Ocena bezpieczeństwa systemów informacyjnych

Projekt – System rezerwacji sal

Autorzy:

*Weronika Jonczek*

*Grzegorz Pazdro*

*Szymon Jarząb*

*Karol Stachowicz*

*Jakub Janik*

# Metryka dokumentu

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Autor** | **Wersja** | **Wprowadzone zmiany** |
| 26.10.20 | Jakub Janik, Weronika Jonczek | 1.0.0 | Utworzenie dokumentu |
| 27.10.20 | Jakub Janik,  Weronika Jonczek | 1.0.1 | Wymagania funkcjonalne oraz niefunkcjonalne |
| 1.11.20 | Jakub Janik,  Weronika Jonczek | 1.0.2 | Aktorzy oraz model przepływu danych |
| 3.11.20 | Jakub Janik,  Weronika Jonczek | 1.0.3 | Diagram klas oraz schematy blokowe |
| 17.11.20 | Jakub Janik,  Weronika Jonczek | 1.0.4 | Przegląd zawartości, perspektywa projektu, funkcje projektu, założenia i zależności, ryzyko projektowe |
| 24.11.2020 | Jakub Janik,  Weronika Jonczek | 1.0.5 | Projekt architektoniczny, zabezpieczenia |
| 1.12.2020 | Jakub Janik,  Weronika Jonczek | 1.06 | Uzupełniony harmonogram prac oraz kamienie milowe |
| 8.12.20 | Jakub Janik,  Weronika Jonczek | 1.07 | Realizacja projektu, dokumentacja kodu źródłowego |

# Przedmowa

Celem niniejszego dokumentu jest przedstawienie oraz opisanie wymagań funkcjonalnych oraz niefunkcjonalnych potrzebnych do utworzenia systemu rezerwacji sal. Dokument zawiera informacje, które będą potrzebne w późniejszej pracy nad projektem oraz do jego finalnego ukończenia.

# Spis treści

[Metryka dokumentu 2](#_Toc58365379)

[Przedmowa 3](#_Toc58365380)

[Spis treści 3](#_Toc58365381)

[1. Wprowadzenie 4](#_Toc58365382)

[1.1. Cel dokumentu 4](#_Toc58365383)

[1.2. Zakres 4](#_Toc58365384)

[1.3. Przegląd zawartości 4](#_Toc58365385)

[2. Ogólny opis systemu 4](#_Toc58365386)

[2.1. Perspektywa produktu – ogólny opis projektu, podstawowe cechy i funkcje 4](#_Toc58365387)

[2.2. Funkcje projektu 5](#_Toc58365388)

[2.3. Charakterystyka użytkowników 5](#_Toc58365389)

[2.4. Aktorzy 6](#_Toc58365390)

[2.5. Model przepływu danych 9](#_Toc58365391)

[2.6. Diagram klas 9](#_Toc58365392)

[2.7. Schemat blokowy 10](#_Toc58365393)

[2.8. Założenia i zależności 11](#_Toc58365394)

[2.9. ryzyko projektu 11](#_Toc58365395)

[3. Specyfikacja wymagań 13](#_Toc58365396)

[3.1. Wymagania funkcjonalne 13](#_Toc58365397)

[3.2. Wymagania niefunkcjonalne 14](#_Toc58365398)

[4. Projekt architektoniczny aplikacji. 15](#_Toc58365399)

[5. Realizacja projektu 18](#_Toc58365400)

[5.1 Dokumentacja kodu źródłowego 18](#_Toc58365401)

[5.2. Screeny z realizacją aplikacji 19](#_Toc58365402)

# Wprowadzenie

## Cel dokumentu

Celem dokumentu jest opis systemu do rezerwacji sal na uczelni. Ukazanie podstawowych funkcji oraz specyfikacji wymagań. Ma przedstawić oraz opisać jak będzie funkcjonował system rezerwacji sal.

## Zakres

Dokument zawiera wymagania funkcjonalne i niefunkcjonalne systemu do rezerwacji sal. Projekt polega na stworzeniu dobrze działającego systemu na podstawie ustalonych wcześniej założeń.

