

Spis treści

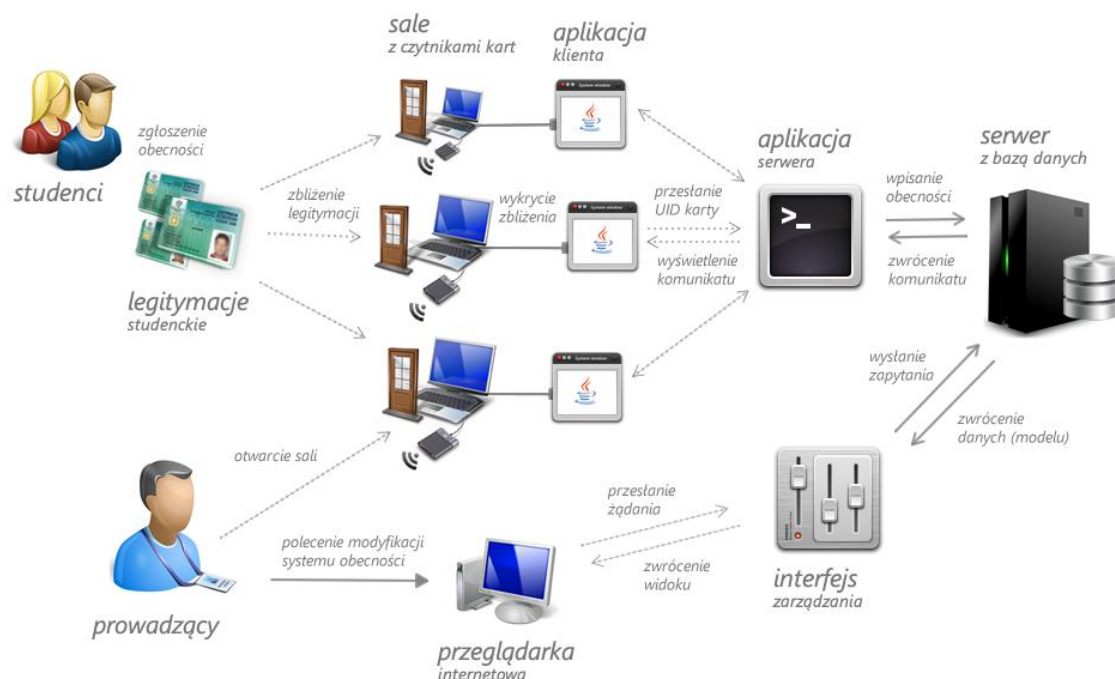
Spis treści	2
1 Wstęp.....	5
2 Zastosowane technologie	6
2.1 MIFARE	6
2.2 APDU	7
2.3 Wi-Fi	8
2.4 Java.....	8
2.5 Swing.....	9
2.6 HTML.....	9
2.7 CSS	10
2.8 JavaScript i AJAX	11
2.9 PHP.....	11
2.10 MySQL	12
3 Architektura systemu	14
3.1 Serwer	14
3.2 Klient	15
3.2.1 Okno główne aplikacji klienta	15
3.2.1.1 Wskaźniki stanu sprawdzania obecności	16
3.2.1.2 Przyciski uruchamiania i zatrzymania sprawdzania obecności	16
3.2.1.3 Panel odczytanego numeru UID i odpowiedzi serwera	16
3.2.1.4 Zakładki nawigacji pomiędzy oknami.....	17
3.2.1.5 Panel historii zdarzeń.....	17
3.2.1.6 Panel ustawień połączenia.....	18
3.2.1.7 Panel ustawień czytnika.....	18
3.2.1.8 Przycisk otwierający panel zarządzania obecnościami.....	19
3.2.2 Działanie w tle.....	19
3.3 Panel zarządzania	21
3.3.1 Autoryzacja	22
3.3.2 Okno główne panelu zarządzania	22
3.3.3 Studenci i prowadzący.....	25
3.3.4 Sale.....	29

3.3.5	Kursy	29
3.3.6	Grupy zajęciowe	30
3.3.7	Terminy	32
3.3.8	Listy obecności	34
3.3.9	Zapisy studentów	35
4	Scenariusz użycia	37
4.1	Utworzenie konta studenta i prowadzącego	37
4.2	Utworzenie sali	39
4.3	Utworzenie kursu i grupy zajęciowej	39
4.4	Otwarcie sali i sprawdzenie obecności	42
5	Instalacja i uruchomienie oprogramowania	45
5.1	Instalacja i konfiguracja serwera	45
5.1.1	Uruchomienie komponentu MySQL	45
5.1.2	Uruchomienie wirtualnego routera	45
5.1.3	Uruchomienie aplikacji serwera	46
5.1.4	Program uruchamiający serwer	47
5.2	Instalacja i konfiguracja klienta	47
5.2.1	Uruchomienie aplikacji	47
5.2.2	Podłączenie czytnika kart	48
5.2.3	Połączenie z serwerem	48
5.2.4	Konfiguracja klienta	48
5.2.5	Konfiguracja połączenia	49
5.2.5.1	Konfiguracja czytnika	49
5.2.5.2	Uruchomienie czytania kart	50
5.3	Instalacja i konfiguracja panelu zarządzania	51
5.3.1	Instalacja serwera Apache, PHP i MySQL	51
5.3.2	Instalacja i konfiguracja panelu zarządzania	52
6	Cytowane prace	54
7	Spis ilustracji	56
8	Dodatek - Kod źródłowy rozwiązania	59
8.1	Serwer	59
8.1.1	Opis architektury programu	59
8.1.2	Diagram UML klas programu serwera	60
8.1.3	Schemat bazy danych MySQL	61

8.2	Klient	61
8.2.1	Opis architektury programu.....	61
8.2.2	Diagram UML klas programu klienta.....	62
8.3	Panel zarządzania	62
8.3.1	Model.....	63
8.3.2	Widok	68
8.3.3	Kontroler	69

1 Wstęp

Celem pracy inżynierskiej było zaprojektowanie i stworzenie systemu kontroli obecności na uczelni wyższej wykorzystującego karty MIFARE. Rys. 1-1 przedstawia schemat blokowy systemu kontroli obecności, który został wykonany i w tej pracy szczegółowo omówiony.



Rys. 1-1 Schemat systemu kontroli obecności

System kontroli obecności składa się z trzech głównych i nieodłącznych elementów – serwera, klienta i interfejsu zarządzania. W pracy omawiane są zawsze w tej kolejności tak, aby wyczerpać zagadnienia instalacji, konfiguracji, uruchomienia i użytkowania oprogramowania, ale jednocześnie dogłębnie poznać jego architekturę i kod.

2 Zastosowane technologie

System kontroli obecności wykorzystuje, poza wymienioną w tytule technologią MIFARE, wiele innych, umożliwiających poprawną komunikację pomiędzy wszystkimi elementami systemu. W tym rozdziale zawarty jest opis wszystkich technologii, które współtworzą system kontroli obecności – MIFARE, Wi-Fi, Java, Swing, HTML, CSS, JavaScript, AJAX, jQuery, PHP i MySQL.

2.1 MIFARE

MIFARE to bezdotykowy standard kart opracowany przez firmę NXP Conductors. Technologia ta jest często stosowana przy tworzeniu rozwiązań programistycznych wykorzystujących mechanizm kart bezstykowych, o czym świadczy liczba 50 milionów sprzedanych czytników i ponad 5 miliardów kart MIFARE. [1] [2] Główną własnością kart MIFARE, określanych też mianem ang. *smart card*, jest zawarty w jej wnętrzu mikroprocesor z anteną, która w wyniku działania indukcji elektromagnetycznej (pojawiającej się przy zbliżaniu karty do czytnika), zasila mikroprocesor i umożliwia bezdotykową komunikację karty z czytnikiem. Umożliwia to zapis danych na karcie, przechowywanie kluczy zabezpieczeń oraz różne operacje arytmetyczne, np. kontrolę poprawności danych.

Wszystkie karty MIFARE stosują transmisję danych bezdotykową przy optymalnym zasięgu czytnika do 10 cm, na częstotliwości 13,56MHz, z maksymalną szybkością transferu danych 100Kb/s. Karta posiada mechanizmy kontroli poprawności danych – zliczanie bitu parzystości, CRC (ang. *cyclic redundancy check*). Może realizować także podstawowe mechanizmy zabezpieczeń i autoryzacji poprzez przechowywanie kodu PIN, kodowanie danych w czasie transmisji i trzypoziomowy system uwierzytelniania. Nowe generacje kart – MIFARE Pro X – umożliwiają autoryzację kluczem PKI. [3]

MIFARE Classic 1K to najczęściej używany typ kart bezstykowych, stosowany m. in. do kontroli dostępu i identyfikacji. Legitymacja studencka na Politechnice Wrocławskiej także jest kartą *smart card* tego typu. Główne cechy kart

MIFARE Classic 1K to 1KB pamięci EEPROM (ale jedynie 768 bajtów do zapisu), unikalny numer seryjny karty UID (długość 4 bajty lub 7 bajty), dopuszczalna liczba 100 tys. operacji zapisu karty i 10 lat ważności przechowywanych danych. Do zabezpieczenia tej karty może być użyty klucz 48-bitowy [4].

2.2 APDU

Komunikacja czytnika z kartą możliwa jest dzięki protokołowi APDU (ang. *application protocol data unit*), który zdefiniowany jest przez normę ISO/IEC 7816-4. Protokół ten wyróżnia dwie kategorie wiadomości – *command* APDU i *response* APDU, czyli żądanie i odpowiedź. Żądanie jest wysyłane do czytnika w formie 4-bajtowego nagłówka (zawierającego informację o typie żądania i określającego liczbę bajtów zawartości żądania) i do 255 bajtów zawartości żądania. Odpowiedź jest wysyłana przez kartę do czytnika; zawiera 2-bajtowy nagłówek (długość odpowiedzi i status odpowiedzi) i do 256 bajtów zawartości odpowiedzi. [5, p. 5.3]

Komenda odczytania numeru UID z karty za pomocą protokołu APDU jest następująca:

> FF CA 00 00

gdzie: FF to wartość nagłówka CLA odpowiadająca za identyfikator klasy instrukcji, CA to wartość nagłówka INS odpowiadającego za kod instrukcji, żądająca pobrania danych (GET DATA). Pozostałe dwa nagłówki – 00 00 – to wartości dodatkowych parametrów.

Poprawny komunikat zwrotny, czyli odpowiedź APDU powinna być następująca:

> 90 00 8B 0E F9 88

gdzie: 90 00 to 2 bajty nagłówka statusu odpowiedzi, które w tym przypadku oznaczają „sukces”, zaś pozostałe 4 bajty – 8B 0E F9 88 – to odczytany przykładowy numer UID z legitymacji studenta o numerze albumu 180394). [5, p. 6]

2.3 Wi-Fi

Technologia Wi-Fi umożliwia bezprzewodowe połączenie dowolnych urządzeń elektronicznych z Internetem i sieciami komputerowymi. Wi-Fi, znane także jako 802.11b, jest wiodącą technologią w dziedzinie sieci bezprzewodowych zarówno w przemyśle, jak i warunkach domowych. Pomimo tego, że zaprojektowane zostało do tworzenia zamkniętych, prywatnych sieci, stosowane jest także do udostępniania sieci bezprzewodowych w miejscach publicznych w postaci tzw. *hotspotów*, przez które dowolne urządzenie wyposażone w moduł Wi-Fi może uzyskać dostęp do Internetu [6].

Współczesne systemy operacyjne zawierają wiele użytecznych funkcji, które pozwalają na szersze wykorzystanie technologii Wi-Fi. Jedną z nich jest wirtualizacja bezprzewodowego interfejsu sieciowego (ang. *Virtual Wi-Fi*), która umożliwia wirtualne powielenie interfejsu Wi-Fi, a w konsekwencji też udostępnienie dostępu do sieci innym komputerom. W praktyce oznacza to, że dowolny komputer, posiadający dostęp do Internetu, może swoje połączenie internetowe udostępnić jako nową i niezależną sieć bezprzewodową, stając się tym samym tzw. wirtualnym routerem. [7]

2.4 Java

Java to wysokopoziomowy, zorientowany obiektowo język programowania, rozwijany przez firmy Sun Microsystems. Kod programów napisanych w języku Java jest kompilowany do kodu bajtowego i uruchamiany na wirtualnej maszynie Java (ang. *JRE Java Runtime Edition*). Zarówno środowisko programistyczne (ang. *JDK Java Development Kit*), bardzo rozbudowane, z wielką liczbą bezpłatnych rozszerzeń i modułów, jak i środowisko uruchomieniowe JRE są dostępne dla wielu współczesnych systemów operacyjnych – Linux, systemy Unixowe, Windows, Mac OS X i inne; skutkuje to ważną cechą programów napisanych w języku Java – wieloplatformowość, czyli identyczne działanie tego samego kodu w różnych środowiskach/systemach operacyjnych. [8] Java posiada własny mechanizm zarządzania pamięcią – *garbage collector* i mechanizm obsługi wyjątków. Istnieje wiele rozbudowanych środowisk programistycznych dla języka Java, np. Eclipse czy Netbeans, które znacznie usprawniają tworzenie i rozwijanie projektów, zapewniają spójność nazewnictwa pomiędzy plikami projektów, posiadają narzędzia debugowania, eksportowania projektu czy generowania dokumentacji. Język Java umożliwia

tworzenie różnego rodzaju projektów – *appletów* (programów uruchamianych w przeglądarkach internetowych), *servletów* (programy serwerowe), aplikacji webowych, czy aplikacji mobilnych działających np. w systemie Android. [9]

Java Class Library/Java Foundation Classes (JFC) to zespół bibliotek poszerzających możliwości aplikacji w języku Java o np. przetwarzanie plików XML, szyfrowanie, zaawansowane struktury danych, łączność z bazami danych, a przede wszystkim interfejs użytkownika GUI (ang. *graphical user interface* – interfejs graficzny użytkownika). [10]

2.5 Swing

Częścią rodziny Java Foundation Classes (JFC) jest biblioteka Swing, popularny moduł do tworzenia interaktywnych aplikacji z graficznym interfejsem użytkownika (GUI). Obecnie Swing (w wersji 1.4) zawiera 85 publicznych interfejsów i 451 publicznych klas.

Swing bazuje na komponencie AWT (Abstract Window Toolkit) – podstawowym zestawie narzędzi do tworzenia GUI w Java, ale poszerza je o nowe komponenty (tabele, drzewa, paski postępu, ramki, komponenty tekstowe, suwaki), umożliwia tworzenie „dymków” z podpowiedziami, dodaje mechanizm obsługi zdarzeń naciśnięcia klawisza nad komponentem. Poza tym Swing umożliwia tworzenie własnych kontroltek (funkcjonalnych elementów graficznego interfejsu) i zmianę stylu wyświetlania całego interfejsu (ang. *Look and Feel* – L&F). [11]

2.6 HTML

HTML (ang. *Hypertext Markup Language*) to powszechnie używany, uniwersalny interpretowany język znaczników opisujących strukturę dokumentu w sieci WWW. Umożliwia publikowanie na stronach internetowych formatowanych tekstów, tabel, obrazów, list i formularzy. [12]

HTML 5 to nowa generacja języka HTML, wyrastająca i czerpiąca z każdego ze standardów HTML 4.01, XHTML 1.0 i XHTML 1.1. HTML 5 udostępnia wiele przydatnych i niezbędnych narzędzi do tworzenia nowoczesnych aplikacji internetowych oraz zawiera definicje metod projektowania aplikacji internetowych, które chociaż

przyjmowane i używane przez programistów od lat, nigdy nie zostały udokumentowane i uznane oficjalnie za standard.

