Specyfikacja wymagań

**v.2025-06-10**

JLibrary

Anzhelika Shtepa

Katsiaryna Novikava

Spis treści

[1. Wstęp 3](#_Toc201868398)

[2. Cele specyfikacji 3](#_Toc201868399)

[3. Zakres specyfikacji 3](#_Toc201868400)

[4. Wymagania funkcjonalne 4](#_Toc201868401)

[5. Wymagania niefunkcjonalne 5](#_Toc201868402)

[5.1 Interfejsy 5](#_Toc201868403)

[5.1.1 Interfejsy użytkownika 5](#_Toc201868404)

[5.1.2 Interfejsy sprzętowe 5](#_Toc201868405)

[5.1.3 Interfejsy komunikacyjne 6](#_Toc201868406)

[5.1.4 Interfejsy programowe 6](#_Toc201868407)

[5.2 Warunki serwisowania (support) 6](#_Toc201868408)

[5.3 Ograniczenia architektury systemu 6](#_Toc201868409)

[5.4 Dokumentacja użytkownika 7](#_Toc201868410)

[5.5 Wymagania licencyjne 7](#_Toc201868411)

[5.6 Prawa autorskie i inne zagadnienia prawne 7](#_Toc201868412)

[6. Wymagania dotyczące jakości modelowanego systemu 7](#_Toc201868413)

[7. Bazy danych 8](#_Toc201868414)

[8. Stosowane standardy 9](#_Toc201868415)

[9. Opis więzów 9](#_Toc201868416)

# Wstęp

Niniejszy dokument zawiera szczegółową specyfikację wymagań dla aplikacji desktopowej JLibrary służącej do zarządzania działalnością biblioteki.

Dokument ten określa wymagania funkcjonalne i niefunkcjonalne systemu, przedstawia interfejsy, ograniczenia architektury, wymagania dotyczące jakości, standardy oraz inne elementy niezbędne do poprawnej implementacji i późniejszego użytkowania systemu.

Celem dokumentu jest dostarczenie kompletnej i jednoznacznej podstawy do realizacji systemu informatycznego, który umożliwia obsługę procesów bibliotecznych: rejestrację i logowanie użytkowników, przeglądanie i zamawianie książek, zarządzanie użytkownikami i zasobami przez personel biblioteki, a także generowanie raportów i statystyk.

Zakres dokumentu obejmuje zarówno ogólne założenia projektowe, jak i techniczne oraz organizacyjne aspekty implementacji systemu. Specyfikacja może być wykorzystywana przez zespół projektowy, zamawiającego, testerów oraz przyszłych administratorów systemu.

# Cele specyfikacji

Celem niniejszej specyfikacji jest przedstawienie wszystkich wymagań dotyczących projektowanego systemu JLibrary w sposób jednoznaczny, zrozumiały i kompletny, tak aby umożliwić jego skuteczne zaprojektowanie, implementację, testowanie oraz późniejsze użytkowanie i utrzymanie.

Dokument ten ma na celu:

* Zdefiniowanie funkcjonalności, które system powinien oferować użytkownikom końcowym (czytelnikom, bibliotekarzom i administratorom),
* Określenie wymagań niefunkcjonalnych, takich jak dostępność, bezpieczeństwo, wydajność czy łatwość obsługi,
* Ustalenie interfejsów i ograniczeń, jakie powinny być wzięte pod uwagę podczas implementacji,
* Sprecyzowanie oczekiwań biznesowych względem systemu oraz warunków jego integracji w środowisku użytkownika.

System JLibrary ma pełnić rolę wszechstronnej aplikacji do zarządzania zasobami bibliotecznymi, która zapewni:

* Rejestrację i logowanie użytkowników (z przypisaną rolą),
* Wyszukiwanie i przeglądanie dostępnych książek,
* Składanie i przetwarzanie zamówień na książki,
* Zarządzanie użytkownikami oraz księgozbiorem (dodawanie/usuwanie),
* Wydawanie i zwracanie książek,
* Generowanie statystyk i raportów dla celów administracyjnych.

Celem końcowym specyfikacji jest umożliwienie stworzenia stabilnego, użytecznego i łatwego w utrzymaniu systemu wspierającego codzienne działania w środowisku bibliotecznym.

# Zakres specyfikacji

Specyfikacja dotyczy aplikacji JLibrary, która działa w architekturze klient–serwer i umożliwia użytkownikom zdalny dostęp do zasobów bibliotecznych przez sieć lokalną lub Internet.