## Przegląd zawartości

1. Metryka dokumentu – zakres modyfikacji wprowadzanych do dokumentu, aktorzy, data oraz wersja
2. Przedmowa – opis koncepcji dokumentu
3. Spis treści – spis treści dokumentu
4. Wprowadzenie – ogólne informacje na temat dokumentu, co jest w min zawarte
5. Ogólny opis dokumentu – perspektywa produktu, czyli ogólny opis projektu, jego podstawowe cechy i funkcje. Charakterystyka użytkowników, aktorzy, model przepływu danych, diagram klas, schemat blokowy, założenia i zależności, ryzyko projektu
6. Specyfikacja wymagań – wymagania funkcjonalne oraz niefunkcjonalne, opisana analiza wymagań

# Ogólny opis systemu

## Perspektywa produktu – ogólny opis projektu, podstawowe cechy i funkcje

Perspektywą produktu jest stworzenie systemu rezerwacji sal w formie strony internetowej. System ten umożliwiać będzie sprawne rezerwowanie pomieszczeń przez pracowników uczelni, jak i przez studentów. Będzie umożliwiał szybkie informowanie o ewentualnych problemach w rezerwacji ( wcześniejsze zajęcie sali ) oraz w takim przypadku łatwą komunikację pomiędzy osobami chętnymi na prowadzenie zajęć w danym pomieszczeniu.

## Funkcje projektu

Funkcje projektu to rezerwowanie sali dla użytkownika, informowanie o ewentualnej kolizji rezerwacji, komunikowanie użytkowników, którzy mają chęci rezerwacji tej samej sali w tej samej godzinie, za pomocą komunikatora.

## Charakterystyka użytkowników

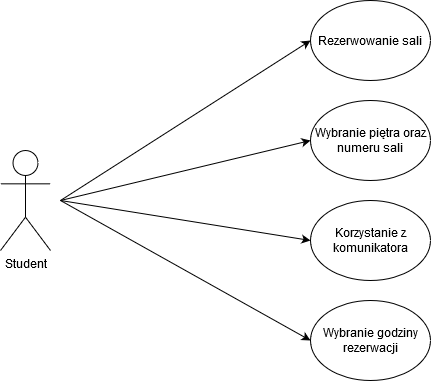
**Administrator** – osoba, która zajmuje się zarządzaniem stroną.

**Student** – osoba, która ma możliwość rezerwacji sali.

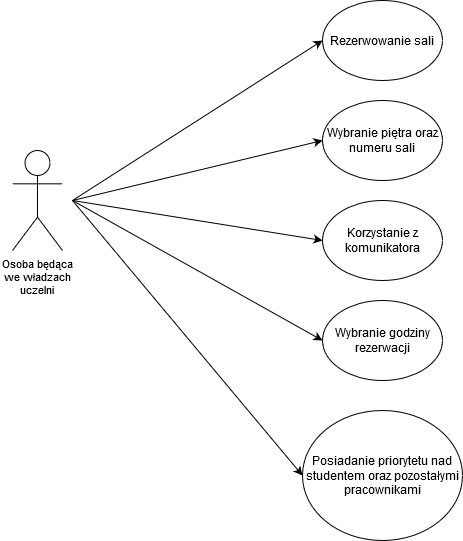
**Pracownik uczelni** – osoba, która ma możliwość rezerwacji sali oraz ma priorytet nad studentem.

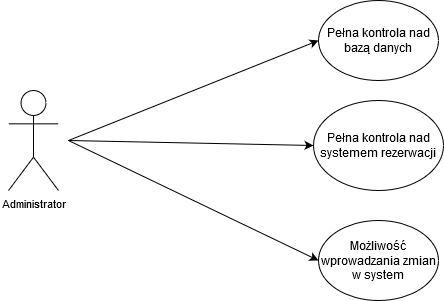
**Osoba będąca we władzach uczelni** – osoba, która ma możliwość rezerwacji sali oraz ma priorytet nad wszystkimi pozostałymi użytkownikami.