HTML 5, tak jak jego poprzednicy, nie wymaga do działania specyficznego systemu operacyjnego czy zainstalowanego środowiska programistycznego na maszynie. Do poprawnej interpretacji HTML 5 wystarczy dowolna nowoczesna przeglądarka internetowa – Mozilla Firefox, Opera, Google Chrome, Apple Safari, które są dostępne w różnych wersjach dla każdego systemu operacyjnego; przeglądarki internetowe zainstalowane domyślnie na urządzeniach mobilnych – iPhone i Android – także doskonale sobie radzą z obsługą HTML 5. [13]

Poza nowościami, które wprowadza HTML 5: semantyczne znaczniki – `<section>`, `<footer>`, `<header>`; element `<canvas>`, który umożliwia rysowanie obiektów 2D/3D oraz ich transformacje; geolokalizacja użytkownika; dostęp do mikrofonu i kamery internetowej; wsparcie dla obsługi mediów, standard ten także bardzo silnie współpracuje z wieloma innymi technologiami internetowymi. Mowa tu o CSS 3 i technice AJAX, czyli kolejno kaskadowych arkuszy styli i asynchronicznym JavaScript; technologie te odpowiadają kolejno za warstwę prezentacyjną aplikacji oraz warstwę interakcji asynchronicznej użytkownika z serwerem. [14]

HTML 5 zatem, chociaż posiada ścisłą specyfikację i sam w sobie skupia się wyłącznie na języku znaczników, które opisują strukturę i semantykę dokumentu, jest też standardem projektowania stron internetowych przy użyciu wyżej wymienionych technologii. Dlatego z tak rozumianym HTML5 wiąże się wiele innych pojęć: dostępność, użyteczność, skalowalność i interaktywność. [15]

2.7 CSS

CSS (ang. *Cascading Style Sheets*) to język opisujący warstwę prezentacyjną dokumentu HTML. Warstwa prezentacyjna i semantyczna (zawierająca „treść” dokumentu, ubrana w znaczniki HTML) powinny być bezwzględnie oddzielone. Dzięki tej separacji możliwe jest stosowanie jednej definicji wyglądu dla setki dokumentów, a także łatwą zmianę sposobu wyświetlania dla różnych platform, rodzajów urządzeń czy przeglądarek internetowych. [16]

Najnowszym standardem arkuszy styli jest CSS3, który poza szerszym zakresem dostępnych selektorów, udostępnia wiele nowości, takich jak efekty graficzne (cienie, poświaty, zaokrąglone narożniki, półprzezroczystość), transformacje (przesunięcia i obroty) oraz dowolne ich animowanie, a nawet umożliwia stosowanie *fontów* użytkownika. [15]

2.8 JavaScript i AJAX

JavaScript to język skryptowy, który wykonuje się po stronie klienta, w przeglądarce internetowej. AJAX (Asynchronous JavaScript and XML) natomiast to technika wykorzystująca język skryptowy w aplikacji internetowej do komunikacji asynchronicznej z modelem danych (np. serwerem, bazą danych, innym dokumentem HTML), mająca na celu stworzenie w pełni interaktywnego interfejsu komunikacji z użytkownikiem (niewymagającego przeładowania całej strony internetowej z nowym żądaniem, jak to ma miejsce w standardowym HTML-u). [14] Istnieje też wiele otwartych bibliotek, które udostępniają m.in. cały interfejs komunikacji asynchronicznej. Najpopularniejszą z nich jest biblioteka jQuery, za pomocą której można łatwo nawiązać połączenie asynchroniczne, a następnie przetworzyć odpowiedź i wprowadzić modyfikację w strukturze dokumentu HTML. Ta sama biblioteka także zawiera znaczne poszerzenie funkcjonalności standardowych formularzy dostępnych w HTML, dzięki czemu można korzystać z takich elementów jak wybór daty, dynamiczna walidacja pól formularzy czy autouzupełnianie. [17]

2.9 PHP

PHP (ang. *PHP Hypertext Preprocessor*) to otwarty język skryptowy wykonywany się po stronie serwera, stosowany do dynamicznego generowania zawartości stron internetowych (np. HTML). PHP skupia się głównie na obsłudze skryptów po stronie serwera – przetwarzanie danych z formularzy tekstowych, generowanie dynamicznej zawartości strony, mechanizm sesji, czytanie i zapisywanie plików ciasteczek (znane pod ang. nazwą *cookies*). Do poprawnego działania tych skryptów wymagany jest serwer z obsługą PHP, np. Apache lub IIS. Co więcej, PHP pozwala również na pisanie skryptów działających w konsoli i tworzenie rozbudowanych interfejsów graficznych do aplikacji

desktopowych (po zainstalowaniu odpowiednich rozszerzeń). Pomimo tego, że jest to interpretowany język skryptowy, wspiera programowanie obiektowe i istnieje wiele platform programistycznych (ang. *framework*) bazujących na języku PHP – nawet oficjalny, tworzony przez twórców języka PHP – Zend Framework; a wraz z nimi mnogość systemów zarządzania treścią (ang. Complete Management System – CMS) do obsługi systemów zakupów internetowych (ang. e-commerce), publikowania treści, portali społecznościowych i wielu innych. [18]

O popularności języka PHP na pewno może świadczyć liczba 20 milionów aplikacji wykorzystujących ten język i 1 milion serwerów z zainstalowanym interpreterem PHP. [19] Język ten może być używany z większością systemów operacyjnych – Linux, systemy Unixowe (m.in. Solaris i OpenBSD), Microsoft Windows, Mac OS X i inne, współpracuje z większością typów serwerów – Apache i IIS oraz lighttpd i nginx. Elastyczność języka PHP zwiększa to, że użytkownik może wybrać odpowiadający mu styl programowania: proceduralny lub obiektowy, oraz jaki typ danych skrypt PHP ma generować; możliwe jest dynamiczne tworzenie nie tylko plików HTML, ale i plików XML, JSON, dokumentów PDF, obrazów, a nawet interaktywnych zawartości Flash (przy użyciu komponentów libswf lub Ming). [20]

Zaletą PHP jest mnogość rozszerzeń i modułów oraz dobra dokumentacja i duża społeczność programistów je tworzących. PEAR [21] to oficjalne repozytorium rozszerzeń, dzięki któremu możemy zainstalować biblioteki obsługujące np. generowanie dokumentów Microsoft Office Excel, czy uwierzytelnianie za pomocą CAPTCHA. PECL natomiast to baza otwartych modułów do samodzielnej kompilacji, którymi także można poszerzyć funkcjonalność PHP o np. obsługę mediów, techniki AJAX i wielu innych. Wszystkie te rozszerzenia, jak i sam język PHP – co warto zaznaczyć – mają otwarte źródła i są dostępne na licencji GPL (GNU General Public License). [22]

2.10 MySQL

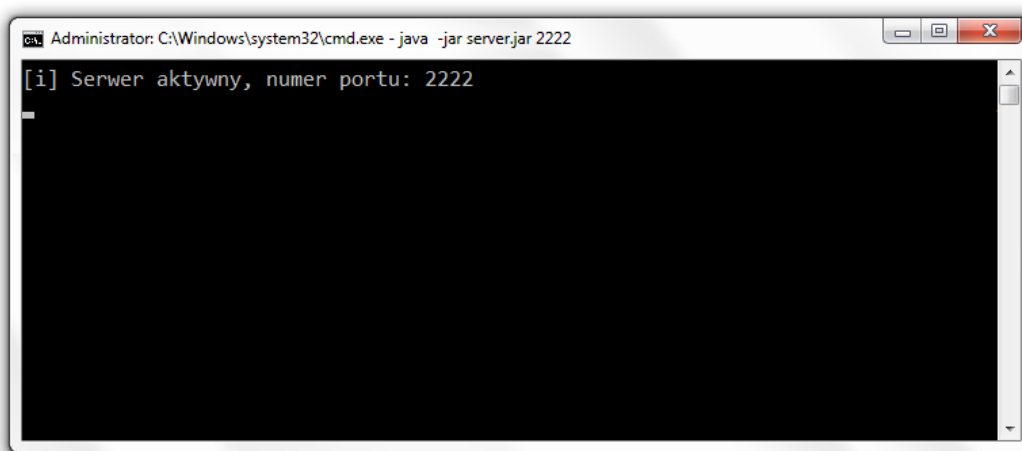
MySQL to system zarządzania relacyjnymi bazami danych wykorzystujący standardowy system zapytań SQL; wydawany na licencji GPL (ang. GNU General Public License), zaprojektowany do obsługi dużych baz danych, przy użyciu wielowątkowego serwera SQL, który byłby w stanie obsługiwać zapytania z różnych środowisk, programów i narzędzi.

MySQL jest napisany w C i C++, może pracować w wielu systemach operacyjnych – Linux, systemy Uniksowe, Windows oraz Mac OS X. Udostępnia również API (Application Programming Interface – interfejs komunikacji między narzędziami/systemami programistycznymi) do niemal każdego popularnego języka programowania – C, C++, .NET, Java, PHP, Python i inne. Ponadto posiada mechanizmy zarządzania pamięcią i jest w stanie wykorzystać moc wieloprocessorowych maszyn. Znany jest przypadek poprawnie działającego serwera MySQL, który korzystał z bazy o 60 tys. tabel i 5 miliardów rekordów. Dodatkowo MySQL wspiera większość popularnych systemów kodowań znaków, włączając w to poprawną obsługę polskich znaków diakrytycznych. [23]

3 Architektura systemu

3.1 Serwer

Wskazówki instalacji, konfiguracji i uruchomienia aplikacji serwera znajdują się w rozdziale 5.1 Instalacja i konfiguracja serwera. Opis zastosowanych rozwiązań programistycznych, wzorców projektowych, klas i funkcji znajduje się w rozdziale 8.1 Dodatek - Kod źródłowy – serwer.



Rys. 3-1 Okno główne aplikacji serwera

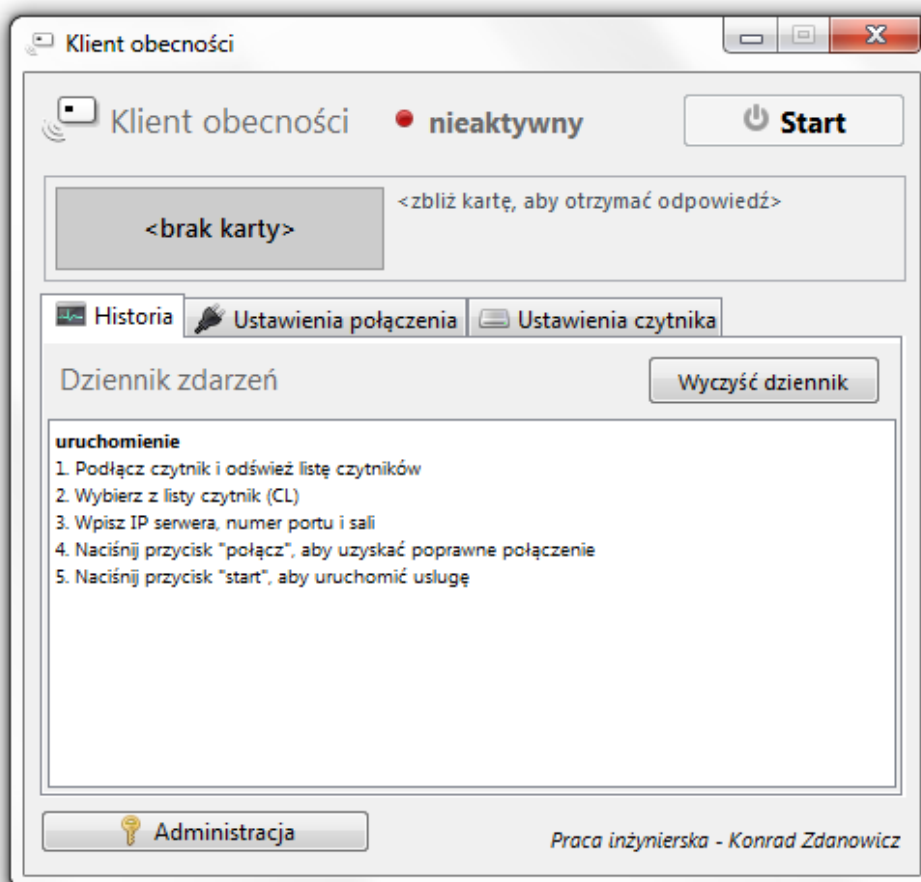
Rys. 3-1 przedstawia okno główne aplikacji serwera. Program, oprócz komunikatu powitalnego, nie wyświetla żadnych informacji. Jeżeli napotka błąd w komunikacji z bazą danych MySQL (w sytuacji gdy usługa MySQL nie została włączona lub nie działa poprawnie), wyświetlony zostanie komunikat [!] Brak połączenia z bazą danych MySQL. Jeśli port, którego numer podany jest w parametrze wywołania programu, jest już nasłuchiwany przez inną aplikację, wyświetlona zostanie wiadomość [!] Wybrany port jest już otwarty i nasłuchiwany przez inną aplikację.

Aby zatrzymać działanie serwera, należy wyłączyć okno za pomocą przycisku *Zamknij* lub kombinacją klawiszy CTRL+C.

3.2 Klient

Wskazówki instalacji, konfiguracji i uruchomienia aplikacji klienta znajdują się w rozdziale 5.2 Instalacja i konfiguracja klienta. Opis zastosowanych rozwiązań programistycznych, wzorców projektowych, klas i funkcji znajduje się w rozdziale 8.2 Dodatek - Kod źródłowy – klient.

3.2.1 Okno główne aplikacji klienta



Rys. 3-2 Okno główne aplikacji klienta

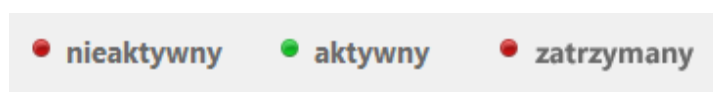
Rys. 3-2 przedstawia okno główne aplikacji klienta. Graficzny interfejs okna głównego aplikacji składa się z następujących elementów:

- wskaźniki stanu sprawdzania obecności (rys. 3-3),
- przyciski uruchamiania i zatrzymania sprawdzania obecności (rys. 3-4),
- panel odczytanego numeru uid i odpowiedzi serwera (rys. 3-5),
- zakładki nawigacji pomiędzy oknami (rys. 3-6),

- e) panel historii zdarzeń (rys. 3-7),
- f) panel ustawień połączenia (rys. 3-8),
- g) panel ustawień czytnika (rys. 3-9),
- h) przycisk otwierający panel zarządzania obecnościami (rys. 3-10).

3.2.1.1 Wskaźniki stanu sprawdzania obecności

Jeżeli sprawdzanie obecności nie jest zainicjalizowane, aplikacja wskazuje stan *nieaktywny*. Jeżeli naciśnięto przycisk *Start* i sprawdzanie obecności jest zainicjalizowane, wskazany jest stan *aktywny*. Jeżeli użytkownik zatrzyma sprawdzanie obecności przyciskiem *Stop*, wskazany jest stan *zatrzymany*.



Rys. 3-3 Wskaźniki stanu sprawdzania obecności

3.2.1.2 Przyciski uruchamiania i zatrzymania sprawdzania obecności

Przycisk *Start* jest domyślnie zablokowany. Dopiero poprawne nawiązanie połączenia z serwerem odblokowuje ten przycisk.



Rys. 3-4 Przyciski uruchamiania i zatrzymania sprawdzania obecności

3.2.1.3 Panel odczytanego numeru UID i odpowiedzi serwera

Panel odczytanego numeru UID i odpowiedzi serwera składa się z dwóch obiektów – ramki z numerem UID i tekstem odpowiedzi serwera.

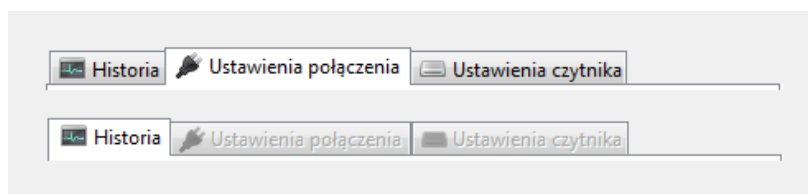


Rys. 3-5 Panel odczytanego numeru UID i odpowiedzi serwera

Szary kolor tła ramki z numerem UID oznacza brak odpowiedzi serwera, czerwony – błąd lub niepowodzenie operacji, zielony – powodzenie operacji.

3.2.1.4 Zakładki nawigacji pomiędzy oknami

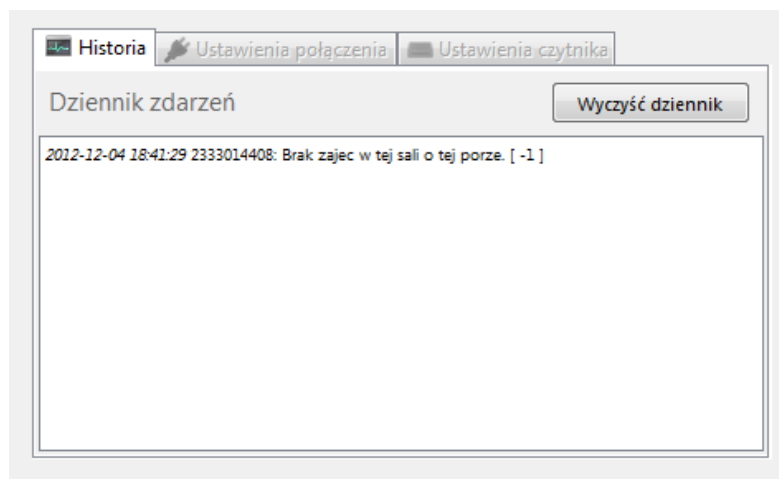
Kliknięcie aktywnej zakładki powoduje otwarcie odpowiedniego panelu. W stanie aktywnym sprawdzania obecności, zakładki ustawień są zablokowane. Dopiero zatrzymanie sprawdzania obecności odblokowuje te zakładki.



Rys. 3-6 Zakładki nawigacji pomiędzy oknami

3.2.1.5 Panel historii zdarzeń

Panel historii zdarzeń zawiera dziennik zdarzeń, do którego wpisywane są wszystkie odnotowane zbliżenia kart wraz z dokładnym czasem zbliżenia i odpowiedzią serwera.

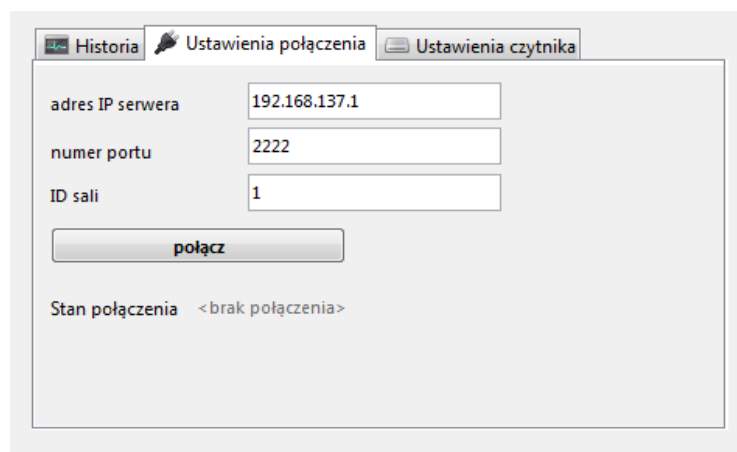


Rys. 3-7 Panel historii zdarzeń

3.2.1.6 Panel ustawień połączenia

Panel ustawień połączenia umożliwia wprowadzenie numeru IP serwera, numeru portu, przez który przebiegać będzie komunikacja klienta z serwerem oraz numer ID sali, z którą powiązana ma być ta aplikacja.

