybszy rozwój funkcji interaktywnych bezpośrednio w języku JavaScript. |
|  | **Bootstrap**  **Opis:** Bootstrap to framework front-endowy, który jest używany do budowy responsywnego i jednolitego interfejsu użytkownika.  **Rola:** Ułatwia projektowanie interfejsu, dostarczając gotowe komponenty i układy. |
|  | **Symfony**  **Opis:** GLPI wykorzystuje części frameworką Symfony, zwłaszcza Symfony Components, do organizacji kodu, zarządzania zależnościami i obsługi wielu aspektów aplikacji.  **Rola:** Zapewnia strukturalne podejście do programowania i ułatwia konserwację kodu. |
|  | **Composer**  **Opis:** Composer jest menedżerem zależności dla PHP, który jest używany w GLPI do zarządzania zewnętrznymi bibliotekami i narzędziami.  **Rola:** Ułatwia kontrolę wersji i zarządzanie zależnościami. |
|  | **Twig**  **Opis:** Twig to silnik szablonów używany w GLPI do renderowania widoków.  **Rola:** Umożliwia separację logiki biznesowej od warstwy prezentacji. |
|  | **PHPUnit**  **Opis:** PHPUnit jest frameworkiem do testowania jednostkowego dla PHP, który jest wykorzystywany do testowania kodu GLPI.  **Rola:** Zapewnia automatyzację testów, co pomaga w utrzymaniu stabilności i jakości kodu. |
|  | **Docker**  **Opis:** Docker jest używany do konteneryzacji aplikacji GLPI, co ułatwia wdrażanie i zarządzanie środowiskiem deweloperskim oraz produkcyjnym.  **Rola:** Ułatwia przenośność i izolację aplikacji |

|  |  |
| --- | --- |
| **IV** | **Dokumentacja** |
|  | Instalacja : [link](https://glpi-install.readthedocs.io/en/latest/install/index.html) |
|  | Model bazy danych : [link](https://glpi-developer-documentation.readthedocs.io/en/master/devapi/database/dbmodel.html) |
|  | Tworzenie wtyczek : [link](https://glpi-developer-documentation.readthedocs.io/en/master/plugins/index.html) |
|  | Przykładowe wtyczki : [link](https://github.com/pluginsGLPI) |
|  | Kod agenta : [link](https://github.com/glpi-project/glpi-agent/releases) |
|  | Komunikacja agent – serwer : [link](https://glpi-agent.readthedocs.io/en/latest/plugins/inventory-server-plugin.html) |
|  | Instalacja i konfiguracja LAB |

|  |  |
| --- | --- |
| **Lp.** | **Ustalenia** |
| **I.** | **Część koncepcyjna i organizacyjna** |
|  |  |
|  |  |
| **II.** | **Część funkcjonalna** |
| 1. |  |
| 2. |  |

**Zadania do realizacji :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Lp.** | **Zagadnienie** | **Zadanie** | **Realizacja** | **Termin real.** |
|  | Instalacja środowiska deweloperskiego (fakultatywnie) | 1,2 |  |  |
|  | Instalacja agentów na urządzeniach | 1,2 |  |  |
|  | Architektura logiczna i fizyczna w narzędziu do modelowania BPMN | 1,2 |  |  |
|  | Inwentory + Discovery zasobów IT | 2 |  |  |
|  | Modyfikacja GUI użytkownika + security | 1 |  |  |
|  | Utworzenie słowników (lokalizacje, statusy, komponenty) | 2 |  |  |
|  | Inicjalne zasilenie danymi z csv (REST API + Plugin DataInjection) | 1 |  |  |
|  | Rejestracja zdarzeń Windows (raport z utylizacji procesów) | 1 |  |  |
|  | Utworzenie nowej wtyczki i jej wyprofilowanie funkcjonalne | 1 |  |  |
|  | Modyfikacja agenta | 1 |  |  |
|  | Konfiguracja brokera + instalacja agenta | 2 |  |  |
|  | Obsługa zdarzeń w bazie danych (insert, trigery) po każdym insercie sprawdza czy mamy stare selecty, które można usunąć (np. na podstawie czasu retencji danych) | 1 |  |  |
|  | Skanowanie sprzętu SSH, SNMP, WMI, WinRM | 2 |  |  |
|  | Skanowanie sieci | 2 |  |  |
|  | Skanowanie VMWare | 2 |  |  |
|  | Testowanie i rozwijanie rozwiązania | 1,2 |  |  |
|  | Dokumentacja i Prezentacja | 1,2 |  |  |

**Dodatkowe materiały:**