## Aktorzy







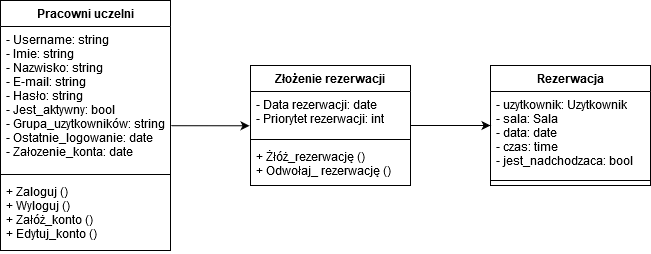


## Model przepływu danych

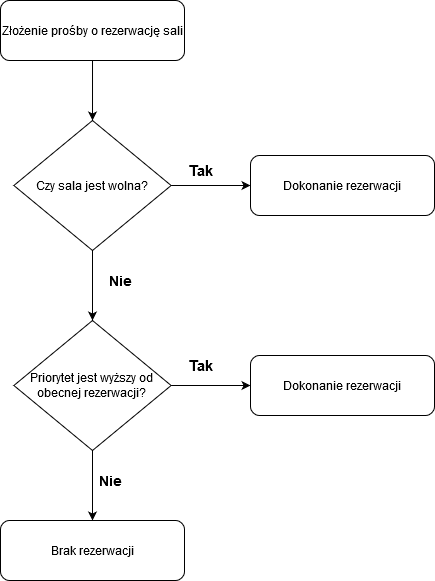
Obraz zawierający tekst

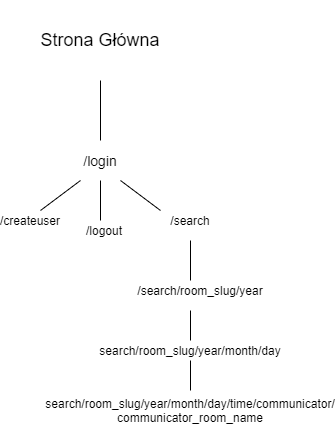
Opis wygenerowany automatycznie

## Diagram klas



## Schemat blokowy





## Założenia i zależności

System spełnia swoje funkcje przy dostępie do internetu oraz obsługuje bazę rezerwacji.

## ryzyko projektu

* Błędy harmonogramowania
* Problemy ze specyfikacją
* Rozszerzanie zakresu: dodatkowe wymagania pojawiające się̨ w trakcie projektu
* Różnice pomiędzy zakładaną, a rzeczywista wydajnością̨
* Braki sprzętowe
* Awaria serwerów

**Zarządzanie ryzykiem**

Zarządzanie ryzykiem stanowi centralny element zarządzania strategicznego każdej organizacji. Jest to proces, w ramach którego organizacja w sposób metodyczny rozwiązuje problemy związane z ryzykiem, które towarzyszy jej działalności, w taki sposób, aby ta działalność – zarówno w poszczególnych dziedzinach jak i traktowana jako całość – przynosiła trwałe korzyści.