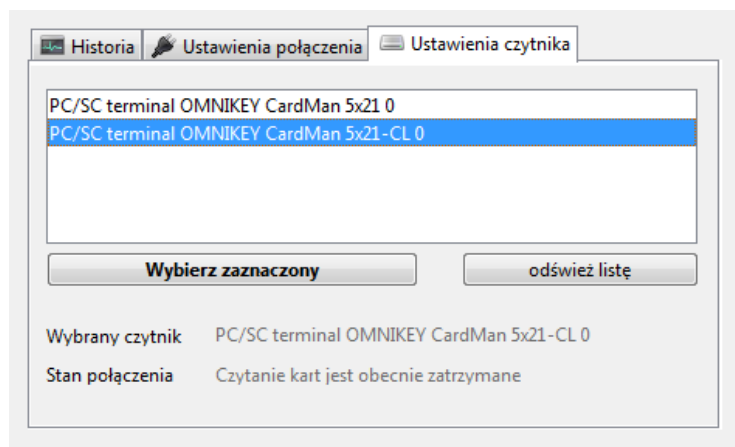
Przycisk *połącz* służy do sprawdzania stanu połączenia. W razie powodzenia, wyświetlony jest komunikat o udanym połączeniu z serwerem.



Rys. 3-8 Panel ustawień połączenia

3.2.1.7 Panel ustawień czytnika

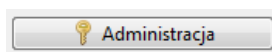
Panel ustawień czytnika wykrywa wszystkie czytniki kart podłączone do komputera. Do rozpoczęcia sprawdzania obecności wymagany jest wybór czytnika – należy odświeżyć listę czytników przyciskiem *odśwież listę*, następnie kliknąć kursorem myszy na wybranym czytniku i potwierdzić wybór przyciskiem *wybierz zaznaczony*.



Rys. 3-9 Panel ustawień czytnika

3.2.1.8 Przycisk otwierający panel zarządzania obecnościami

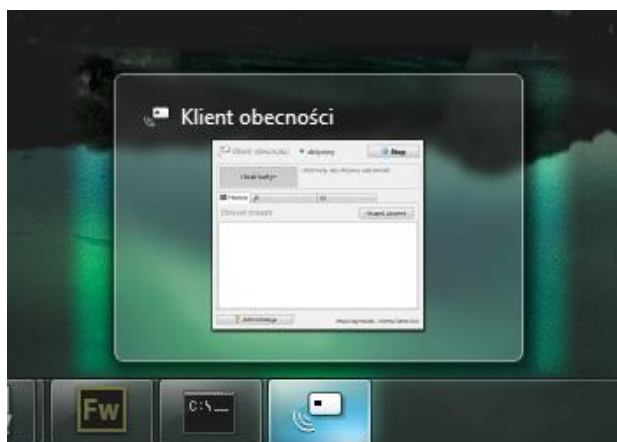
Klient posiada także przycisk szybkiego przejścia do panelu zarządzania obecnościami. Wciśnięcie przycisku *Administracja* spowoduje otwarcie nowego okna przeglądarki z wpisanym adresem panelu zarządzania.



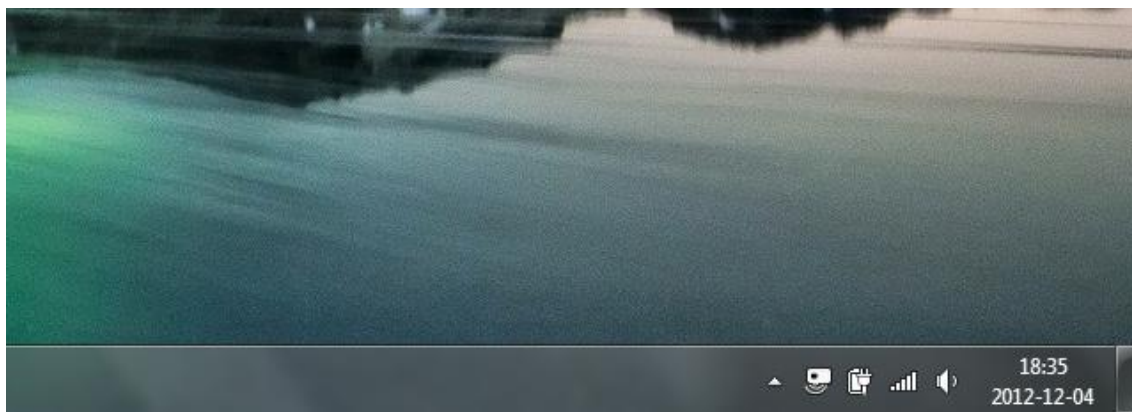
Rys. 3-10 Przycisk otwierający panel zarządzania obecnościami

3.2.2 Działanie w tle

Aplikacja klienta może zostać zminimalizowana do obszaru powiadomień. Aby zminimalizować aplikację, wystarczy nacisnąć przycisk *Zamknij*. Wówczas zniknie domyślna ikona z paska zadań (rys. 3-11), a pojawi się mała ikona w obszarze powiadomień (rys. 3-12).

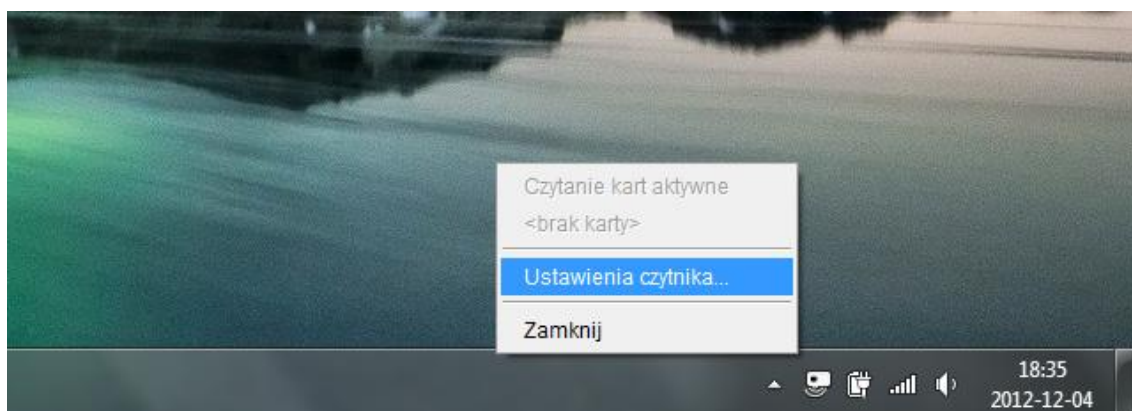


Rys. 3-11 Standardowa ikona okna głównego aplikacji klienta



Rys. 3-12 Ikona klienta w obszarze powiadomień

Aby przywrócić okno główne aplikacji klienta lub podejrzeć stan pracy, wystarczy nacisnąć prawym przyciskiem na ikonie. Spowoduje to rozwinięcie menu kontekstowego (rys. 3-13). Wybranie pozycji *Ustawienia czytnika...* otworzy okno główne aplikacji, *Zamknij* wyłączy całkowicie program. Menu kontekstowe wyświetla także informacje o stanie działania systemu obecności i o numerze UID aktualnie zbliżonej karty.

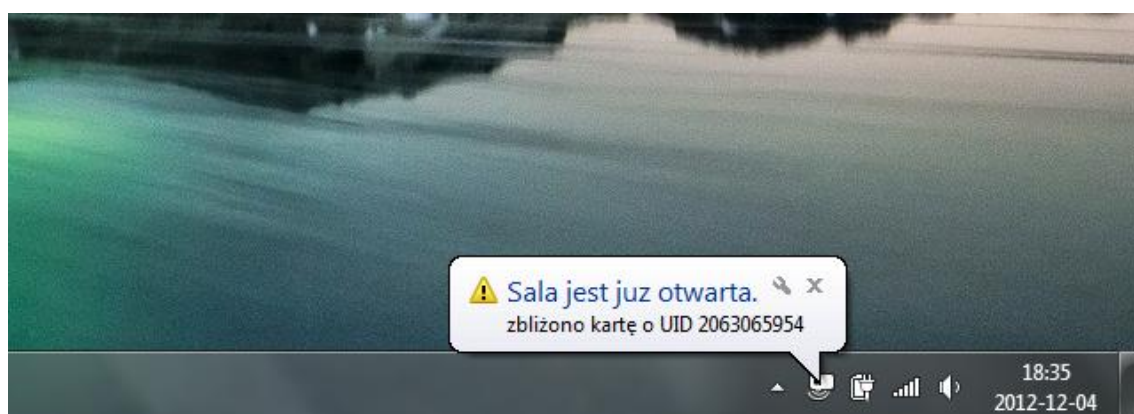


Rys. 3-13 Menu kontekstowe ikony klienta obecności w obszarze powiadomień

Każde zarejestrowane zbliżenie karty do czytnika jest sygnalizowane użytkownikowi przez program za pomocą dymków z powiadomieniami, które zawierają sygnalizację powodzenia operacji, treść odpowiedzi serwera oraz numer UID zbliżonej karty (rys. 3-14, rys. 3-15 i rys. 3-16).



Rys. 3-14 Powiadomienia – informacja



Rys. 3-15 Powiadomienia – ostrzeżenie



Rys. 3-16 Powiadomienia – błąd

3.3 Panel zarządzania

Wskazówki instalacji, konfiguracji i uruchomienia panelu zarządzania obecnościami znajdują się w rozdziale 5.3 Instalacja i konfiguracja panelu zarządzania.

Opis zastosowanych rozwiązań programistycznych, wzorców projektowych, klas i funkcji znajduje się w rozdziale 8.3 Dodatek - Kod źródłowy – panel zarządzania.

Panel zarządzania umożliwia nieograniczoną administrację wszystkimi danymi systemu obecności, czyli danymi studentów, prowadzących, sal, kursów, zajęć, obecności i zapisów.

3.3.1 Autoryzacja

Rys. 3-17 przedstawia okno logowania do panelu zarządzania, które wyświetlane jest po uruchomieniu panelu zarządzania w oknie przeglądarki.



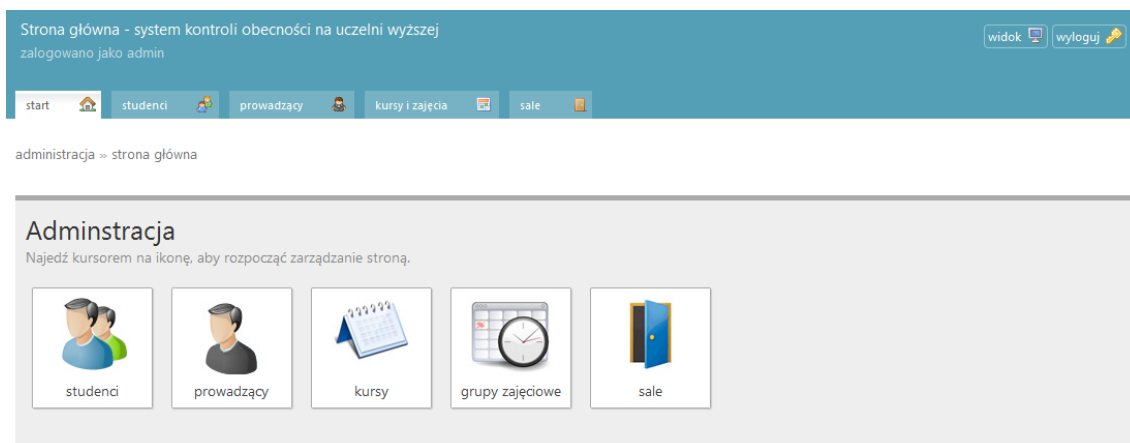
Rys. 3-17 Okno logowania do panelu zarządzania

W polach *login* i *hasło* należy wpisać poprawne dane użytkownika, a następnie nacisnąć przycisk *Zaloguj się*, aby się zalogować do systemu. Pięciokrotna nieudana próba autoryzacji domyślnie powoduje zablokowanie możliwości logowania się do końca trwania sesji, czyli aż do ponownego uruchomienia przeglądarki.

Administracja użytkowników mających dostęp do panelu zarządzania opisana jest szczegółowo w rozdziale 5.3.2 Instalacja i konfiguracja panelu zarządzania.

3.3.2 Okno główne panelu zarządzania

Po pomyślnym przejściu autoryzacji, wyświetlony zostaje widok okna głównego panelu zarządzania (rys. 3-18).

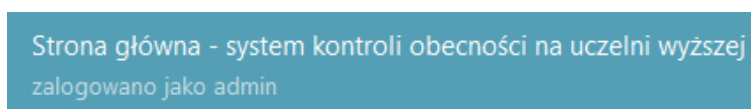


Rys. 3-18 Okno główne panelu zarządzania


Elementami stałymi każdego widoku w panelu zarządzania są:




- tytuł strony panelu zarządzania (rys. 3-19),
- przełącznik widoku i przycisk wylogowania panelu zarządzania (rys. 3-20),
- nawigacja główna panelu zarządzania (rys. 3-21),
- lokalizacja w panelu zarządzania (rys. 3-22).

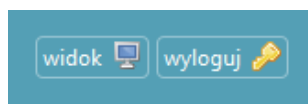
W tytule strony (rys. 3-19) wyświetlana jest nazwa obecnie otwartego widoku, nazwa systemu oraz nazwa użytkownika, który jest obecnie zalogowany i korzysta z systemu.








Rys. 3-19 Tytuł strony panelu zarządzania

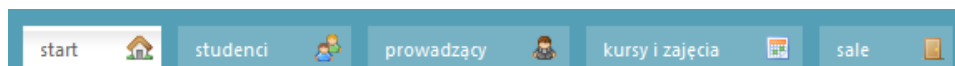
Przełącznik widoku (rys. 3-20) umożliwia wylogowanie użytkownika z systemu za pomocą przycisku  *wyloguj*, oraz zmianę sposobu wyświetlania strony w trzech wariantach:

-  – dopasowanie rozmiaru strony do rozdzielczości ekranu,
-  – widok wąski o stałej szerokości (domyślny),
-  – widok szeroki o stałej szerokości.



Rys. 3-20 Przełącznik widoku i przycisk wylogowania panelu zarządzania

Nawigacja główna (rys. 3-21) pozwala na przemieszczanie się pomiędzy podstawowymi widokami panelu administracyjnego za pomocą zakładek  *start*,  *studenci*,  *prowadzący*,  *kursy i zajęcia*,  *sale*.



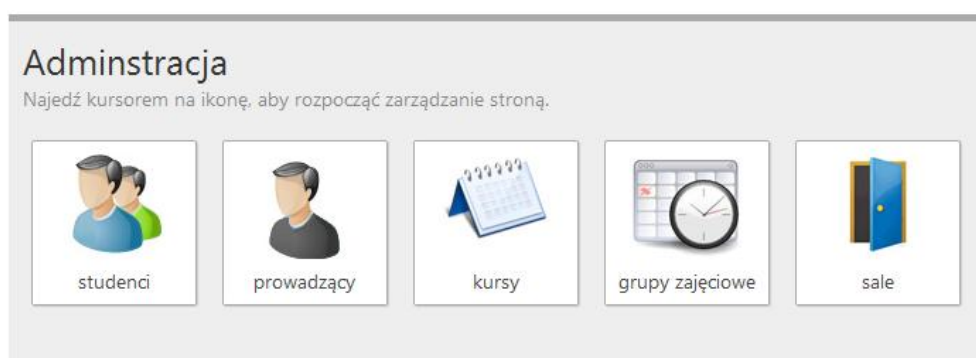
Rys. 3-21 Nawigacja główna panelu zarządzania

Przydatnym elementem nawigacyjnym jest również pasek lokalizacji (historii lub ścieżki), gdzie zgodnie z hierarchią wykazane są kolejne widoki, przez które użytkownik dotarł do widoku obecnego (rys. 3-22). Kliknięcie nazwy widoku spowoduje jego otwarcie.

administracja » strona główna

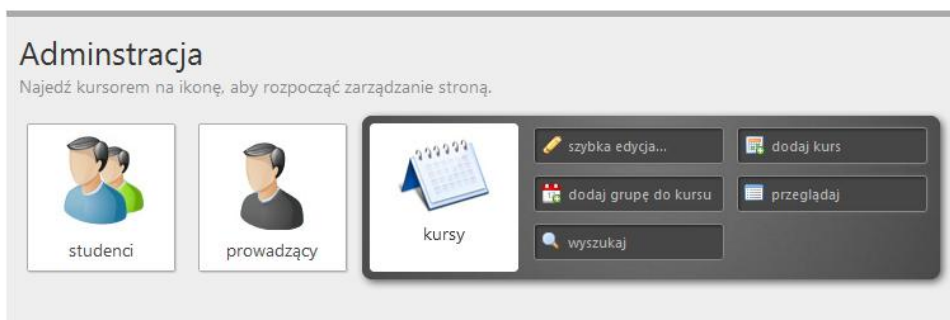
Rys. 3-22 Lokalizacja w panelu zarządzania

Poza wyżej wymienionymi podstawowymi elementami interfejsu, które widoczne są na każdej podstronie systemu, ważnym i przydatnym elementem nawigacyjnym jest okno szybkiego wyboru, dostępne wyłącznie z widoku strony głównej (rys. 3-23, rys. 3-24).