Zakres funkcjonalny obejmuje:

* Rejestrację i logowanie użytkowników,
* Zdalne przeglądanie i wyszukiwanie książek w bazie danych biblioteki,
* Składanie zamówień na książki przez czytelników — z dowolnego komputera,
* Obsługę zamówień przez bibliotekarzy (zatwierdzanie, odrzucanie, wydawanie książek),
* Zarządzanie kontami użytkowników przez administratora (dodawanie, usuwanie),
* Dodawanie książek ręcznie lub przez podanie numeru ISBN,
* Zwracanie książek oraz ewidencję wypożyczeń.

Charakterystyka użytkowników:

* Czytelnik – korzysta z aplikacji zdalnie, loguje się i przegląda katalog, składa zamówienia,
* Bibliotekarz – pracuje na stanowisku w bibliotece, przetwarza zamówienia, wydaje i przyjmuje książki,
* Administrator – zarządza systemem, kontami i nadzoruje poprawność działania aplikacji.

Wymagania niefunkcjonalne zawarte w zakresie:

* Interfejs graficzny oparty na JavaFX powinien być dostępny na urządzeniach użytkowników końcowych,
* System powinien zapewniać wielodostęp i odpowiednią synchronizację operacji,
* Interfejs powinien być intuicyjny, dostępny w języku polskim i dopasowany do różnych ról,
* Wymagany jest dostęp do sieci wewnętrznej lub Internetu,
* Wszystkie dane są przechowywane centralnie, na serwerze zarządzanym przez bibliotekę,
* Aplikacja klienta wymaga środowiska Java 17+ oraz połączenia sieciowego z serwerem.

# Wymagania funkcjonalne

System JLibrary musi zapewniać zestaw funkcjonalności dostosowanych do trzech typów użytkowników: czytelnik, bibliotekarz oraz administrator. Każdy z użytkowników po zalogowaniu się do systemu powinien mieć dostęp tylko do funkcji przypisanych do swojej roli.

* 1. Rejestracja i logowanie użytkownika

System umożliwia utworzenie nowego konta (rejestracja) poprzez formularz zawierający login, hasło. Podczas rejestracji użytkownik domyślnie otrzymuje rolę czytelnik. Rolę bibliotekarza może przypisać tylko administrator.

System umożliwia logowanie na podstawie loginu i hasła. W przypadku błędnych danych system wyświetla komunikat o błędzie. Dane logowania są weryfikowane względem bazy danych.

* 1. Przeglądanie i wyszukiwanie książek

Czytelnik może przeglądać listę książek dostępnych w systemie, wraz z ich podstawowymi danymi (tytuł, autor, dostępność, ISBN). Użytkownik może filtrować i wyszukiwać książki według tytułu, autora lub numeru ISBN.

System powinien wyświetlać aktualną dostępność książki.

* 1. Składanie zamówień

Po zalogowaniu, czytelnik może złożyć zamówienie na wybraną książkę. System sprawdza, czy książka jest dostępna (czy nie została już wypożyczona). Zamówienie trafia do panelu bibliotekarza jako "oczekujące". Czytelnik może przeglądać historię swoich zamówień i status (oczekujące, zatwierdzone, odrzucone).

* 1. Przetwarzanie zamówień przez bibliotekarza

Bibliotekarz ma dostęp do listy wszystkich zamówień oczekujących na decyzję. Może zatwierdzić zamówienie (książka zostaje oznaczona jako wypożyczona), lub je odrzucić. Po zatwierdzeniu, system przypisuje wypożyczenie do konkretnego użytkownika i aktualizuje liczbę dostępnych egzemplarzy. System powinien rejestrować datę wypożyczenia i termin zwrotu.

* 1. Wydawanie i zwracanie książek

Bibliotekarz może oznaczyć książkę jako "wydana" (fizyczne wydanie). Czytelnik przynosi książkę do biblioteki, a bibliotekarz oznacza zwrot w systemie. Po zwrocie książki status zmienia się na "dostępna".

* 1. Zarządzanie książkami

Bibliotekarz może dodawać nowe książki do systemu na dwa sposoby:

* Poprzez formularz ręczny: tytuł, autor, ISBN,
* Poprzez podanie numeru ISBN, po którym system pobiera dane z zewnętrznej bazy.

Bibliotekarz może usuwać książki z systemu (np. uszkodzone, nieaktualne).

* 1. Zarządzanie użytkownikami

Administrator ma dostęp do pełnej listy użytkowników:

* Może dodawać nowych użytkowników ręcznie (bibliotekarze),
* Może usuwać konta użytkowników (np. nieaktywnych).
  1. Statystyki i raporty

Administrator ma dostęp do panelu statystyk zawierającego:

* Liczbę zarejestrowanych użytkowników,
* Liczbę książek w bazie,
* Wypożyczane książki.