|  |  |  |  |
| --- | --- | --- | --- |
| **Czynniki ryzyka** | **Kategoria** | **Prawdopodobieństwo zaistnienia** | **Działania** |
| Brak zaangażowania zespołu | Zespół projektowy | Małe ryzyko | Zwrócenie większej uwagi na członków zespołu, zaspokojenie ich indywidualnych potrzeb, przekazywanie im wszystkich szczegółów aby mogli poczuć się ważną częścią projektu. |
| Brak komunikacji | Zespół projektowy | Umiarkowane ryzyko | Regularne spotkania wszystkich członków zespołu zarówno w celu wykonywania zadań, jak i dzielenia się wiedzą powstałą w ramach projektu. |
| Słaba dokumentacja | Zespół projektowy | Umiarkowane ryzyko | Określenie jakie informacje powinny być zawsze dostępne. Wykorzystanie wyspecjalizowanych narzędzi pomagających w pisaniu dokumentacji.  Konsekwentne używanie określonej konwencji nazywania i opisywania zadań. |
| Braki znajomości technologii | Zespół projektowy | Duże ryzyko | Dodatkowe szkolenia, położenie nacisku na wykwalifikowanie zespołu, ciągłe dokształcanie, |
| Zaangażowanie użytkowników | Użytkownik | Umiarkowane ryzyko | Promowanie aplikacji, zachęcanie do użytku, przeszkolenie i zaznajomienie z aplikacją użytkowników |
| Nieskuteczna współpraca użytkowników i twórców SI | Projekt zespołowy, Użytkownik | Duże ryzyko | Przeprowadzenie wywiadu z użytkownikami, wprowadzanie zmian do projektu, |
| Zakłócenia we współpracy urządzeń i oprogramowania | Rozwój środowiska | Umiarkowane ryzyko | Wybór odpowiedniego sprzętu i oprogramowania |
| Lekceważenie poufności, prywatności i bezpieczeństwa SI | Rozwój środowiska | Duże ryzyko | Dogłębna analiza potrzeb i ograniczeń |
| pracochłonność aktualizowania danych | Projekt zespołowy, użytkownik | Duże ryzyko | Dbałość o ciągłą komunikację z organami administracyjnymi uczelni, bieżące aktualizowanie zasobów sal/ zmian w planie zajęć |
| niepełne przygotowanie do wdrażania i eksploatacji | Projekt zespołowy | Umiarkowane ryzyko | Ustalenie i trzymanie się harmonogramu i preliminarza prac projektowo - wdrożeniowych |

# Specyfikacja wymagań

## Wymagania funkcjonalne

**SYSTEM:**

Jako **administrator** – chcę mieć możliwość dowolnego ingerowania w bazę danych, chcę mieć nad nią pełną kontrolę. Chcę mieć dostęp do większości funkcji systemowych z panelu głównego.

Jako użytkownik **zalogowany** – chcę mieć możliwość zarządzania swoim kontem. Chcę mieć możliwą zmianę hasła oraz adresu e-mail.

Jako użytkownik **niezalogowany** – chcę mieć możliwość do zalogowania się

**EKRAN LOGOWANIA:**

Jako użytkownik **zarejestrowany** – chcę mieć możliwość do zalogowania się do systemu

Jako użytkownik **niezarejestrowany** – chcę mieć możliwość do rejestracji do systemu.

**SYSTEM REZERWACJI SAL:**

Jako **student** – chcę:

* Mieć możliwość zarezerwowania sali
* Mieć możliwość wybrania piętra oraz numeru sali
* Mieć możliwość dołączenia do komunikatora danej rezerwacji, w celu posiadania aktualnych informacji na temat ewentualnych zmian w rezerwacjach

Jako **pracownik uczelni** – chcę:

* Mieć możliwość zarezerwowania sali
* Mieć możliwość wybrania piętra oraz numeru sali
* Mieć możliwość dołączenia do komunikatora danej rezerwacji, w celu posiadania aktualnych informacji na temat ewentualnych zmian oraz w celu informowania studentów
* Mieć priorytet w rezerwacji sali nad studentami

Jako **osoba będąca we władzach uczelni** – chcę:

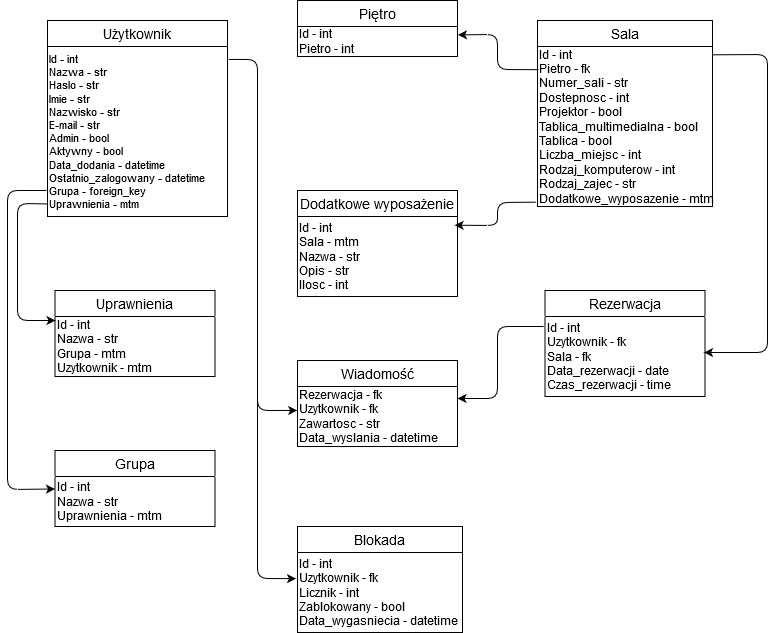
* Mieć możliwość zarezerwowania sali
* Mieć możliwość wybrania piętra oraz numeru sali
* Mieć możliwość dołączenia do komunikatora danej rezerwacji, w celu posiadania aktualnych informacji na temat ewentualnych zmian oraz w celu informowania studentów
* Mieć priorytet w rezerwacji sali nad studentami oraz nad wykładowcami

## Wymagania niefunkcjonalne

* Wymagania wydajnościowe
* Wymagania wsparcia
* Wymagania użyteczności
* Wymagania bezpieczeństwa
* Obsługa bazy danych
* Język polski

## Projekt architektoniczny aplikacji.

* 1. Architektura bazy danych



* 1. Zabezpieczenia

**Przechowywanie hasła**

* <algorytm>$<iteracje>$<sól>$<hash>
* Algorytm PBKDF2
* Hash SHA256
* 216000 iteracji

Możemy przechowywać w bazie danych hasła zaszyfrowane różnymi algorytmami, ale wtedy pojawia się podatność na user enumeration timing attack. Bronimy się przed nim „owijając” zapisane hasła w nowy algorytm.

**Logowanie**

* Uwierzytelnianie
* Login + hasło
* Dozwolone 5 nieprawidłowych prób uwierzytelnienia pod rząd, później blokada na godzinę

**Rejestracja**

Rejestrować nowych użytkowników mogą władze uczelni.

* Nazwa użytkownika
* Imię
* Nazwisko
* E-mail
* Hasło
* Rodzaj Użytkownika ( student/wykładowca)

**Walidacja**

* Rejestracja
* Wszystkie pola <50 znaków
* Walidacja adresu e-mail
* Sprawdzenie czy username występuje w bazie
* Sprawdzenie czy e-mail występuje w bazie

Walidacja hasła:

* 8 – 50 znaków
* Nie może być podobne do innych atrybutów
* Musi zawierać duże i małe litery, cyfry i znaki specjalne
* Nie może być pospolite (sprawdzamy czy występuje w bazie 20000 pospolitych haseł)
* Hasło i potwierdzenie hasła muszą być identyczne

**Uwierzytelnianie komunikatora**

* WebSocket kontaktuje się z serwerem HTTP, żeby dostać ticket zawierający dane użytkownika
* Przechowuje go w pamięci cache
* Użytkownik otwiera komunikator (websockety)
* Przesyłany jest ticket jako handshake
* Serwer sprawdza ticket (IP, czy jest ważny, czy nie jest zużyty)
* łączy użytkownika