Rys. 3-23 Okno szybkiego wyboru na stronie głównej

Przeniesienie kursora nad ikonę powoduje rozwinięcie menu z przyciskami umożliwiającymi szybkie przejście do każdego widoku w systemie.

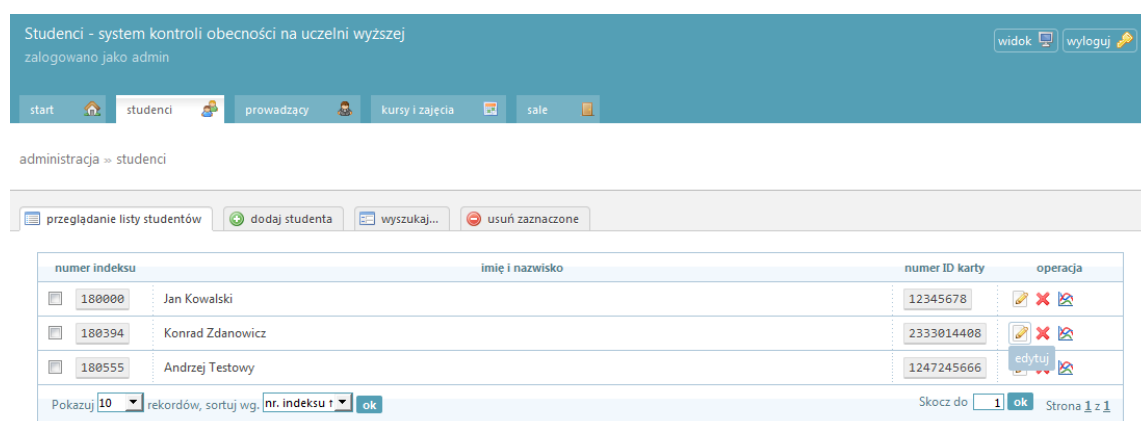


Rys. 3-24 Okno szybkiego wyboru na stronie głównej z rozwiniętym menu „kursy”

Okno szybkiego wyboru umożliwia przeszukiwanie, dodawanie, przeglądanie i edycję studentów, prowadzących, kursów, sal, grup zajęciowych, zapisów studenta, dzienników i list obecności za pomocą jednego kliknięcia myszy.

3.3.3 Studenci i prowadzący

Wybranie zakładki *studenci* z nawigacji głównej panelu zarządzania otwiera widok zarządzania/edycji studentów, *prowadzący* otwiera widok zarządzania/edycji prowadzących (rys. 3-25).



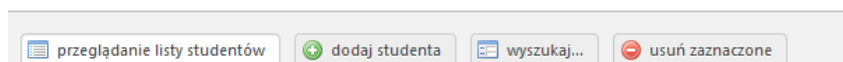
Rys. 3-25 Widok listy studentów

Interfejs widoku studentów i prowadzących (oraz wszystkich pozostałych widoków przeglądania listy rekordów) składa się ze stałych elementów:


- nawigacja widoku (rys. 3-26),
- tabela rekordów (rys. 3-29),
- kolumna identyfikatora (rys. 3-30),
- kolumna wyboru operacji (rys. 3-31),

e) pasek sortowania i nawigacji między stronami (rys. 3-32).

Nawigacja widoku studentów (rys. 3-26) pozwala na przełączanie między widokami przeglądania listy studentów/prowadzących lub dodawania/edycji studenta/prowadzącego, wyszukiwaniem i usuwaniem zaznaczonych rekordów.



Rys. 3-26 Nawigacja widoku

Naciśnięcie zakładki  *wyszukaj...* powoduje rozwinięcie formularza wyszukiwania (rys. 3-27). Umożliwia on przeszukanie listy studentów według numerów albumu, nazwiska lub numeru UID karty.


szukaj 23 szukaj

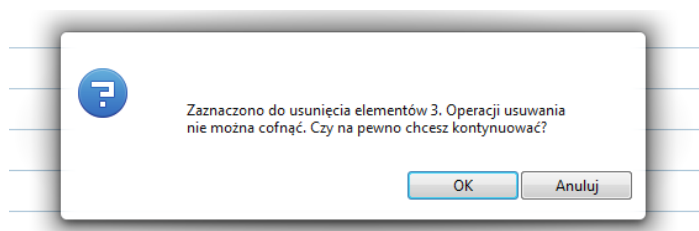
obszar wyszukiwań ☒ numery albumu ☐ nazwiska ☒ numery ID karty

przeglądanie listy studentów dodaj studenta wyszukiwanie usuń zaznaczone

Wyniki wyszukiwania dla frazy 23 (2 wyników)










Rys. 3-27 Formularz wyszukiwania studentów


Naciśnięcie zakładki  *usuń zaznaczone...* spowoduje otwarcie okna dialogowego potwierdzającego chęć usunięcia zaznaczonych elementów (rys. 3-28).



Rys. 3-28 Usuwanie zaznaczonych rekordów

Lista studentów wyświetlana jest w postaci tabeli rekordów (rys. 3-29).

numer indeksu	imię i nazwisko	numer ID karty	operacja
<input type="checkbox"/> 180000	Jan Kowalski	12345678	  
<input type="checkbox"/> 180394	Konrad Zdanowicz	2333014408	  
<input type="checkbox"/> 180555	Andrzej Testowy	1247245666	  




Pokazuj 10 rekordów, sortuj wg. nr. indeksu  ok Skocz do 1 ok Strona 1 z 1










Rys. 3-29 Tabela rekordów

Każda tabela rekordów posiada stałe elementy, niezależnie od widoku, który aktualnie obsługuje. Pierwsza kolumna tabeli jest zawsze listą unikalnych identyfikatorów (rys. 3-30). W przypadku listy studentów, są to numery indeksu. Obok identyfikatora znajduje się pole *checkbox*, które umożliwia zaznaczenie kilku rekordów i operację na nich.

numer indeksu	
<input type="checkbox"/>	180000
<input type="checkbox"/>	180394
<input type="checkbox"/>	180555

Rys. 3-30 Kolumna identyfikatora

Operacja to z kolei ostatnia kolumna każdej tabeli rekordów (rys. 3-31). Umożliwia wykonanie operacji na pojedynczym widoku; są to akcje  – edycji,  – usunięcia rekordu i  – wyświetlenia statystyk obecności. Najechnięcie kursorem na ikonę powoduje wyświetlenie tekstu podpowiedzi.



operacja		
		
		
		

Rys. 3-31 Kolumna wyboru operacji

W stopce tabeli zawsze widoczny jest pasek sortowania i nawigacji między stronami (rys. 3-32). Pozwala na zmianę liczby wyświetlanych rekordów i sposobu ich sortowania. Przy prawej krawędzi stopki dostępna jest funkcja szybkiego skoku do pożądanej strony rekordów, wyświetlany jest numer obecnej strony i numer wszystkich stron, oraz – jeżeli stron jest kilka – pojawiają się przyciski nawigacyjne *poprzednia strona* i *następna strona*.

Pokazuj <input type="text" value="10"/> rekordów, sortuj wg. <input type="text" value="nr. indeksu ↑"/> <input type="button" value="ok"/>	Skocz do <input type="text" value="1"/> <input type="button" value="ok"/>	Strona 1 z 1
-------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------	--------------

Rys. 3-32 Pasek sortowania i nawigacji między stronami

Wybranie zakładki  *dodaj studenta/dodaj prowadzącego* lub naciśnięcie ikony akcji  *edycja studenta/edycja prowadzącego* w wierszu rekordu otworzy widok dodawania/edycji

studenta, z poziomu którego możliwa jest modyfikacja danych istniejącego studenta bądź utworzenie nowego. Rys. 3-33 przedstawia przykładowe okno edycji studenta.

administracja » studenci » modyfikacja danych studenta

przeglądanie listy studentów modyfikacja danych studenta

modyfikacja danych studenta		
numer indeksu Numer indeksu studenta.	<input type="text" value="180394"/>	
Imię Imię studenta.	<input type="text" value="Konrad"/>	✓
Nazwisko Nazwisko studenta	<input type="text" value="Zdanowicz"/>	✓
UID karty Unikalny numer identyfikacyjny karty	<input type="text"/>	✗ To pole jest wymagane

zapisz

Rys. 3-33 Widok edycji studenta

Każde pole formularza jest zatytułowane i podpisane tak, aby użytkownik nie miał wątpliwości, jaki format danych należy wpisać. Naciśnięcie przycisku *zapisz* powoduje dodanie/modyfikację rekordu.

Szare tło pola oznacza, że użytkownik nie ma prawa modyfikacji tego pola. Panel zarządzania nie pozwala na modyfikację identyfikatora rekordu (w tym przypadku – numer indeksu). System także dynamicznie sprawdza poprawność wprowadzanych danych – na bieżąco informuje użytkownika o tym, czy pola są wypełnione prawidłowo, to znaczy:

- wymagane pole są wypełnione,
- wpisany ciąg znaków ma odpowiednią długość,
- typ/format danych w polu jest poprawny (np. pole zawiera tylko cyfry/ poprawną wartość godziny/datę),
- wpisane dane są unikalne.

Jeżeli którekolwiek z wpisanych danych są niepoprawne, naciśnięcie przycisku wysłania formularza nie powoduje żadnej akcji, ale wyświetla użytkownikowi komunikat o błędzie. Rys. 3-33 przedstawia próbę wysłania formularza, w którym wymagane pole jest puste, zaś rys. 3-34 przedstawia reakcję systemu na próbę przypisania studentowi UID karty, który istnieje już w bazie danych.

UID karty
Unikalny numer identyfikacyjny karty


12345678

✖ Istnieje w bazie użytkownik o tym numerze ID karty



Rys. 3-34 Asynchroniczne sprawdzanie unikalności wpisywanych danych












Numer UID karty musi być unikalny, dlatego przed wysłaniem formularza sprawdzane jest, czy w bazie nie istnieje prowadzący lub student, który ma przypisany taki sam numer UID. Naruszenie jednoznaczności relacji „użytkownik–numer UID” uniemożliwiłoby poprawne działanie systemu kontroli obecności. Na tej samej zasadzie kontrolowana jest unikalność wpisanego numeru albumu i kodu kursu przy dodawaniu rekordu.

3.3.4 Sale

Naciśnięcie zakładki *sale* z nawigacji głównej otworzy widok przeglądania sal (rys. 3-35). Widoki wyświetlania, dodawania i edycji sal wyglądają i działają bardzo podobnie do poprzednich widoków. Naciśnięcie zakładki  *dodaj salę* otwiera okno dodawania/edycji sali. Identyfikator sali tworzony jest automatycznie, nie ma możliwości jego modyfikacji.

administracja » sale




 przeglądanie sal  dodaj salę  usuń zaznaczone

	id sali	sala	budynek	operacja
	11	019	C-3	 
	5	114	D-20	 
	7	208	C-3	 
	10	21	C-3	 
	12	215	C-3	 
	8	305	C-3	 
	9	319	C-3	 

Pokazuj 10 rekordów, sortuj wg. numer sali t ok Skocz do 1 ok Strona 1 z 1

Rys. 3-35 Widok listy sal

3.3.5 Kursy

Wybranie zakładki *kursy* z głównej nawigacji otworzy widok listy kursów (rys. 3-36). Naciśnięcie na nazwę kursu otworzy listę zajęć kursu (rozdz. 3.3.6 Grupy zajęciowe). Obok nazwy kursu w liście widnieje nazwa formy kursu oraz wskaźnik  liczby grup zajęciowych utworzonych do tego kursu. Naciśnięcie zakładki  *dodaj kurs* lub ikony  operacji edycji otworzy panel edycji kursu (rys. 3-37).

administracja » kursy

przeglądanie listy kursów + dodaj kurs wyszukaj... - usuń zaznaczone

kod kursu	forma, nazwa kursu [liczba zajęć]	ECTS	operacja
<input type="checkbox"/> ARES00510P	projekt E-media 3	2	
<input type="checkbox"/> ARES00510W	wykład E-media 3	2	
<input type="checkbox"/> ARES00509W	wykład Inteligentne budynki 1	2	
<input type="checkbox"/> TEST0001	ćwiczenia Nazwa testowa 2	0	
<input type="checkbox"/> ZMZ000340W	wykład Podstawy zarządzania jakością 1	2	
<input type="checkbox"/> ARES00506P	projekt Projekt specjalnościowy ART 4	2	
<input type="checkbox"/> ARES00409S	seminarium Seminarium dyplomowe 4	2	
<input type="checkbox"/> ARES00507P	projekt Sieci neuronowe i neurosterowniki 3	2	
<input type="checkbox"/> ARES00508W	wykład Technologie WWW 1	2	

Pokazuj 10 rekordów, sortuj wg nazw kursów ↑ ok Skocz do 1 ok Strona 1 z 1

Rys. 3-36 Widok listy kursów

administracja » kursy » modyfikacja kursu

przeglądanie listy kursów + modyfikacja kursu

modyfikacja kursu

Kod kursu
Unikalny kod kursu: ARES00409S

Nazwa kursu
Nazwa kursu: Seminarium dyplomowe

Forma kursu
Forma kursu:
☐ Wykład
☐ Ćwiczenia
☐ Laboratorium
☐ Projekt
☒ Seminarium
☐ Inne

ECTS
Liczba punktów ECTS za kurs (jeżeli 0 - zostaw puste): 2

zapisz

Rys. 3-37 Widok edycji kursu

Przy dodawania/modyfikacji kursów sprawdzana jest unikalność kod kursu. Forma kursu zaś powinna być elementem wybranym z listy form kursu.

3.3.6 Grupy zajęciowe

Kliknięcie na nazwie kursu powoduje otwarcie widoku listy grup zajęciowych dla tego kursu (rys. 3-38). W widoku listy zajęć zestawione są dzień, godzina i sala,

w której odbywają się zajęcia wraz z informacjami o liczbie terminów, zapisanych studentach i nazwiskiem prowadzącego.

administracja » kursy » Projekt specjalnościowy ART (Projekt)

--	--	--	--

id	dzień, tydzień, godzina zajęć	ilość terminów, studentów zapisanych, prowadzący	operacja
21	poniedziałek 16:10 - 18:45 sala 305, C-3	0 0 Dr inż. M. Gorczyca	
22	wtorek 16:00 - 18:15 sala 319, C-3	2 2 Dr inż. A. Rusiecki	
23	wtorek 18:30 - 20:35 sala 319, C-3	0 0 Dr inż. A. Rusiecki	
24	środa 10:45 - 13:00 sala 319, C-3	0 0 Dr inż. . Halawa	

Pokazuj rekordów, sortuj wg:

Skocz do Strona 1 z 1

Rys. 3-38 Widok listy grup zajęciowych

Kliknięcie terminu z kolumny *dzień, tydzień, godzina zajęć* otworzy widok listy terminów (rozdz. 3.3.7 Terminy). Naciśnięcie ikony w kolumnie *ilość terminów, studentów zapisanych, prowadzący* spowoduje otwarcie odpowiedniego widoku, w zależności od wybranej ikony, są to kolejno:

- a) – lista terminów danej grupy zajęciowej,
- b) – lista zapisanych studentów,
- c) – edycja prowadzącego.

Naciśnięcie ikony operacji edycji otworzy okno edycji wybranej grupy zajęciowej.

administracja » kursy » Projekt specjalnościowy ART (Projekt) » dodaj zajęcia

przeglądanie listy kursów

przeglądanie listy zajęć

dodaj zajęcia

dodaj zajęcia

Numer ID zajęć

Wartość jest przydzielana automatycznie

(przydzielany automatycznie)

Prowadzący

ID prowadzącego zajęcia

22 test

dr hab. mgr inż. Piotr Testowy

Sala

Sala, w której odbywają się zajęcia

208, C-3

✓

Dzień tygodnia

Dzień, w którym odbywają się zajęcia

Wtorek

✓

Tydzień

Wybierz, czy zajęcia mają odbywać się co tydzień, czy co dwa tygodnie

co tydzień

☒ nieparzysty

parzysty

Godzina rozpoczęcia

Wybierz godzinę rozpoczęcia zajęć (HH:MM)

15 : 15

✓ ✓

Godzina zakończenia

Wpisz godzinę zakończenia zajęć (HH:MM)

16 : 45

✓ ✓

Kod kursu

Kod kursu, do którego przypisane zostaną zajęcia

ARES00506P

zapisz

Rys. 3-39 Widok modyfikacji grupy zajęciowej

Różnicą w stosunku do poprzednich widoków modyfikacji rekordów jest pole wyboru prowadzącego zajęcia. Aby powiązać prowadzącego z grupą zajęciową, należy w polu wyszukiwania (obok zablokowanego pola numeru ID prowadzącego) wpisać fragment nazwiska prowadzącego, aby system wyszukał ciągu znaków w bazie danych i pokazał listę sugestii prowadzących, których nazwisko pasuje do wzorca. Jeśli użytkownik nie zdecyduje się na wybranie odpowiedniego prowadzącego z listy, dodanie grupy zajęciowej nie powiedzie się. Kliknięcie zaś wyszukanego prowadzącego spowoduje automatyczne wpisanie jego numeru ID do – zablokowanego przed ręczną edycją – pola obok.

System sprawdza także poprawność wpisywanych godzin i minut – niepoprawne jest wpisanie godziny mniejszej niż 6 i większej niż 22.