# Wymagania niefunkcjonalne

Wymagania niefunkcjonalne odnoszą się do sposobu działania systemu JLibrary – jego jakości, użyteczności, bezpieczeństwa, wydajności i struktury technicznej.

## Interfejsy

## Interfejsy użytkownika

Interfejs użytkownika jest zrealizowany w technologii JavaFX.

Aplikacja zawiera osobne panele graficzne dla każdego typu użytkownika:

* Czytelnik: dostęp do katalogu książek, składanie zamówień, przegląd historii,
* Bibliotekarz: panel obsługi zamówień, dodawania i usuwania książek, zwroty,
* Administrator: zarządzanie użytkownikami i dostęp do statystyk.

Interfejs musi być:

* Intuicyjny – łatwy do nauki bez szkolenia,
* Responsywny – szybko reagujący na działania użytkownika,
* Zgodny z zasadami UX/UI – logiczna nawigacja, spójne oznaczenia,
* W języku polskim – wszystkie komunikaty, przyciski i formularze.

## Interfejsy sprzętowe

System nie wymaga specjalistycznego sprzętu poza komputerem klasy PC z systemem Windows, Linux lub macOS.

Minimalne wymagania sprzętowe:

* Procesor 2 GHz,
* 4 GB RAM,
* 500 MB wolnego miejsca na dysku,
* Połączenie sieciowe (Ethernet lub Wi-Fi).

## Interfejsy komunikacyjne

Aplikacja działa w architekturze klient–serwer.

Klient komunikuje się z serwerem poprzez:

* protokół HTTP/HTTPS (jeśli REST API),
* lub gniazda TCP (jeśli sockety).

Serwer aplikacji powinien być dostępny w sieci LAN lub publicznej, przez adres IP lub nazwę hosta.

## Interfejsy programowe

System JLibrary korzysta z kilku warstw interfejsów programowych służących do komunikacji pomiędzy komponentami wewnętrznymi oraz z systemami zewnętrznymi.

Interfejsy wewnętrzne:

Architektura systemu oparta jest na wzorcu MVC z wyraźnym podziałem na warstwy:

* Widok (JavaFX) – interfejs graficzny użytkownika;
* Warstwa usługowa (Services) – zawiera logikę biznesową, walidację, przetwarzanie danych;
* Warstwa dostępu do danych (DAO) – komunikuje się bezpośrednio z relacyjną bazą danych SQL.

Interfejsy pomiędzy warstwami są jasno zdefiniowane (klasy, metody, przekazywanie obiektów modelowych). Komunikacja odbywa się lokalnie w ramach tej samej aplikacji (brak zdalnych wywołań metod).

Integracja z zewnętrznym API:

System integruje się z zewnętrzną usługą API służącą do pobierania danych bibliograficznych książek na podstawie numeru ISBN. Po wpisaniu numeru ISBN w formularzu dodawania książki, system wysyła zapytanie do usługi takiej jak Google Books API lub OpenLibrary API.

Odpowiedź w formacie JSON zawiera informacje takie jak: tytuł, autorzy, rok wydania, wydawnictwo. Dane te są automatycznie wypełniane w formularzu dodawania książki, oszczędzając czas użytkownika. Komunikacja odbywa się przez protokół HTTPS, z wykorzystaniem klucza API. W przypadku błędnej odpowiedzi, braku połączenia lub nieznalezienia książki, system informuje użytkownika i umożliwia wprowadzenie danych ręcznie.

## Warunki serwisowania (support)

System powinien być łatwy do aktualizacji – pliki konfiguracyjne powinny być oddzielone od logiki aplikacji.

Wszystkie błędy powinny być logowane z poziomem szczegółowości (np. INFO, WARNING, ERROR). Dokumentacja dla działu IT powinna zawierać instrukcję instalacji i aktualizacji. Obsługa błędów krytycznych musi umożliwiać szybki restart systemu bez utraty danych.

## Ograniczenia architektury systemu

System napisany w języku Java (min. wersja 17). Zastosowana struktura MVC: oddzielenie warstw logiki biznesowej, danych i interfejsu użytkownika. Projekt zorganizowany jako aplikacja typu Maven. Konwencje kodowania zgodne z Java Code Conventions. Brak zewnętrznych zależności, które wymagają komercyjnych licencji. System nie wspiera działania offline dla użytkowników końcowych – wymagane połączenie z serwerem.