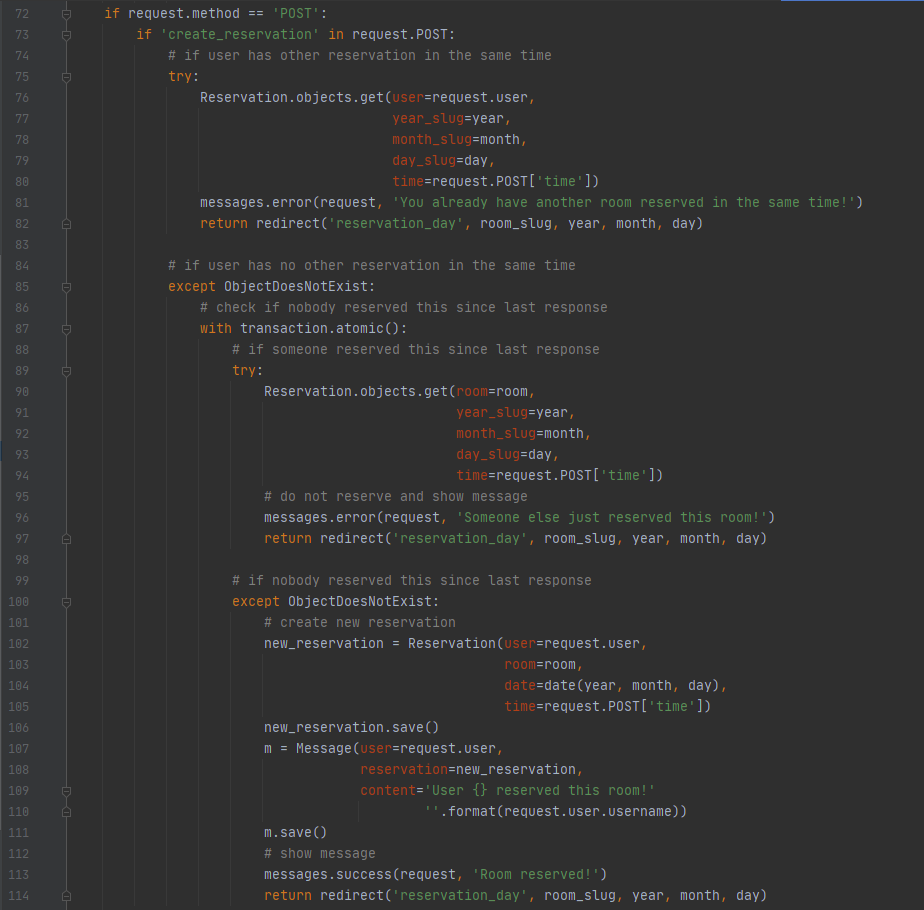
**Zabezpieczenia**

|  |  |
| --- | --- |
| **Rodzaj ataku** | **Zabezpieczenia jakich używamy** |
| Cross site request forgery | Token dołączamy do każdego formularza |
| User enumeration timing attack | Przy zmianie algorytmu szyfrowania wszystkie hasła w bazie danych są szyfrowane nowym algorytmem |
| Przepełnienie bazy danych | Dodawanie użytkowników przez zaufane osoby, dodatkowo ograniczenie długości wprowadzanych danych |
| SQL injection | Query parametrization |
| Cross site scripting | HTML escaping |
| Clickjacking | Header dołączany do wszystkich odpowiedzi serwera |
| Sniffing | Zapytania HTTP przekierowane automatycznie do HTTPS, szyfrowanie ciasteczek |
| Fake host | Lista dozwolonych hostów |
| Brute force attack | Limit nieudanych prób logowania |
| Dodawanie niepożądanych rezerwacji | Limit 1 rezerwacji w tym samym czasie, wywłaszczanie rezerwacji |
| DDos | Możliwość blokowania IP, ograniczenie metod http, ograniczenie liczby możliwych zapytań dla hosta, możliwość wyłączenia zapisów logów |