3.3.7 Terminy

Widok listy terminów (rys. 3-40) osiągalny jest z poziomu poprzedniego widoku – listy grup zajęciowych poprzez kliknięcie dnia i godziny grupy zajęciowej.

S

administracja » kursy » Projekt specjalnościowy ART (Projekt) » wtorek 16:00-18:15, sala 319, C-3

przeglądanie listy kursów	przeglądanie listy zajęć	lista terminów	tabela obecności	zapisani studenci	dodaj termin	usuń zaznaczone
---------------------------	--------------------------	----------------	------------------	-------------------	--------------	-----------------

id	data zajęć	obecności	otwarcie sali	operacja
69	02.10.2012	1 / 2	16:00:00	
52	09.10.2012	1 / 2	16:09:52	

kliknij, aby otworzyć listę obecności dla tego terminu

Pokazuj 10 rekordów, sortuj wg: Data ↑ ok Skocz do 1 ok Strona 1 z 1

Rys. 3-40 Widok listy terminów

W widoku listy terminów zawarte są wszystkie terminy zajęć grupy, które się odbyły. Obok daty zajęć widnieje ikona liczby studentów obecnych i godziny otwarcia sali. Kliknięcie kursorem myszy na datę terminu otworzy widok listy obecności z tego dnia (rys. 3-42), a wybranie ikony edycji otworzy widok modyfikacji terminu (rys. 3-41). Do nawigacji można także posłużyć się menu nawigacyjnym widoku. Wybranie zakładki *tabela obecności* otworzy tabelę z listą studentów i obecnościami w każdym z terminów (rys. 3-43), kliknięcie zakładki *zapisani studenci* otworzy widok listy zapisanych studentów (rys. 3-46), wybranie zakładki *dodaj termin* otworzy widok dodania terminu (rys. 3-41).

administracja » kursy » Projekt specjalnościowy ART (Projekt) » wtorek TN 15:15-16:45, sala 208, C-3 » dodaj termin

przeglądanie listy kursów	przeglądanie listy zajęć	przeglądanie listy terminów	dodaj termin
---------------------------	--------------------------	-----------------------------	--------------

dodaj termin

Numer ID terminu

Wartość jest przydzielana automatycznie

Data zajęć

Dzień, w którym odbyły się zajęcia (YYYY-MM-DD)

Godzina rozpoczęcia

Godzina rozpoczęcia zajęć (HH:MM:SS)

Grudzień 2012

Tydz	Pn	Wt	Śr	Cz	Pt	So	N
48	26	27	28	29	30	1	2
49	3	4	5	6	7	8	9
50	10	11	12	13	14	15	16
51	17	18	19	20	21	22	23
52	24	25	26	27	28	29	30
1	31	1	2	3	4	5	6

Rys. 3-41 Widok modyfikacji terminu

W widoku modyfikacji terminu, data zajęć wybierana jest z kalendarza, gdzie dostępne są wyłącznie dni, w których mogą odbywać się zajęcia. Na rysunku wyżej zajęcia powinny odbywać się we wtorek, w tygodniu nieparzystym, dlatego pozostałe dni

i tygodnie parzyste są zablokowane i nie można ich wybrać. System także nie pozwala na tworzenie zajęć jednej grupy w tym samym dniu.

3.3.8 Listy obecności

Kliknięcie kursorem myszy na datę terminu z widoku listy terminów otworzy widok listy obecności z tego dnia (rys. 3-42).

administracja » kursy » Projekt specjalnościowy ART (Projekt) » wtorek 16:00-18:15, sala 319, C-3 » lista obecności

przeglądanie listy kursów	przeglądanie listy zajęć	lista terminów	tabela obecności	lista obecności
---------------------------	--------------------------	----------------	------------------	-----------------

nr indeksu	obecny	imię i nazwisko studenta
180000	✗	Jan Kowalski
180394	✓	Konrad Zdanowicz

Pokazuj 10 rekordów, sortuj wg: Nazwisko ↑ ok Skocz do 1 ok Strona 1 z 1

Rys. 3-42 Widok listy obecności


W tabeli widoku listy obecności zamieszczona jest lista studentów zapisanych do wybranej grupy zajęciowej wraz ze wskaźnikami obecności studenta na zajęciach. Ikona ✓ oznacza obecność, ✗ – nieobecność, ⌚ – spóźnienie na zajęcia. Dostępny jest także widok tabeli obecności (rys. 3-43).

administracja » kursy » Projekt specjalnościowy ART (Projekt) » wtorek 16:00-18:15, sala 319, C-3 » tabela obecności

przeglądanie listy kursów	przeglądanie listy zajęć	lista terminów	tabela obecności	zapisani studenci	dodaj termin
---------------------------	--------------------------	----------------	------------------	-------------------	--------------

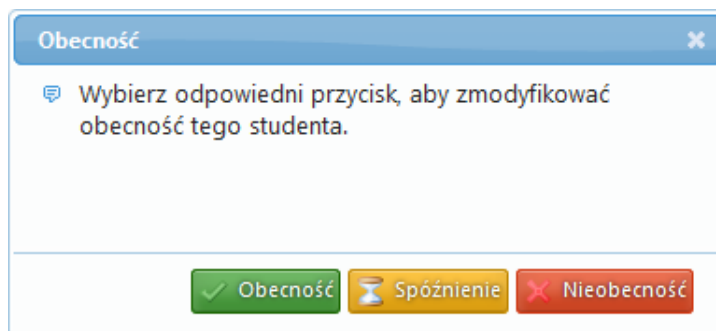
student		02.10	09.10
180000	Jan Kowalski	✗	⌚
180394	Konrad Zdanowicz	✓	✗
dodaj studenta do tych zajęć...		dodaj termin...	nieobecny

Rys. 3-43 Widok tabeli obecności

Naciśnięcie przycisku *dodaj studenta do tych zajęć...* otwiera widok zapisów studenta (rys. 3-47), naciśnięcie przycisku  *dodaj termin* otworzy widok dodawania terminu (rys. 3-41). Kliknięcie na datę terminu, podkreśloną linią przerywaną, otwiera widok modyfikacji terminu (rys. 3-41).

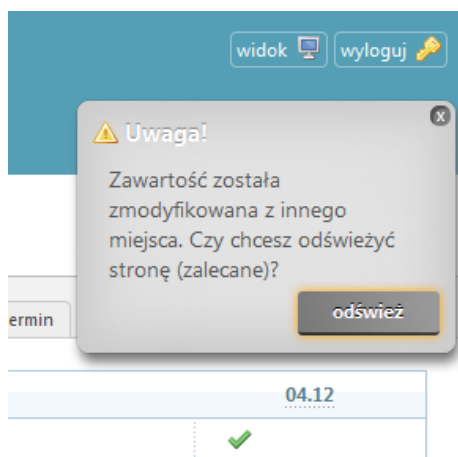
Aby wpisać studentowi obecność, lub zmodyfikować jej stan na spóźnienie lub nieobecność, należy nacisnąć kursorem myszy na wskaźnik obecności. Spowoduje to otwarcie okna wyboru typu obecności (rys. 3-44). Kliknięcie przycisku spowoduje

asynchroniczne wpisanie odpowiedniego typu obecności i odświeżenie ikony wskaźnika obecności.



Rys. 3-44 Okno wyboru typu obecności

System zarządzania obecnościami wyposażony jest w asynchroniczny mechanizm ostrzeżenia o modyfikacji obecnie przeglądanej listy/tabeli obecności. W sytuacji, gdy prowadzący przegląda listę obecności, a w tym samym czasie student z grupy zajęciowej odnotuje swoją obecność zbliżając legitymację do czytnika kart, w widoku listy/tabeli obecności pojawi się powiadomienie z informacją, że zawartość dziennika obecności, którego widok jest przeglądany, została zmodyfikowana. Naciśnięcie przycisku *odśwież* spowoduje aktualizację wskaźników obecności.



Rys. 3-45 Ostrzeżenie o aktualizacji danych

3.3.9 Zapisy studentów

Kliknięcie zakładki *zapisani studenci* z widoku terminów otworzy widok listy zapisanych studentów (rys. 3-46).

administracja » kursy » Projekt specjalnościowy ART (Projekt) » wtorek 16:00-18:15, sala 319, C-3 » zapisy

numer indeksu	imię i nazwisko	operacja
<input type="checkbox"/> 180000	Jan Kowalski	
<input type="checkbox"/> 180394	Konrad Zdanowicz	

Pokazuj

10

 rekordów, sortuj wg:

Nazwisko

ok

Skocz do

1

ok

 Strona

1

 z

1

Rys. 3-46 Widok listy zapisanych studentów

Naciśnięcie ikony spowoduje wypisanie wybranego studenta z zajęć. Naciśnięcie przycisku otworzy widok zapisów studenta na zajęcia (rys. 3-47).

administracja » kursy » Projekt specjalnościowy ART (Projekt) » wtorek 16:00-18:15, sala 319, C-3 » zapisy » dodaj studenta

przeglądanie listy kursów	przeglądanie listy zajęć	lista terminów	tabela obecności	zapisani studenci	
modyfikacja terminu					
Student					
Wpisz imię lub nazwisko studenta w dużym polu, aby wyszukać i wybrać					
<input type="text" value="180"/>					
<div>(180394) Konrad Zdanowicz</div> <div>(180555) Andrzej Testowy</div> <div> (180000) Jan Kowalski</div>					

Rys. 3-47 Widok okna zapisów

Wybór studenta, który ma być zapisany na zajęcia, wykonywany jest podobnie do wprowadzania prowadzącego grupy zajęciowej – w polu tekstowym należy wpisać imię, nazwisko lub numer indeksu studenta. System wyszuka wpisanego ciągu tekstowego w bazie studentów i zwróci listę sugestii. Wybranie studenta z listy podpowiedzi spowoduje wpisanie numeru indeksu do pola obok.

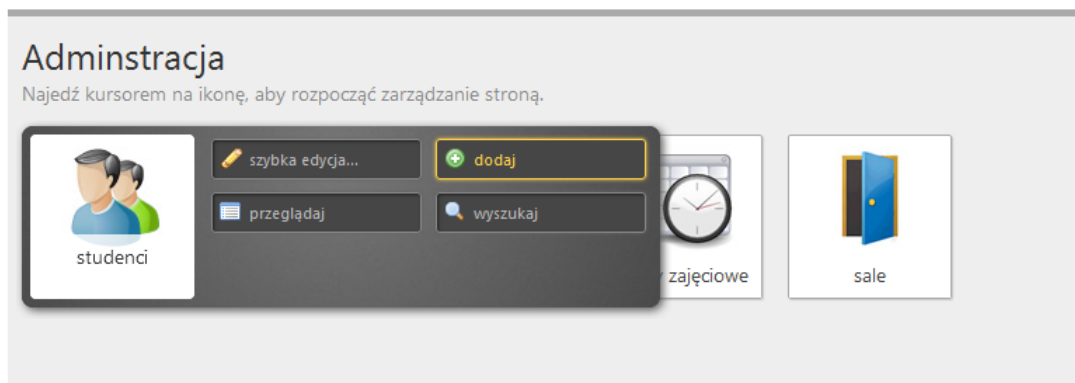
4 Scenariusz użycia

W tym dziale zawarty jest scenariusz użycia systemu obecności do wprowadzenia przykładowych danych w panelu zarządzania, a następnie sprawdzenia obecności za pomocą klienta obecności.

Uruchomić serwer, a następnie otworzyć panel zarządzania wpisując w pasku adresu przeglądarki adres `http://localhost/systemobecnosci/`, zalogować się wcześniej wprowadzonymi danymi użytkownika. Szczegółowe instrukcje konfiguracji i uruchomienia serwera, klienta i panelu zarządzania systemem kontroli obecności znaleźć można w rozdz. 5 Instalacja i uruchomienie oprogramowania.

4.1 Utworzenie konta studenta i prowadzącego

Z menu szybkiego wyboru najechać kursorem na ikonę *studenci*, a następnie wybrać przycisk *dodaj* (rys. 4-1). Wypełnić formularz danymi, a następnie nacisnąć przycisk *zapisz* (rys. 4-2).



Rys. 4-1 Menu szybkiego wyboru – dodaj studenta

dodaj studenta		
numer indeksu Numer indeksu studenta.	<input type="text" value="123456"/>	✓
Imię Imię studenta.	<input type="text" value="Jan"/>	✓
Nazwisko Nazwisko studenta	<input type="text" value="Student"/>	✓
UID karty Unikalny numer identyfikacyjny karty	<input type="text" value="2333014408"/>	✓

[zapisz](#)

Rys. 4-2 Formularz dodawania studenta

Jeżeli akcja przebiegła poprawnie, powinien zostać wyświetlony komunikat o sukcesie operacji (rys. 4-3).

Akcja przebiegła poprawnie. Naciśnij powrót, aby wrócić do listy studentów.
← powrót

Rys. 4-3 Powodzenie dodania studenta

Za pomocą zakładki *start* przejść na stronę główną, i za pomocą szybkiej nawigacji utworzyć formularz dodawania prowadzących. Wypełnić formularz przykładowymi danymi (rys. 4-4) i nacisnąć przycisk *zapisz*. Jeżeli akcja przebiegła poprawnie, powinien zostać wyświetlony komunikat o sukcesie operacji.

dodaj prowadzącego		
numer ID prowadzącego Wartość jest przydzielana automatycznie	(przydzielany automatycznie)	
Tytuł Tytuł naukowy prowadzącego	<input type="text" value="mgr inż."/>	✓
Imię Imię prowadzącego	<input type="text" value="Anna"/>	✓
Nazwisko Nazwisko prowadzącego	<input type="text" value="Prowadząca"/>	✓
UID karty Unikalny numer identyfikacyjny karty	<input type="text" value="1247245666"/>	✓

zapisz

Rys. 4-4 Formularz dodawania prowadzącego

4.2 Utworzenie sali

Z menu szybkiego wyboru najechać kursorem na ikonę *sale*, a następnie wybrać przycisk *dodaj*. Wypełnić formularz dodawania sal i nacisnąć przycisk *zapisz* (rys. 4-5). Jeżeli akcja przebiegła poprawnie, powinien zostać wyświetlony komunikat o sukcesie operacji.

dodaj salę		
numer ID sali Wartość jest przydzielana automatycznie	<input type="text"/>	
Numer sali Numer sali	<input type="text" value="123"/>	✓
Budynek Budynek, w którym znajduje się sala	<input type="text" value="C-1"/>	✓

zapisz

Rys. 4-5 Formularz dodawania sali

4.3 Utworzenie kursu i grupy zajęciowej

Analogicznie jak w poprzednich podpunktach, otworzyć okno dodawania kursu za pomocą menu szybkiego wyboru i przycisku *dodaj kurs*. Formularz dodawania kursu (rys. 4-6) wypełnić przykładowymi danymi. Dane zatwierdzić przyciskiem *zapisz*.

dodaj kurs

Kod kursu <small>Unikalny kod kursu</small>	<input type="text" value="TEST00001"/>	✓
Nazwa kursu <small>Nazwa kursu</small>	<input type="text" value="Test"/>	✓
Forma kursu <small>Forma kursu</small>	<div style="display: flex; flex-direction: column; gap: 5px;"> <div><input type="radio"/> Wykład</div> <div><input type="radio"/> Ćwiczenia</div> <div><input checked="" type="radio"/> Laboratorium</div> <div><input type="radio"/> Projekt</div> <div><input type="radio"/> Seminarium</div> <div><input type="radio"/> Inne</div> </div>	
ECTS <small>Liczba punktów ECTS za kurs (jeżeli 0 - zostaw puste)</small>	<input type="text" value="1"/>	✓

zapisz

Rys. 4-6 Formularz dodawania kursu

Jeżeli akcja przebiegła poprawnie, powinien zostać wyświetlony komunikat o sukcesie operacji. Nacisnąć przycisk *powrót*, aby wrócić do widoku listy kursów. Odnaleźć na liście kursów nowo dodany i kliknąć na jego nazwę (rys. 4-7), aby otworzyć widok listy terminów.

kod kursu	forma, nazwa kursu [liczba zajęć]	ECTS	operacja
TEST00001	laboratorium Test 0	1	
Pokazuj 10 rekordów, sortuj wg. kodów kursów ↑ ok			
			Skocz do 1 ok Strona 1 z 1

Rys. 4-7 Lista kursów – wybranie nowo dodanego kursu

Ponieważ nowo utworzony kurs nie ma utworzonych żadnych grup zajęciowych, wyświetlony zostanie komunikat o braku zajęć (rys. 4-8).

Do tego kursu nie są przypisane żadne zajęcia.