## Dokumentacja użytkownika

Do aplikacji musi być dołączona instrukcja obsługi w formacie PDF, zawierająca:

* Proces instalacji klienta,
* Logowanie i nawigacja po interfejsie,
* Opis ról i funkcji systemu,
* Rozwiązywanie najczęstszych problemów.

Interfejs zawiera kontekstową pomoc (komunikaty).

## Wymagania licencyjne

System objęty jest licencją MIT – użytkownik może go modyfikować i rozpowszechniać z zachowaniem oryginalnych praw autorskich. Zewnętrzne biblioteki (np. JavaFX, JDBC) również muszą być zgodne z licencją open source. Zabrania się integracji z komponentami komercyjnymi bez wykupienia odpowiednich licencji.

## Prawa autorskie i inne zagadnienia prawne

Wszystkie elementy systemu (kod źródłowy, interfejsy, dokumentacja) są objęte prawem autorskim zespołu projektowego. W przypadku użycia komponentów zewnętrznych, należy zachować wymagania licencyjne. (np. podanie autorstwa). System nie przetwarza danych osobowych wrażliwych, ale wymagane jest odpowiednie zabezpieczenie danych kont użytkowników. Odpowiedzialność za utrzymanie systemu po wdrożeniu ponosi administrator wyznaczony przez instytucję klienta.

# Wymagania dotyczące jakości modelowanego systemu

System JLibrary powinien spełniać szereg wymagań jakościowych, które zapewnią jego poprawne działanie, wysoką dostępność i bezpieczeństwo danych. Poniżej przedstawiono szczegółowe wymagania z podziałem na kategorie.

* 1. Niezawodność (Reliability)

System musi działać stabilnie i przewidywalnie w różnych warunkach (różna liczba użytkowników, różna jakość sieci). Wszystkie krytyczne operacje (np. wypożyczenie książki, rejestracja użytkownika) muszą być transakcyjne i odporne na błędy. System powinien obsługiwać wyjątki i awarie (np. przerwanie połączenia z bazą danych) w sposób kontrolowany i bezpieczny. Dane wprowadzone przez użytkownika nie mogą zostać utracone w przypadku awarii aplikacji.

* 1. Dostępność (Availability)

Aplikacja serwerowa powinna być dostępna w trybie ciągłym (24/7), z minimalnym czasem przestoju. Klient aplikacji powinien móc się połączyć z serwerem z dowolnego miejsca (przy założeniu dostępności sieci). System powinien umożliwiać szybki restart bez długotrwałej rekonfiguracji.

* 1. Użyteczność (Usability)

Interfejs użytkownika musi być intuicyjny, zrozumiały i dopasowany do ról użytkowników. Wszystkie formularze powinny zawierać walidację wprowadzanych danych oraz komunikaty błędów w języku polskim. System powinien być prosty w obsłudze również dla użytkowników bez zaawansowanej wiedzy technicznej.

* 1. Wydajność (Efficiency)

Czas odpowiedzi systemu na zapytania użytkownika nie powinien przekraczać 2 sekund w typowych warunkach. Operacje takie jak: logowanie, wyszukiwanie książki, złożenie zamówienia powinny być realizowane w czasie rzeczywistym. System powinien być przystosowany do obsługi wielu jednoczesnych użytkowników (co najmniej 20 sesji równolegle) bez utraty wydajności.

* 1. Obsługiwalność (Maintainability)

Kod źródłowy systemu musi być modularny, zgodny z zasadami Clean Code i dokumentowany. Zmiana jednej funkcjonalności nie powinna wpływać na działanie pozostałych komponentów. System powinien zawierać narzędzia logowania błędów i monitorowania stanu aplikacji (np. konsola logów, pliki dziennika). Proces aktualizacji aplikacji musi być prosty (podmiana plików JAR, migracja bazy).

* 1. Przenośność (Portability)

Aplikacja klienta powinna działać na wszystkich głównych systemach operacyjnych wspierających JVM: Windows, Linux, macOS. System nie jest zależny od specyficznych sterowników ani sprzętu. Możliwość uruchomienia aplikacji z poziomu pendrive’a lub jako pakiet ZIP bez instalatora. Konfiguracja po stronie klienta powinna być minimalna lub zautomatyzowana.

* 1. Bezpieczeństwo danych (Data Security)

Dane wrażliwe (np. dane konta, logi operacji) są zabezpieczone przed nieautoryzowanym dostępem. Komunikacja z serwerem powinna odbywać się z użyciem szyfrowania TLS, jeśli wdrożenie obejmuje połączenia przez Internet. System powinien umożliwiać przypisanie ról oraz dostęp tylko do dozwolonych operacji (autoryzacja).