# **Realizacja projektu**

### 5.1 Dokumentacja kodu źródłowego

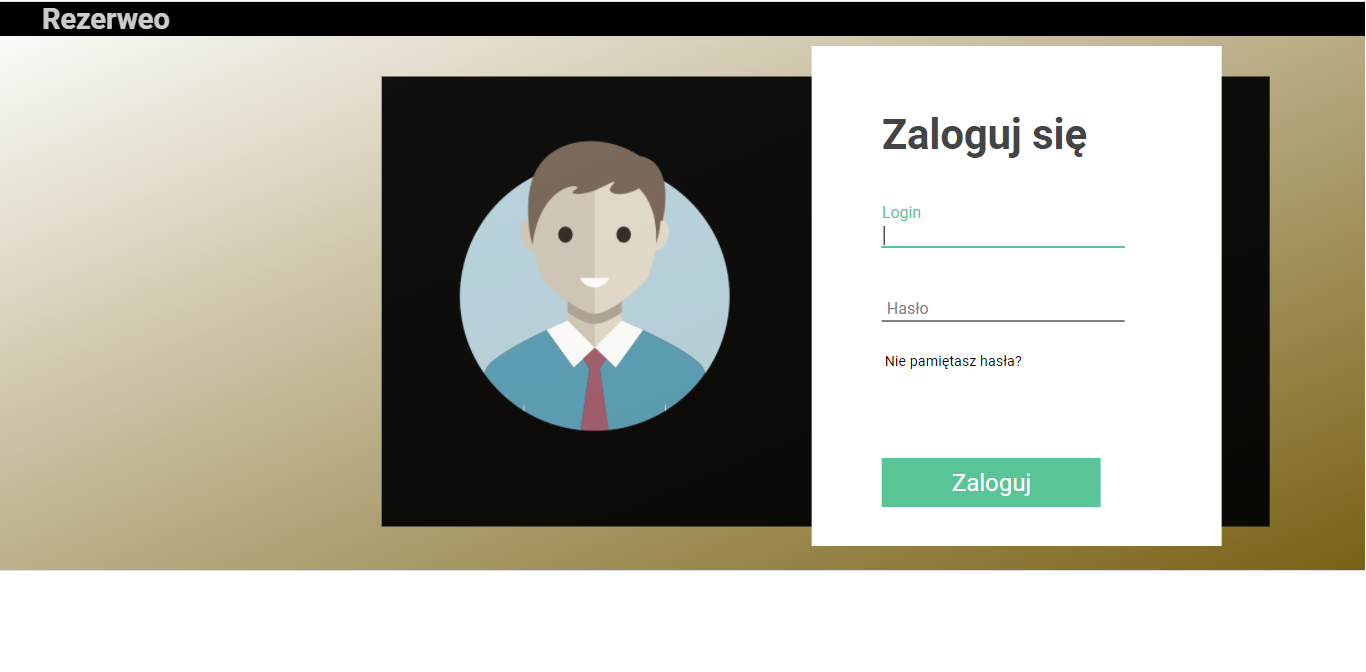
Rezerwacja sali



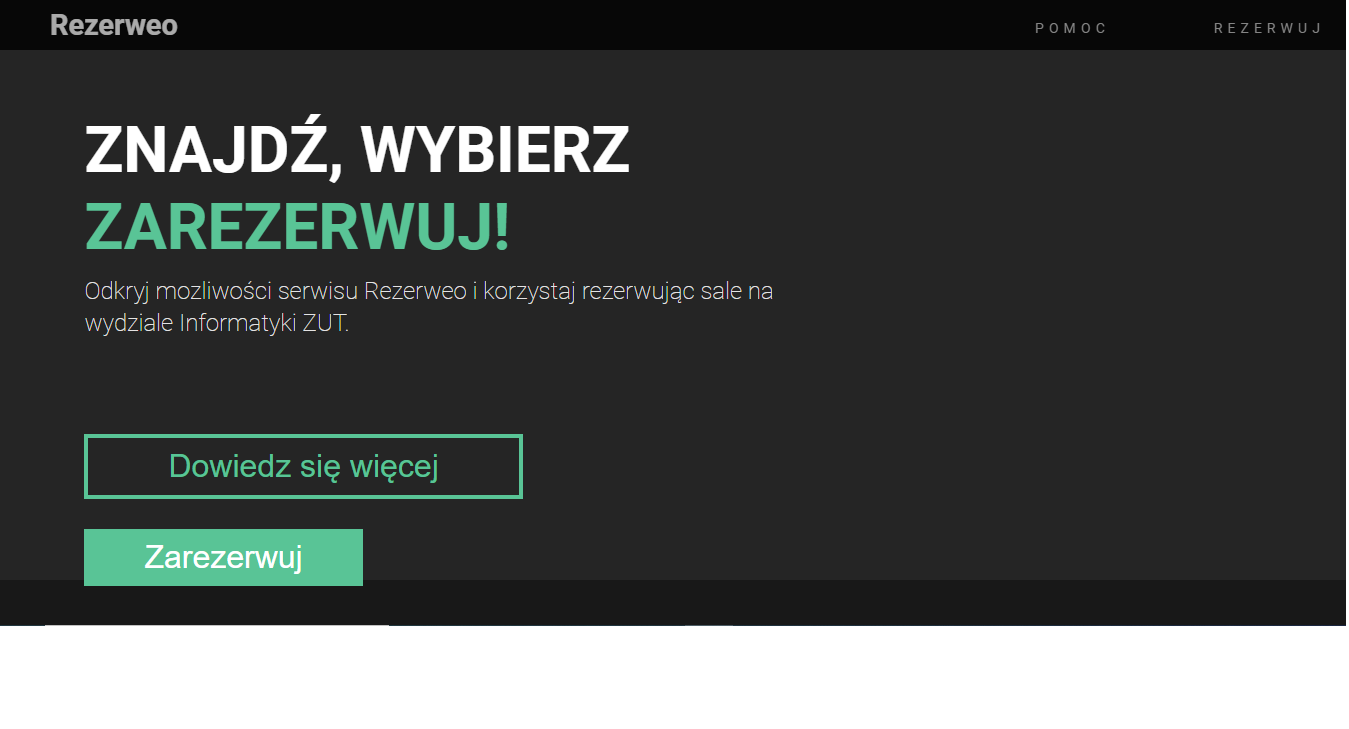
Aplikacja została napisana w języku Python Django. Przy wyborze technologii, kierowaliśmy się przeznaczeniem naszej aplikacji oraz do kogo jest ona kierowana. Jako młody i niedoświadczony w tworzeniu projektów zespół, zależało nam na tym aby język był wszechstronny, szybki, prosty w implementacji. Python posiada wiele dostępnych bibliotek w razie gdybyśmy chcieli rozszerzyć naszą aplikację. Jedną z głównych zalet wyboru tej technologii jest automatycznie tworzony kompletny panel admina jak i sprawdzony podział kodu pomiędzy modele, formularze, widoki i szablony.

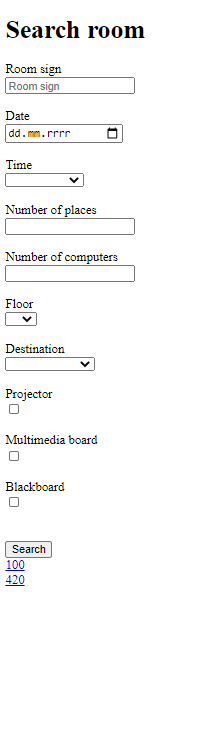
Link do plików z kodem źródłowym: <https://github.com/js40598/OBSI.git>

### 5.2. Screeny z realizacją aplikacji



Każdy użytkownik, aby móc dokonać rezerwacji musi przejść przez proces logowania. W przypadku zapomnienia hasła, istnieje możliwość odzyskania go za pomocą maila przez wygenerowanie tymczasowego kodu umożliwiającego przywrócenie hasła.





Wyszukiwarka sal polega na podaniu daty oraz godziny na kiedy chcemy zarezerwować salę, liczbę potrzebnych na sali miejsc, piętro, przeznaczenie sali, a także mamy możliwość zaznaczenia dodatkowych opcji takich jak czy na sali ma znajdować się projektor bądź tablica. Użytkownik student musi liczyć się z tym, że jego rezerwacja może zostać anulowana jeśli pracownik uczelni będzie potrzebował zarezerwować daną salę na ten sam termin. Wówczas student zostanie o tym fakcie poinformowany przez komunikator.