Aby dodać zajęcia, naciśnij przycisk **dodaj nowe zajęcia** poniżej.

dodaj nowe zajęcia

powrót

Rys. 4-8 Komunikat o braku zajęć

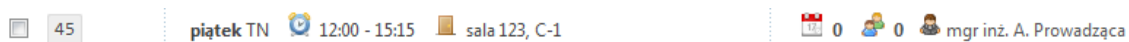
Nacisnąć przycisk *dodaj nowe zajęcia*, aby otworzyć formularz dodawania grupy zajęciowej (rys. 4-9).

dodaj zajęcia		
Numer ID zajęć <small>Wartość jest przydzielana automatycznie</small>	<input type="text" value="(przydzielany automatycznie)"/>	
Prowadzący <small>ID prowadzącego zajęcia</small>	<input type="text" value="pro"/> <div>mgr inż. Anna Prowadząca</div>	
Sala <small>Sala, w której odbywają się zajęcia</small>	<input type="text" value="123, C-1"/>	✓
Dzień tygodnia <small>Dzień, w którym odbywają się zajęcia</small>	<input type="text" value="Piątek"/>	✓
Tydzień <small>Wybierz, czy zajęcia mają odbywać się co tydzień, czy co dwa tygodnie</small>	<input type="radio"/> co tydzień <input checked="" type="radio"/> nieparzysty <input type="radio"/> parzysty	
Godzina rozpoczęcia <small>Wybierz godzinę rozpoczęcia zajęć (HH:MM)</small>	<input type="text" value="12"/> : <input type="text" value="00"/>	✓ ✓
Godzina zakończenia <small>Wpisz godzinę zakończenia zajęć (HH:MM)</small>	<input type="text" value="15"/> : <input type="text" value="15"/>	✓ ✓
Kod kursu <small>Kod kursu, do którego przypisane zostaną zajęcia</small>	<input type="text" value="TEST00001"/>	

zapisz

Rys. 4-9 Formularz dodawania grupy zajęciowej

Wpisane przykładowo dane zatwierdzić przyciskiem *zapisz*. Jeżeli akcja przebiegła poprawnie, powinien zostać wyświetlony komunikat o sukcesie operacji. Nacisnąć przycisk *powrót*, aby wrócić do widoku listy kursów. Na liście zajęć odnaleźć nowo utworzoną grupę (Rys. 4-10) i nacisnąć na nią kursorem myszy.



Rys. 4-10 Lista zajęć – nowa grupa zajęciowa

Otwarty zostanie widok terminów, jednak zasygnalizowany zostanie brak rekordów w bazie i zasugerowane będzie dodanie nowego terminu. Nacisnąć zakładkę *zapisani studenci*, aby uruchomić widok zapisów studentów. Wyświetlony zostanie jednak komunikat o braku studentów (rys. 4-11). Nacisnąć przycisk *zapisz studenta*.

Do tych zajęć nie zapisano jeszcze żadnego studenta.

Aby zapisać studentów do tych zajęć, naciśnij przycisk **zapisz studenta** poniżej.

zapisz studenta **powrót**

Rys. 4-11 Komunikat o braku zapisanych studentów

W nowo otwartym formularzu zapisu studenta wpisać w polu pierwsze cyfry numeru indeksu lub nazwisko studenta (rys. 4-12). Wybrać z listy sugestii żadaną pozycję. Modyfikację zatwierdzić przyciskiem *zapisz*.

modyfikacja terminu

Student

Wpisz imię lub nazwisko studenta w dużym polu, aby wyszukać i wybrać

(123456) Jan Student

zapisz

Rys. 4-12 Okno zapisu studenta na zajęcia

4.4 Otwarcie sali i sprawdzenie obecności

Uruchomić aplikację klienta sprawdzania obecności. Wpisać numer nowo utworzonej sali w polu ID sali w zakładce *ustawienia połączenia* (rys. 4-13), aby powiązać aplikację klienta obecności z salą. Skonfigurować pozostałe ważne parametry połączenia i czytnika.

Historia **Ustawienia połączenia** **Ustawienia czytnika**

adres IP serwera 192.168.137.1

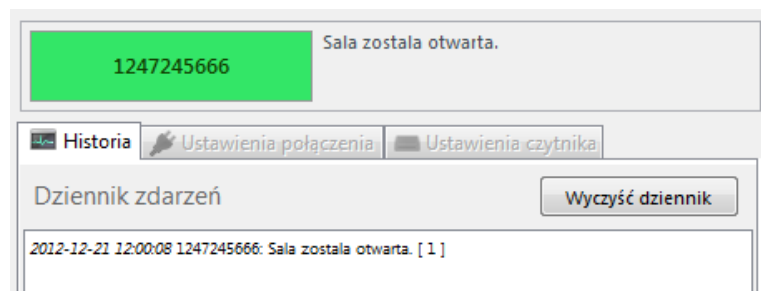
numer portu 2321

ID sali 15






Rys. 4-13 Powiązanie numeru ID sali z klientem obecności

Naciśnąć przycisk *Start*, aby rozpocząć działanie systemu sprawdzania obecności. Jeżeli system sprawdzania obecności uruchomił się poprawnie, poczekać na nadejście godziny i dnia tygodnia nowo utworzonych zajęć. Zbliżyć kartę prowadzącego, aby otworzyć salę. Jeżeli kartę zbliżono o odpowiedniej godzinie i w odpowiednim dniu

tygodnia, wyświetlony zostanie komunikat o otwarciu sali (rys. 4-14) i utworzony zostanie nowy termin w panelu zarządzania (rys. 4-15).





Rys. 4-14 Otwarcie sali za pomocą aplikacji klienta

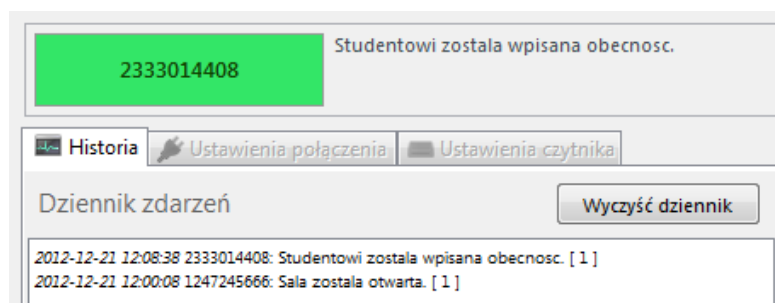
id	data zajęć	obecności	otwarcie sali	operacja
 74	21.12.2012	 0 / 1	 12:00:07	 
Pokazuj <input type="text" value="10"/> rekordów, sortuj wg. <input type="text" value="Data ↑"/> <input type="button" value="ok"/>				
Skocz do <input type="text" value="1"/> <input type="button" value="ok"/> Strona <u>1</u> z <u>1</u>				

Rys. 4-15 Nowy termin zajęć na liście terminów

Nacisnąć kursorem myszy na nowej dacie zajęć, aby otworzyć listę obecności z tego terminu (rys. 4-16). W tym samym czasie zbliżyć legitymację studenta zapisanego na zajęciu. Zbliżenie legitymacji spowoduje wpisanie obecności do systemu (rys. 4-17).

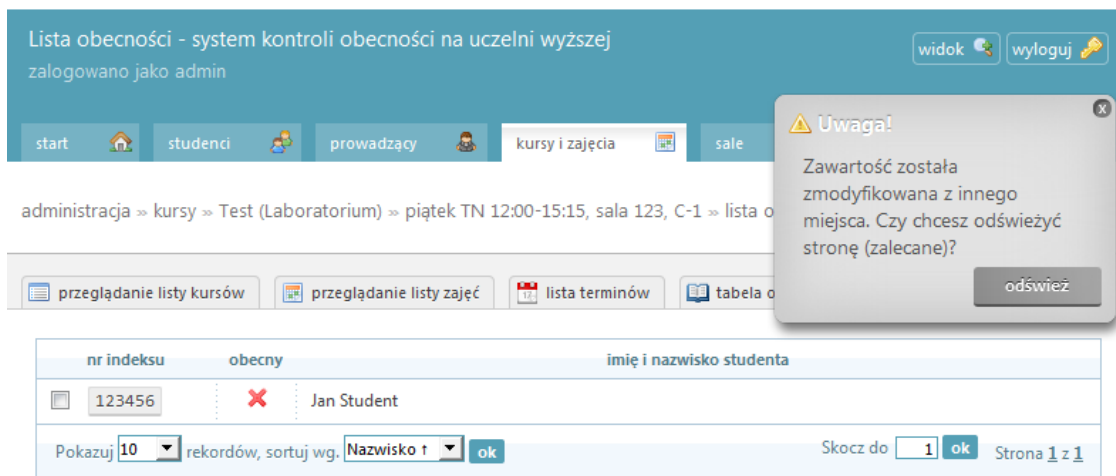
nr indeksu	obecny	imię i nazwisko studenta
 123456		Jan Student
Pokazuj <input type="text" value="10"/> rekordów, sortuj wg. <input type="text" value="Nazwisko ↑"/> <input type="button" value="ok"/>		
Skocz do <input type="text" value="1"/> <input type="button" value="ok"/> Strona <u>1</u> z <u>1</u>		

Rys. 4-16 Widok listy obecności przed wpisaniem obecności



Rys. 4-17 Wpisanie obecności studentowi za pomocą aplikacji klienta

Jeżeli jednak w przeglądarce pozostawiono otwarty widok listy obecności, wyświetlone zostanie powiadomienie o wprowadzeniu nowej obecności (rys. 4-18). Naciśnięcie przycisku *odśwież* zmodyfikuje wskaźnik obecności (rys. 4-19).



Rys. 4-18 Sygnalizacja wprowadzenia nowej obecności



Rys. 4-19 Odświeżony wskaźnik obecności – obecność wpisana

5 Instalacja i uruchomienie oprogramowania

Wszystkie instrukcje instalacji, konfiguracji i uruchomienia oprogramowania są przetestowane na platformie Windows, jednak w podobny sposób możliwa jest konfiguracja i instalacja tego samego oprogramowania na platformach Linux i Mac OS, jako że wszystkie sterowniki i biblioteki, a przede wszystkim środowisko uruchomieniowe aplikacji, są dostępne dla każdego systemu operacyjnego.

5.1 Instalacja i konfiguracja serwera

5.1.1 Uruchomienie komponentu MySQL

Do poprawnego działania aplikacji serwera wymagany jest aktywny moduł baz danych MySQL. Instrukcję instalacji i uruchomienia można znaleźć w rozdziale 5.3.1 Instalacja serwera Apache, PHP i MySQL.

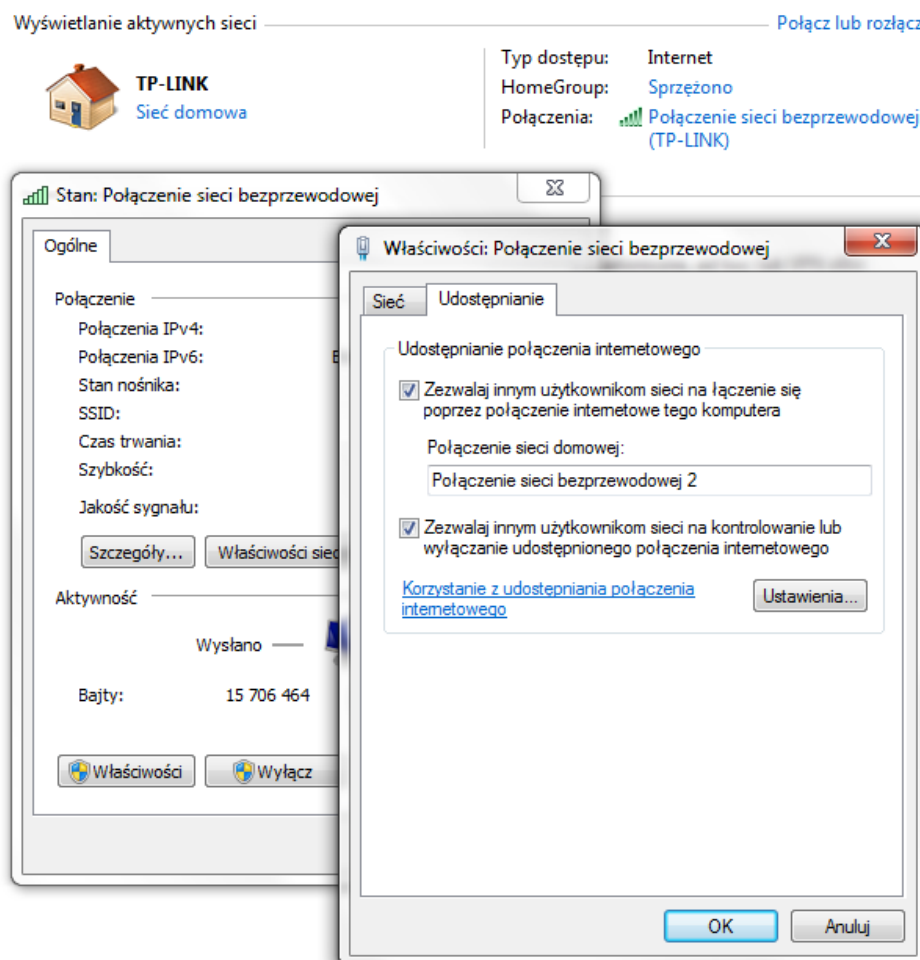
5.1.2 Uruchomienie wirtualnego routera

Komunikacja serwera z klientami (aplikacjami, które będą komunikowały się z serwerem) wymaga połączenia wszystkich maszyn, na której znajdują się aplikacje klienta, do jednej sieci. W realizacji tego z pomocą przychodzi funkcjonalność wirtualnego routera, którą łatwo można aktywować na platformie Windows. W tym celu należy uruchomić okno konsoli i wpisać polecenie:

```
netsh wlan set hostednetwork mode=allow ssid=KlientObecnosciSerwer  
key=<haslo>
```

Następnie tak utworzony wirtualny interfejs sieci bezprzewodowej należy powiązać z siecią, która ma dostęp do Internetu. W tym celu należy w otworzyć okno *Stan połączenia* z poziomu *Centrum sieci i udostępniania* poprzez kliknięcie na nazwę połączenia sieci bezprzewodowej w polu *Połączenia*. Następnie należy nacisnąć przycisk *Właściwości*, aby otworzyć okno *Właściwości połączenia*. W tym oknie należy otworzyć zakładkę *Udostępnianie*. Po wybraniu nowo utworzonego interfejsu połączenia bezprzewodowego (zazwyczaj jest to połączenie oznaczone numerem 2, jeżeli w komputerze istnieje fizycznie

jeden interfejs sieci bezprzewodowej)), aktywować opcje *Zezwalaj innym użytkownikom sieci na łączenie się poprzez połączenie internetowe tego komputera*.



Rys. 5-1 Konfiguracja udostępniania sieci w Centrum sieci i udostępniania

Po takiej konfiguracji sieci bezprzewodowej, wystarczy w konsoli wpisać polecenie `netsh wlan start hostednetwork`, aby uruchomić wirtualny router.

5.1.3 Uruchomienie aplikacji serwera

Serwer systemu kontroli obecności jest aplikacją napisaną w języku Java, wobec czego, aby ją uruchomić, na platformie musi być zainstalowana wirtualna maszyna Javy JRE (*Java Runtime Environment*) przynajmniej w wersji 1.6. Można ją pobrać ze strony internetowej <http://www.java.com>.

Po zainstalowaniu środowiska Java JRE, aplikację można uruchomić poprzez wpisanie w konsoli polecenia `java -jar server.jar "2048"` w katalogu, gdzie znajduje

się plik wykonywalny aplikacji serwera – server.jar. Czterocyfrowa liczba to numer portu, który przez aplikacja serwera będzie nasłuchiwany. W cudzysłowie można wpisać dowolny numer portu (pod warunkiem oczywiście, że nie jest on w użytku przez inne usługi – standardowo porty o numerze 1 – 1023 są przypisane odgórnie różnym protokołom, toteż każdy inny port o wartości większej teoretycznie może być bezpiecznie przypisany aplikacji serwera).

Wyłączenie okna konsoli spowoduje wyłączenie aplikacji serwera.

5.1.4 Program uruchamiający serwer

Poprawne wykonanie powyższych instrukcji umożliwi poprawne uruchomienie serwera. Aby jednak nie powtarzać ręcznego wykonywania operacji przy każdorazowym uruchomieniu komputera, można wykorzystać skrypt, który sam uruchamia wszystkie niezbędne usługi – wirtualny router, usługę MySQL i Javową aplikację serwera. Kod takiego programu wsadowego jest bardzo krótki:

```
@echo off
netsh wlan start hostednetwork
C:\xampp\xampp_start.exe
java -jar server.jar "2048"
```

Plik tekstowy z taką zawartością można zapisać jako plik o rozszerzeniu *.bat* i uruchomić jak program wykonywalny.

5.2 Instalacja i konfiguracja klienta

5.2.1 Uruchomienie aplikacji

Gdy na komputerze będzie zainstalowane środowisko uruchomieniowe Java, program w formacie *.jar można uruchomić na dwa sposoby:

- a) jak zwyczajną aplikację, z poziomu interfejsu okienkowego, poprzez dwukrotne kliknięcie lewym przyciskiem myszy na ikonie aplikacji,
- b) z poziomu konsoli, poleceniem `java -jar client.jar` wywołanym w katalogu, gdzie znajduje się plik wykonywalny aplikacji klienta – client.jar.

5.2.2 Podłączenie czytnika kart

Aby aplikacja klienta poprawnie komunikowała się z czytnikiem kart, na komputerze powinny być zainstalowane sterowniki czytnika kart, które można pobrać ze strony <http://www.hidglobal.com> po wcześniejszym podaniu wersji czytnika i systemu operacyjnego. W innym razie czytnik pozostanie nierozpoznanym urządzeniem USB.