* 1. Bezpieczne użycie (Safety)

System nie powinien zawierać elementów mogących uszkodzić dane systemowe użytkownika. Aplikacja nie wykonuje żadnych operacji na systemie plików poza własnym katalogiem roboczym. W przypadku nieprawidłowego działania komponentu (np. brak połączenia z bazą) system powinien informować użytkownika i nie kontynuować operacji, które mogą prowadzić do błędów. Obsługa błędów powinna być czytelna i nie powodować zamknięcia aplikacji bez ostrzeżenia.

# Bazy danych

System JLibrary korzysta z relacyjnej bazy danych SQL, przechowywanej i zarządzanej lokalnie na serwerze aplikacji. Baza danych jest podstawowym źródłem informacji o użytkownikach, książkach oraz historii wypożyczeń i zamówień.

* 1. Wymagania ogólne

System wykorzystuje relacyjną bazę danych zgodną z SQL (np. MySQL, PostgreSQL lub SQLite). Baza danych musi być dostępna lokalnie na serwerze aplikacji (np. localhost lub adres IP w sieci LAN). Dostęp do bazy danych odbywa się z poziomu aplikacji przez JDBC. Struktura bazy danych powinna umożliwiać łatwe dodawanie nowych tabel i kolumn bez konieczności przebudowy systemu. Wymagana jest możliwość tworzenia kopii zapasowych oraz migracji danych między środowiskami.

* 1. Struktura danych

Baza danych powinna zawierać co najmniej następujące tabele:

Tabela users – dane użytkowników:

|  |  |  |
| --- | --- | --- |
| Koluma | Typ | Uwagi |
| id | INT, PK, AI | Klucz główny |
| login | VARCHAR(50) | Unikalny login użytkownika |
| password | VARCHAR(255) | Hasłó użytkownika |
| role | ENUM | 'reader', 'librarian', 'admin' |

Tabela books – dane książek:

|  |  |  |
| --- | --- | --- |
| Kolumna | Typ | Uwagi |
| id | INT, PK, AI | Klucz główny |
| title | VARCHAR(255) | Tytuł książki |
| author | VARCHAR(255) | Autor książki |
| is\_available | TINYINT | Dostępność książki |
| isbn | VARCHAR(20) | Unikalny numer ISBN |

Tabela loans – historia wypożyczeń:

|  |  |  |
| --- | --- | --- |
| Kolumna | Typ | Uwagi |
| id | INT, PK, AI | Klucz główny |
| book\_id | INT, FK | ID książki |
| reader\_id | INT, FK | ID czytelnika |
| librarian\_id | INT, FK | ID bibliotekarza |
| loan\_date | DATE | Data wypożyczenia |
| return\_date | DATE (nullable) | Data zwrotu (jeśli zwrócono) |
| is\_returned | TINYINT | Czy zwrócona |
| status | ENUM | 'oczekujące', 'zatwierdzone', 'odrzucone', 'zwrócone' |
| due\_date | DATE | Termin zwrotu |

* 1. Bezpieczeństwo danych

Zapytania do bazy danych muszą być realizowane z wykorzystaniem PreparedStatement w celu zapobiegania atakom typu SQL Injection. Uprawnienia użytkowników muszą być respektowane przy dostępie do danych (np. czytelnik nie może edytować książek). Baza danych powinna mieć regularne kopie zapasowe.

* 1. Wydajność i integralność:

Wszystkie tabele muszą posiadać odpowiednie indeksy, szczególnie dla kolumn używanych w wyszukiwaniach (np. isbn, title, user\_id). Relacje pomiędzy tabelami muszą być spójne i zgodne z zasadami integralności referencyjnej (klucze obce). Operacje wypożyczania, zwrotu i zamawiania powinny być transakcyjne.

# Stosowane standardy

System JLibrary tworzony jest w języku Java 17 z użyciem struktury Maven i wzorca MVC. Dostęp do bazy danych realizowany jest przez JDBC z użyciem zapytań SQL. Komunikacja z zewnętrznym API odbywa się przez HTTPS w formacie JSON.

# Opis więzów

Dostęp do funkcji systemu możliwy tylko po zalogowaniu się. Czytelnicy nie mogą dodawać ani usuwać książek, ani zarządzać użytkownikami. System działa w modelu klient–serwer i wymaga połączenia sieciowego z serwerem. Połączenie z zewnętrznym API do pobierania danych książek wymaga dostępu do Internetu.