**Opis specyfikacji systemu - uzasadnienie i cele**

**1. Wybór technologii**

**1.1. Frontend: Vue.js + Tailwind CSS**

**Uzasadnienie: Vue.js został wybrany ze względu na jego lekkość, elastyczność i łatwość integracji z innymi bibliotekami. Composition API wprowadzone w Vue 3 umożliwia bardziej modułowe i czytelne pisanie kodu, co ułatwi współpracę w zespole. Tailwind CSS znacząco przyspiesza proces tworzenia interfejsu dzięki gotowym klasom, zapewniając jednocześnie spójny wygląd aplikacji.**

**Korzyści:**

* **Szybsze tworzenie interfejsu użytkownika**
* **Bardziej czytelny i łatwiejszy w utrzymaniu kod**
* **Lepsza wydajność aplikacji**
* **Mniejszy rozmiar końcowej aplikacji w porównaniu do innych frameworków**

**1.2. Backend: Node.js + NestJS**

**Uzasadnienie: NestJS dostarcza solidną strukturę do budowy skalowalnych aplikacji serwerowych w TypeScript. Framework ten wspiera modułową architekturę, dependency injection i jest zainspirowany przez Angular, co ułatwia organizację kodu. Dodatkowo, dzięki wykorzystaniu TypeScript, zwiększa się bezpieczeństwo aplikacji poprzez statyczne typowanie.**

**Korzyści:**

* **Spójna i skalowalna architektura**
* **Mocne wsparcie dla TypeScript**
* **Wbudowane mechanizmy zabezpieczeń**
* **Łatwa integracja z różnymi bazami danych i usługami zewnętrznymi**

**1.3. Baza danych: PostgreSQL + Prisma ORM**

**Uzasadnienie: PostgreSQL to solidna, relacyjna baza danych o otwartym kodzie, która doskonale sprawdza się w aplikacjach z kompleksowym modelem danych. Prisma ORM zapewnia typebezpieczne zapytania, automatyczną migrację i czytelny sposób definiowania modelu danych, co znacząco przyspiesza rozwój aplikacji.**

**Korzyści:**

* **Niezawodne przechowywanie danych**
* **Wsparcie dla zaawansowanych zapytań i relacji**
* **Typebezpieczne operacje na bazie danych**
* **Łatwiejsza migracja i zarządzanie schematem**

**1.4. Autoryzacja: MS Teams i Active Directory**

**Uzasadnienie: Integracja z MS Teams i Active Directory uczelni eliminuje potrzebę tworzenia osobnego systemu rejestracji i logowania. Studenci i nauczyciele mogą korzystać z tych samych danych uwierzytelniających, których używają na co dzień, co upraszcza dostęp do aplikacji i zwiększa bezpieczeństwo.**

**Korzyści:**

* **Brak potrzeby zapamiętywania dodatkowych haseł**
* **Automatyczne przydzielanie ról na podstawie grup w Active Directory**
* **Wyższy poziom bezpieczeństwa dzięki centralnej kontroli dostępu**
* **Łatwiejsza integracja z innymi systemami uczelni**

**2. Architektura systemu**

**2.1. Trójwarstwowa architektura**

**Uzasadnienie: Podział na frontend, backend i bazę danych pozwala na niezależny rozwój każdej warstwy, zwiększa modularność, ułatwia testowanie i umożliwia skalowanie poszczególnych komponentów. Jest to sprawdzony standard w nowoczesnych aplikacjach webowych.**

**Korzyści:**

* **Łatwiejsze zarządzanie złożonością projektu**
* **Możliwość niezależnego skalowania warstw**
* **Lepsza separacja obowiązków w zespole**
* **Łatwiejsze testowanie poszczególnych komponentów**

**2.2. Konteneryzacja z Docker**

**Uzasadnienie: Docker zapewnia jednolite środowisko dla rozwoju, testowania i produkcji, eliminując problemy typu "u mnie działa". Konteneryzacja ułatwia również wdrażanie aplikacji i zarządzanie zależnościami.**

**Korzyści:**

* **Spójne środowisko na wszystkich etapach rozwoju**
* **Łatwiejsze wdrażanie i aktualizacje**
* **Izolacja komponentów zwiększająca bezpieczeństwo**
* **Efektywniejsze wykorzystanie zasobów serwera**

**3. Moduły systemu**

**3.1. Moduł Użytkowników**

**Uzasadnienie: System ról i uprawnień jest kluczowy dla kontroli dostępu do różnych funkcjonalności aplikacji. Podział na administratorów, nauczycieli, studentów i gości odpowiada rzeczywistym rolom w środowisku akademickim i zapewnia odpowiedni poziom kontroli.**

**Korzyści:**

* **Dokładna kontrola dostępu do funkcji systemu**
* **Dostosowany interfejs dla różnych typów użytkowników**
* **Zgodność z procesami uczelnianymi**
* **Prostsze zarządzanie uprawnieniami**

**3.2. Moduł Projektów**

**Uzasadnienie: Centralna funkcjonalność systemu, umożliwiająca zarządzanie całym cyklem życia projektu - od utworzenia, przez recenzję, aż po publikację. Rozbudowane metadane i możliwość kategoryzacji ułatwiają organizację i wyszukiwanie projektów.**

**Korzyści:**

* **Kompleksowe zarządzanie projektami w jednym miejscu**
* **Przejrzysta ścieżka zatwierdzania projektów**
* **Łatwe wyszukiwanie i filtrowanie projektów**
* **Możliwość przeglądania projektów bez pobierania**