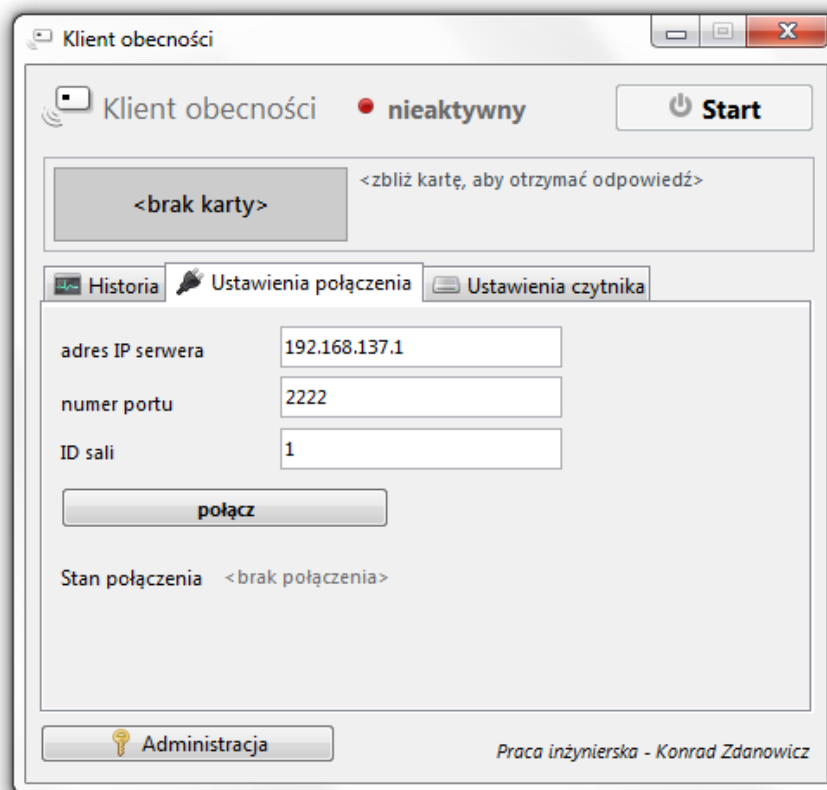
5.2.3 Połączenie z serwerem

Do właściwej komunikacji klienta z serwerem niezbędne jest połączenie się z odpowiednią siecią bezprzewodową. O ile poprawnie został uruchomiony serwer, powinna być dostępna sieć bezprzewodowa o nazwie *KlientObecnosciServer* (lub innej, podanej przez administratora przy zakładaniu wirtualnego routera na komputerze serwerowym). Wystarczy połączyć się z tą siecią, podając ustalone wcześniej hasło.

5.2.4 Konfiguracja klienta

Po podłączeniu czytnika kart, połączeniu do sieci bezprzewodowej i uruchomieniu aplikacji, należy skonfigurować połączenie i ustawienia czytnika.

5.2.5 Konfiguracja połączenia

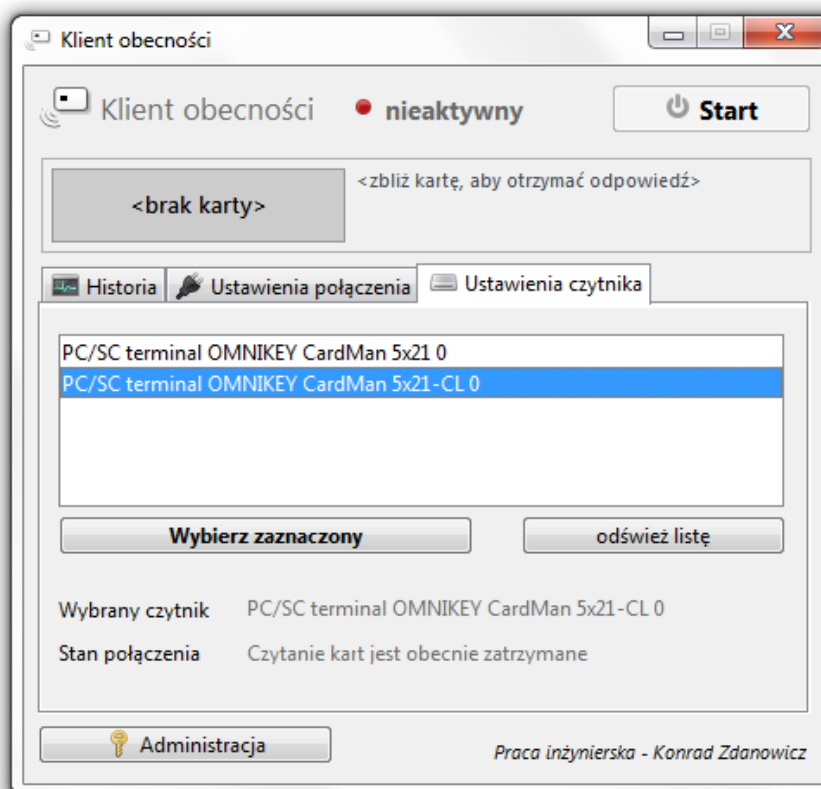


Rys. 5-2 Okno konfiguracji połączenia

Rys. 5-2 Okno konfiguracji połączenia włącza się po naciśnięciu zakładki *Ustawienia połączenia*. W polach *adres IP serwera* i *numer portu*, należy wpisać adres IP serwera i numer portu, który jest nasłuchiwany przez aplikację serwer. Następnie w polu *ID Sali* należy wpisać numer ID sali, w której ma być sprawdzana obecność. Po wpisaniu danych nacisnąć przycisk *Połącz*. Jeżeli połączenie zostanie nawiązane poprawnie, wyświetlony zostanie komunikat **Stan połączenia Połączenie z hostem 192.168.137.1:2222 poprawne.**, a następnie wybrać podłączony czytnik kart (typ – CL – bezstykowe). Jeżeli udało się nawiązać połączenie z hostem i połączyć z czytnikiem, zostanie odblokowany przycisk włączający działanie klienta.

5.2.5.1 Konfiguracja czytnika

Rys. 5-3 Okno ustawień czytnika włącza się po naciśnięciu zakładki *Ustawienia czytnika*.

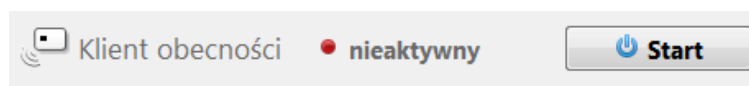


Rys. 5-3 Okno ustawień czytnika

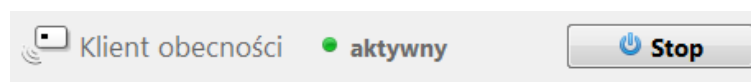
Jeżeli czytnik jest podłączony, naciśnięcie przycisku *Odśwież listę* spowoduje wyświetlenie listę terminali podpiętych czytników. Po wybraniu (zaznaczeniu na liście) odpowiedniego modelu czytnika kart bezstykowego, tutaj *PC/S.C. terminal OMNIKEY CardMan 5x21-CL 0*, należy nacisnąć przycisk *Wybierz zaznaczony*. Poprawne wybranie czytnika spowoduje wpisanie jego całej nazwy w polu *Wybrany czytnik*.

5.2.5.2 Uruchomienie czytania kart

Poprawna konfiguracja połączenia i czytnika odblokuje możliwość uruchomienia czytania kart. Rys. 5-4 i Rys. 5-5 przedstawiają wskaźniki czytania kart. Aby uruchomić czytanie kart, należy nacisnąć przycisk *Start*.



Rys. 5-4 Wskaźnik czytania kart nieaktywny



Rys. 5-5 Wskaźnik czytania kart aktywny

5.3 Instalacja i konfiguracja panelu zarządzania

Panel zarządzania obecnościami i innym informacjami jest aplikacją webową, napisaną w języku PHP i wykorzystuje system baz danych MySQL. Wobec tego, do poprawnej instalacji samego panelu zarządzania niezbędny jest aktywny serwer Apache oraz moduły PHP i MySQL.

5.3.1 Instalacja serwera Apache, PHP i MySQL

Instalację pakietów Apache, PHP i MySQL można wykonać na dwa sposoby. Popularnym i prostym rozwiązaniem jest pobranie paczki zawierającej wszystkie wymienione komponenty, która oprócz instalacji, udostępnia także przydatne narzędzia (np. PhpMyAdmin – graficzny interfejs do zarządzania bazami danych) a przede wszystkim prosty panel kontroli działania komponentów. Przykładami takich pakietów są programy XAMPP, WampSever czy Vertigo. W swojej pracy używam pierwszego z nich, który pobrać można ze strony internetowej <http://www.apachefriends.org>. Instalacja i uruchomienie są na tyle intuicyjne, że nie wymagają żadnego komentarza. Poniżej natomiast opiszę instalację manualną i konfigurację wszystkich niezbędnych komponentów.

Instalację serwera Apache można pobrać ze strony <http://httpd.apache.org>. Podczas instalacji pojawi się pytanie o nazwę domeny i serwera, gdzie wystarczy wpisać *localhost* (czyli domyślna domena serwera lokalnego).

Następnie, po ukończeniu instalowania i ponownym uruchomieniu komputera, można uruchomić program instalacyjny PHP dostępny na stronie <http://php.net>. Program poprosi o wybór rodzaju serwera lokalnego i jego lokalizację, gdzie należy zaznaczyć pierwszą opcję – *Apache 2.2* i wpisać adres katalogu, w którym Apache zainstalowano. Gdy instalator zakończy pracę, należy dokonać konfiguracji interpretera PHP. W tym celu należy odnaleźć plik *php.ini* w folderze, gdzie zainstalowano paczkę PHP. W tym pliku następnie odszukać liniijkę `extension=php_pdo_mysql.dll`, która odpowiedzialna jest za załączenie rozszerzenia PDO (PHP Data Objects). Jeżeli

przed tą linią znajduje się znak średnika, należy go usunąć, aby biblioteka, wykorzystywana do komunikacji z bazami danych, była poprawnie załączana.

Po ukończeniu instalacji i konfiguracji należy również poprawnie uruchomić komputer, aby dalej zainstalować paczkę MySQL pobraną ze strony internetowej <http://dev.mysql.com>. Gdy już zostanie zainstalowany komponent MySQL, w lokalizacji w folderze *bin* należy odnaleźć plik *winmysqladmin.exe* i go uruchomić, aby zainstalować MySQL w trybie *service*.

5.3.2 Instalacja i konfiguracja panelu zarządzania

Gdy wszystkie trzy składniki – serwer Apache, PHP i MySQL zostaną zainstalowane, należy je uruchomić zgodnie z kolejnością, w której były zainstalowane.

Następnie należy otworzyć folder *htdocs* w folderze instalacyjnym Apache i rozpakować do niego cały kod źródłowy i wszystkie pliki systemu obecności. Następnie w wypakowanym katalogu należy odszukać i otworzyć plik *admin/configs/database.php*. W tym pliku znajdują się definicje stałych odpowiedzialnych za połączenie systemu z bazą MySQL:

```
define('DB_HOST', 'localhost');  
define('DB_USER', 'root');  
define('DB_PASS', '');  
define('DB_NAME', 'systemobecnosci');  
define('DB_SALT', '');
```

O ile nie zmienialiśmy ustawień bazy danych, *DB_HOST* – nazwa hosta, *DB_USER* – użytkownika i *DB_PASS* – hasło powinny być pozostawione bez zmian. Pod stałą *DB_NAME* kryje się nazwa bazy danych, która będzie używana. Można tu wpisać dowolną ciąg znaków. *DB_SALT* zaś to unikalny ciąg znaków, którym zasalone jest hasło.

Przed uruchomieniem należy jeszcze dodać bazę danych *systemobecnosci*. W tym celu należy uruchomić konsolę i otworzyć katalog *bin* w folderze, gdzie zainstalowano komponent MySQL. Następnie należy uruchomić program konsolowy poleceniem *mysql.exe -u root -p*. System zapyta użytkownika o hasło (takie jak wartość stałej *DB_PASS*, domyślnie puste). Po pomyślnej autoryzacji ukaże się ekran powitalny. Teraz należy kolejno utworzyć bazę danych poleceniem **CREATE DATABASE systemobecnosci;**,

wybrać tę bazę do wykonania nań operacji: **USE systemobecnosci;** i w końcu zaimportować strukturę bazy danych systemu obecności poleceniem **SOURCE systemobecnosci.sql;**, gdzie nazwa pliku podana w parametrze polecenia SOURCE to ścieżka do pliku .sql zawierającego ciąg poleceń tworzących strukturę bazy.

Jeżeli baza została utworzona poprawnie, możemy także dodać konto administratora. W tym celu, bez wyłączania konsoli, należy wpisać komendę:

```
INSERT INTO userzy(UserLogin,PassHash) VALUES('login','hashHasla');
```

Przy czym wartość *hashHasla* powinna być skrótem hasła wygenerowanym algorytmem MD5. Do wygenerowania tego skrótu można posłużyć aplikacja *md5encoder.php* w katalogu *addons*, który znajduje się w katalogu głównym systemu obecności.

Tak skonfigurowany system obecności z bazą danych jest już gotowy do użytku. Aby go uruchomić, w pasku adresu przeglądarki należy wpisać adres <http://localhost/systemobecnosci/> i zalogować się wcześniej wprowadzonymi danymi użytkownika.

6 Cytowane prace

- [1] „The success of MIFARE™,” NXP Conductors, [Online]. Available: <http://www.mifare.net>. [Data uzyskania dostępu: 11 Grudzień 2012].
- [2] N. Semiconductors, „MIFARE™ Milestones,” Syria, 2004.
- [3] *ISO/IEC 14443-3 - Type A Identification Cards - Contactless Integrated circuit(s) cards - Proximity Cards- Part 3: Initialisation and Anticollision*, 1999.
- [4] „MIFARE™ Classic 1K,” MIFARE, [Online]. Available: <http://mifare.net/products/mifare-smartcard-ic-s/mifare-1k/>. [Data uzyskania dostępu: 11 Grudzień 2012].
- [5] *ISO 7816-4 - Interindustry Commands for Interchange*, 1990.
- [6] P. Henry, „Communications Magazine,” *IEEE*, pp. 66-72, Grudzień 2002.
- [7] „Wi-Fi bez routera,” *PC Format*, Luty 2010.
- [8] J. Gosling, „Preface,” w *The Java™ Language Specification Second Edition*, Sun Microsystems, 2000.
- [9] „Android SDK,” [Online]. Available: <http://developer.android.com/sdk/index.html>.
- [10] „Java™ Platform, Standard Edition 6 API Specification,” Oracle, 1993. [Online]. Available: <http://docs.oracle.com/javase/6/docs/api/>. [Data uzyskania dostępu: 2011].
- [11] M. Loy, „Introducing Swing,” w *Java Swing*, O'Reilly Media, 2012.
- [12] W3C, „HTML 4.01 Specification - Introduction to HTML 4,” 24 Grudzień 1999. [Online]. Available: <http://www.w3.org/TR/REC-html40/>.
- [13] M. Pilgrim, „Detecting HTML5 Features,” w *HTML5: Up and Running*, O'Reilly Media,

2010.

- [14] J. J. Garrett, „Ajax: A New Approach to Web Applications,” 18 Luty 2005. [Online]. Available: <http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications>.
- [15] C. Mills, „1: Introduction to CSS 3 and modern web design,” w *Practical CSS3: Develop and Design*, Peachpit Press, 2012.
- [16] E. Meyer, w *Cascading style sheets: The definitive guide*, O'Reilly Media, 2004.
- [17] C. Lindley, w *jQuery Cookbook*, O'Reilly Media, 2010.
- [18] L. Argerich, Professional PHP 4 programming, Computing Reviews 46.1, 2005.
- [19] „Usage Stats for April 2007,” [Online]. Available: <http://pl.php.net/usage.php>. [Data uzyskania dostępu: 2007].
- [20] „Introduction: What can PHP do?,” [Online]. Available: <http://php.net/manual/pl/intro-whatcando.php>. [Data uzyskania dostępu: 5 Marzec 2009].
- [21] „PEAR - PHP Extension and Application Repository,” [Online]. Available: <http://pear.php.net>.
- [22] „Basic contepts all PEAR users should understand,” [Online]. Available: <http://pear.php.net/manual/en/guide.users.concepts.php>.
- [23] M. Widenius i D. Axmark, „1.2 What is MySQL?,” w *MySQL Reference Manual*, O'Reilly Media, Inc., 2002.