**3.3. Moduł Administracyjny**

**Uzasadnienie: Zapewnia narzędzia do zarządzania całym systemem, pozwalając administratorom na kontrolę użytkowników, kategorii i treści. Generowanie raportów umożliwia analizę wykorzystania systemu i podejmowanie decyzji na podstawie danych.**

**Korzyści:**

* **Centralne zarządzanie całym systemem**
* **Możliwość szybkiej reakcji na problemy**
* **Analityka wykorzystania systemu**
* **Utrzymanie jakości treści i spójności danych**

**4. Model danych**

**4.1. Kompleksowy model relacyjny**

**Uzasadnienie: Rozbudowany model danych z wieloma powiązanymi encjami umożliwia przechowywanie wszystkich niezbędnych informacji o projektach, użytkownikach i ich interakcjach. Relacyjna struktura zapewnia integralność danych i umożliwia złożone zapytania.**

**Korzyści:**

* **Kompletne i spójne dane w całym systemie**
* **Możliwość zaawansowanego wyszukiwania i filtrowania**
* **Łatwe rozszerzanie modelu o nowe elementy**
* **Efektywne przechowywanie danych minimalizujące redundancję**

**4.2. Hierarchia kategorii**

**Uzasadnienie: Struktura kategorii z możliwością zagnieżdżania pozwala na tworzenie logicznej organizacji projektów, odzwierciedlającej strukturę kierunków i przedmiotów na uczelni.**

**Korzyści:**

* **Intuicyjna nawigacja po projektach**
* **Lepsze grupowanie podobnych projektów**
* **Możliwość dostosowania do struktury organizacyjnej uczelni**
* **Ułatwione wyszukiwanie tematyczne**

**5. Interfejs użytkownika**

**5.1. Responsywny design**

**Uzasadnienie: Dostosowanie interfejsu do różnych urządzeń zwiększa dostępność aplikacji i pozwala studentom i nauczycielom korzystać z systemu zarówno na komputerach, jak i urządzeniach mobilnych.**

**Korzyści:**

* **Dostęp do systemu z dowolnego urządzenia**
* **Konsystentne doświadczenie użytkownika**
* **Większa wygoda użytkowania**
* **Lepsza dostępność dla osób z niepełnosprawnościami**

**5.2. Intuicyjna nawigacja**

**Uzasadnienie: Przejrzysty układ interfejsu z logicznym grupowaniem funkcji przyspiesza naukę korzystania z systemu i zwiększa efektywność pracy użytkowników.**

**Korzyści:**

* **Krótszy czas potrzebny na przyswojenie systemu**
* **Mniejsza liczba błędów użytkownika**
* **Wyższa satysfakcja z użytkowania**
* **Mniejsze zapotrzebowanie na wsparcie techniczne**

**6. Bezpieczeństwo**

**6.1. Wielopoziomowe zabezpieczenia**

**Uzasadnienie: Kompleksowe podejście do bezpieczeństwa, obejmujące autoryzację, zabezpieczenia infrastruktury i ochronę danych, minimalizuje ryzyko nieautoryzowanego dostępu i utraty danych.**

**Korzyści:**

* **Ochrona wrażliwych danych uczelnianych**
* **Zgodność z wymogami prawnymi dotyczącymi ochrony danych**
* **Zwiększone zaufanie użytkowników**
* **Minimalizacja ryzyka incydentów bezpieczeństwa**

**6.2. HTTPS i SSL**

**Uzasadnienie: Szyfrowanie komunikacji między klientem a serwerem chroni dane przesyłane przez sieć i potwierdza autentyczność serwera.**

**Korzyści:**

* **Ochrona przed podsłuchiwaniem komunikacji**
* **Weryfikacja tożsamości serwera**
* **Zgodność z nowoczesnymi standardami bezpieczeństwa**
* **Lepsze pozycjonowanie w wyszukiwarkach**

**7. Etapy rozwoju i harmonogram**

**7.1. Podejście iteracyjne**

**Uzasadnienie: Podział projektu na etapy umożliwia wcześniejsze dostarczanie funkcjonalności, zbieranie informacji zwrotnej i iteracyjne udoskonalanie systemu.**

**Korzyści:**

* **Szybsze dostarczenie podstawowej funkcjonalności**
* **Możliwość wczesnej weryfikacji założeń**
* **Łatwiejsze zarządzanie ryzykiem i zasobami**
* **Elastyczność w dostosowywaniu się do zmieniających się wymagań**

**7.2. Podział zadań według specjalizacji**

**Uzasadnienie: Przydzielenie zadań zgodnie z kompetencjami członków zespołu maksymalizuje efektywność i jakość wykonania poszczególnych elementów systemu.**

**Korzyści:**

* **Lepsze wykorzystanie umiejętności zespołu**
* **Wyższa jakość poszczególnych komponentów**
* **Szybszy rozwój projektu**
* **Jaśniejsza odpowiedzialność za poszczególne elementy**

**Podsumowanie**

**Przyjęta specyfikacja została zaprojektowana z myślą o stworzeniu nowoczesnego, bezpiecznego i użytecznego systemu zarządzania projektami studenckimi. Wybrane technologie, architektura i podejście do rozwoju mają na celu maksymalizację wydajności pracy zespołu, jakości produktu końcowego i satysfakcji użytkowników.**

**System będzie wspierał cały proces tworzenia i zarządzania projektami studenckimi, zapewniając odpowiedni poziom kontroli jakości, łatwość wyszukiwania i dostępu do zasobów, przy jednoczesnym zachowaniu bezpieczeństwa danych i zgodności z procesami uczelnianymi.**