7 Spis ilustracji

Rys. 1-1 Schemat systemu kontroli obecności	5
Rys. 3-1 Okno główne aplikacji serwera	14
Rys. 3-2 Okno główne aplikacji klienta	15
Rys. 3-3 Wskaźniki stanu sprawdzania obecności	16
Rys. 3-4 Przyciski uruchamiania i zatrzymania sprawdzania obecności	16
Rys. 3-5 Panel odczytanego numeru UID i odpowiedzi serwera	17
Rys. 3-6 Zakładki nawigacji pomiędzy oknami	17
Rys. 3-7 Panel historii zdarzeń.....	18
Rys. 3-8 Panel ustawień połączenia	18
Rys. 3-9 Panel ustawień czytnika	19
Rys. 3-10 Przycisk otwierający panel zarządzania obecnościami.....	19
Rys. 3-11 Standardowa ikona okna głównego aplikacji klienta.....	19
Rys. 3-12 Ikona klienta w obszarze powiadomień.....	20
Rys. 3-13 Menu kontekstowe ikony klienta obecności w obszarze powiadomień.....	20
Rys. 3-14 Powiadomienia – informacja	21
Rys. 3-15 Powiadomienia – ostrzeżenie	21
Rys. 3-16 Powiadomienia – błąd	21
Rys. 3-17 Okno logowania do panelu zarządzania	22
Rys. 3-18 Okno główne panelu zarządzania.....	23
Rys. 3-19 Tytuł strony panelu zarządzania	23
Rys. 3-20 Przełącznik widoku i przycisk wylogowania panelu zarządzania	24
Rys. 3-21 Nawigacja główna panelu zarządzania.....	24
Rys. 3-22 Lokalizacja w panelu zarządzania	24
Rys. 3-23 Okno szybkiego wyboru na stronie głównej.....	24
Rys. 3-24 Okno szybkiego wyboru na stronie głównej z rozwiniętym menu „kursy”	25
Rys. 3-25 Widok listy studentów	25
Rys. 3-26 Nawigacja widoku	26
Rys. 3-27 Formularz wyszukiwania studentów	26
Rys. 3-28 Usuwanie zaznaczonych rekordów	26
Rys. 3-29 Tabela rekordów.....	26
Rys. 3-30 Kolumna identyfikatora	27

Rys. 3-31 Kolumna wyboru operacji	27
Rys. 3-32 Pasek sortowania i nawigacji między stronami	27
Rys. 3-33 Widok edycji studenta.....	28
Rys. 3-34 Asynchroniczne sprawdzanie unikalności wpisywanych danych	29
Rys. 3-35 Widok listy sal.....	29
Rys. 3-36 Widok listy kursów.....	30
Rys. 3-37 Widok edycji kursu.....	30
Rys. 3-38 Widok listy grup zajęciowych	31
Rys. 3-39 Widok modyfikacji grupy zajęciowej.....	32
Rys. 3-40 Widok listy terminów	33
Rys. 3-41 Widok modyfikacji terminu	33
Rys. 3-42 Widok listy obecności.....	34
Rys. 3-43 Widok tabeli obecności	34
Rys. 3-44 Okno wyboru typu obecności	35
Rys. 3-45 Ostrzeżenie o aktualizacji danych.....	35
Rys. 3-46 Widok listy zapisanych studentów	36
Rys. 3-47 Widok okna zapisów.....	36
Rys. 4-1 Menu szybkiego wyboru – dodaj studenta	37
Rys. 4-2 Formularz dodawania studenta	38
Rys. 4-3 Powodzenie dodania studenta.....	38
Rys. 4-4 Formularz dodawania prowadzącego.....	39
Rys. 4-5 Formularz dodawania sali	39
Rys. 4-6 Formularz dodawania kursu	40
Rys. 4-7 Lista kursów – wybranie nowo dodanego kursu	40
Rys. 4-8 Komunikat o braku zajęć	40
Rys. 4-9 Formularz dodawania grupy zajęciowej.....	41
Rys. 4-10 Lista zajęć – nowa grupa zajęciowa	41
Rys. 4-11 Komunikat o braku zapisanych studentów	42
Rys. 4-12 Okno zapisu studenta na zajęcia	42
Rys. 4-13 Powiązanie numeru ID sali z klientem obecności.....	42
Rys. 4-14 Otwarcie sali za pomocą aplikacji klienta	43
Rys. 4-15 Nowy termin zajęć na liście terminów	43
Rys. 4-16 Widok listy obecności przed wpisaniem obecności	43
Rys. 4-17 Wpisanie obecności studentowi za pomocą aplikacji klienta.....	43

Rys. 4-18 Sygnalizacja wprowadzenia nowej obecności	44
Rys. 4-19 Odświeżony wskaźnik obecności – obecność wpisana	44
Rys. 5-1 Konfiguracja udostępniania sieci w Centrum sieci i udostępniania	46
Rys. 5-2 Okno konfiguracji połączenia	49
Rys. 5-3 Okno ustawień czytnika	50
Rys. 5-4 Wskaźnik czytania kart nieaktywny	50
Rys. 5-5 Wskaźnik czytania kart aktywny	51
Rys. 6-1 Schemat blokowy przetworzenia zapytania przez serwer	60
Rys. 6-2 Schemat UML klas programu serwera	60
Rys. 6-3 Schemat bazy danych MySQL	61
Rys. 6-4 Diagram UML klas programu klienta	62
Rys. 6-5 Schemat ideowy wzorca MVC	63

8 Dodatek - Kod źródłowy rozwiązania

8.1 Serwer

8.1.1 Opis architektury programu

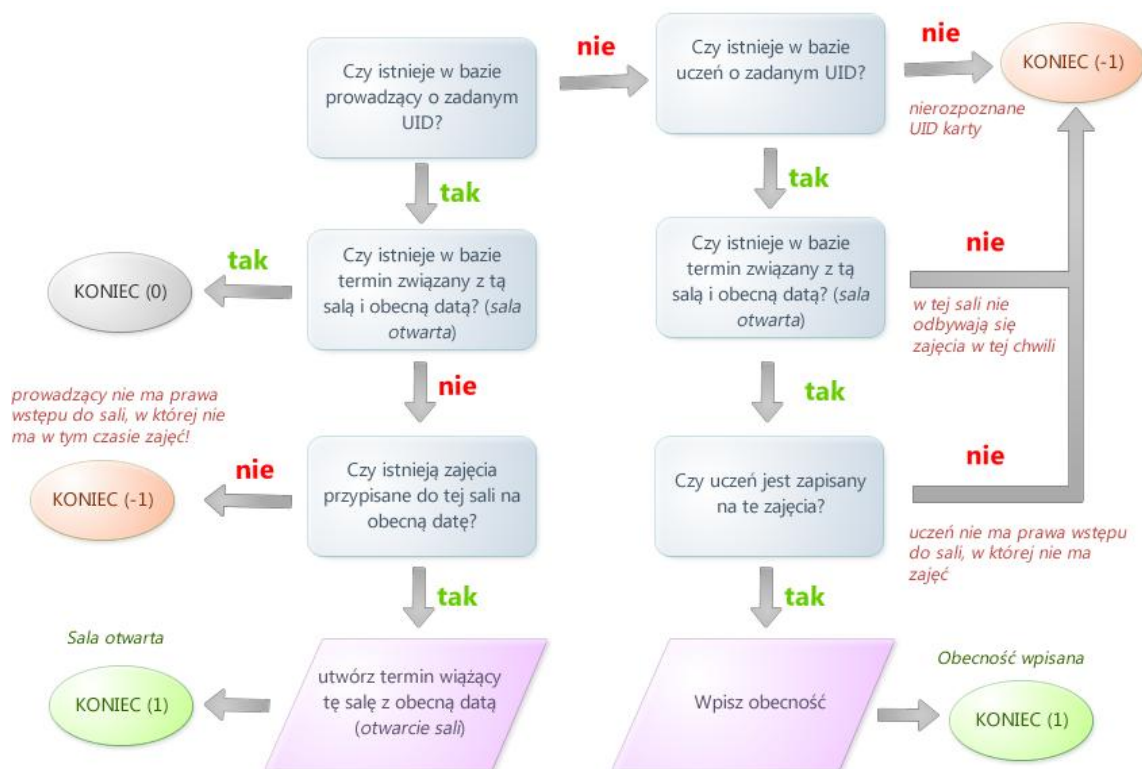
Aplikacja serwera jest zaprojektowana tak, aby zapewnić jak najmniejszy czas oczekiwania żądania na odpowiedź. Kod główny, czyli klasa *Server* najpierw deklaruje tablicę wątków o maksymalnej liczbie połączeń obsługiwanych równej 200, a następnie w nieskończonej pętli nasłuchuje przychodzących połączeń i, o ile istnieje wolny wątek do obsługi klienta, tworzy na nim nowy obiekt klasy *ClientThread* obsługujący klienta.

Do konstruktora obiektu obsługującego klienta dostarczony jest ciąg tekstowy, w którym zawarte jest zapytanie w następującym formacie uid <uid> sala <id sali>.

Jest ono odpowiednio przetworzone tak, aby wyluskać parametry – ID sali i UID karty. Następnie program określa istotne parametry, takie jak np. obecną datę i godzinę, maksymalny czas spóźnienia, w ramach którego studentowi przyznana jest jeszcze obecność, i maksymalny czas spóźnienia otwarcia sali, po którym system blokuje możliwość otwarcia sali.

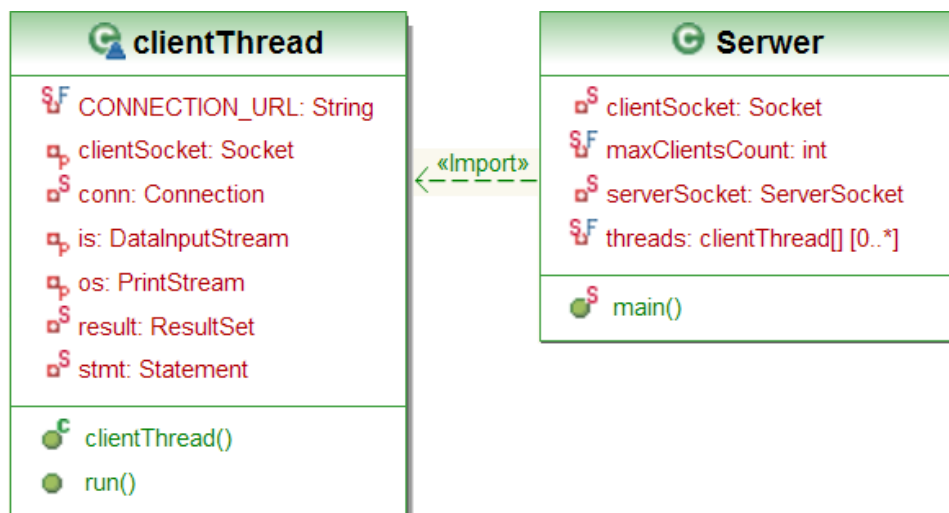
Po uzyskaniu numeru ID sali i UID karty, system łączy się z bazą danych i wykonuje serię zapytań w celu identyfikacji, czy zbliżono kartę studenta, czy prowadzącego, a także czy w tym czasie w sali o podanym numerze ID odbywają się zajęcia i czy można wpisać obecność lub otworzyć salę.

Po przejściu przez schemat decyzyjny (rys. 8-1) program konstruuje odpowiedź w postaci: <tekst odpowiedzi> [<kod odpowiedzi>], gdzie kod odpowiedzi może być równy -1, 0 lub 1, co oznacza kolejno: niepowodzenie/błąd, brak zmian, powodzenie operacji.



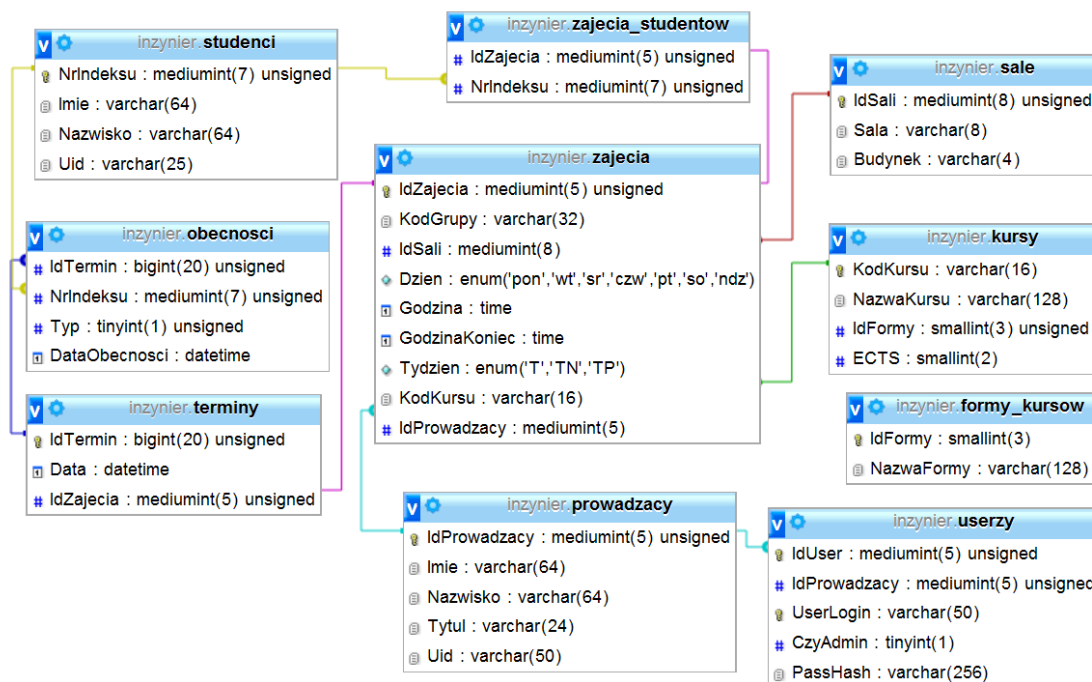
Rys. 8-1 Schemat blokowy przetworzenia zapytania przez server

8.1.2 Diagram UML klas programu serwera



Rys. 8-2 Schemat UML klas programu serwera

8.1.3 Schemat bazy danych MySQL



Rys. 8-3 Schemat bazy danych MySQL

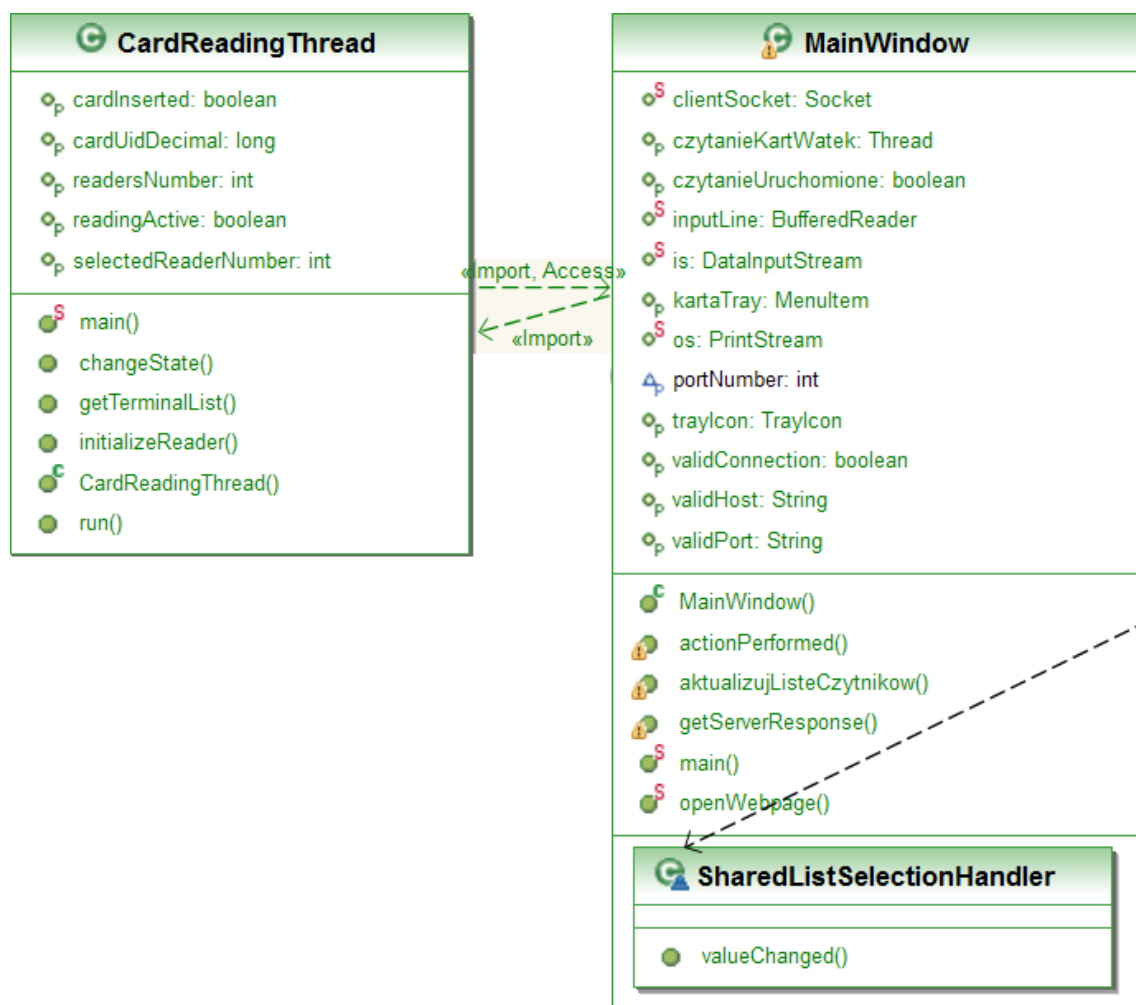
8.2 Klient

8.2.1 Opis architektury programu

Aplikacja klienta jest programem okienkowym napisanym w języku Java, przy użyciu biblioteki Swing do tworzenia interfejsów okienkowych. Składa się z dwóch klas – *MainWindow* i *CzytnikWatek*. Pierwsza klasa definiuje interfejs okna i obsługuje wszystkie zdarzenia naciśnięcia przycisków i modyfikacji pól. Ponadto w polach przechowuje np. uchwyt do połączenia z serwerem, listę obecnie podłączonych czytników i przede wszystkim obiekt *CzytnikWatek*. Ten z kolei jest klasą implementującą interfejs *Runnable*, dzięki czemu możliwe jest zrealizowanie za pomocą tego obiektu obsługę zbliżenia karty do czytnika.

Gdy użytkownik naciśnie przycisk *Start*, rozpoczynający system sprawdzania obecności, wcześniej utworzony obiekt *CzytnikWatek* dostaje sygnał, aby rozpocząć działanie i uruchamia metodę *run()*, która jest nieskończoną pętlą próbującą zarejestrować zbliżenie karty do czytnika.

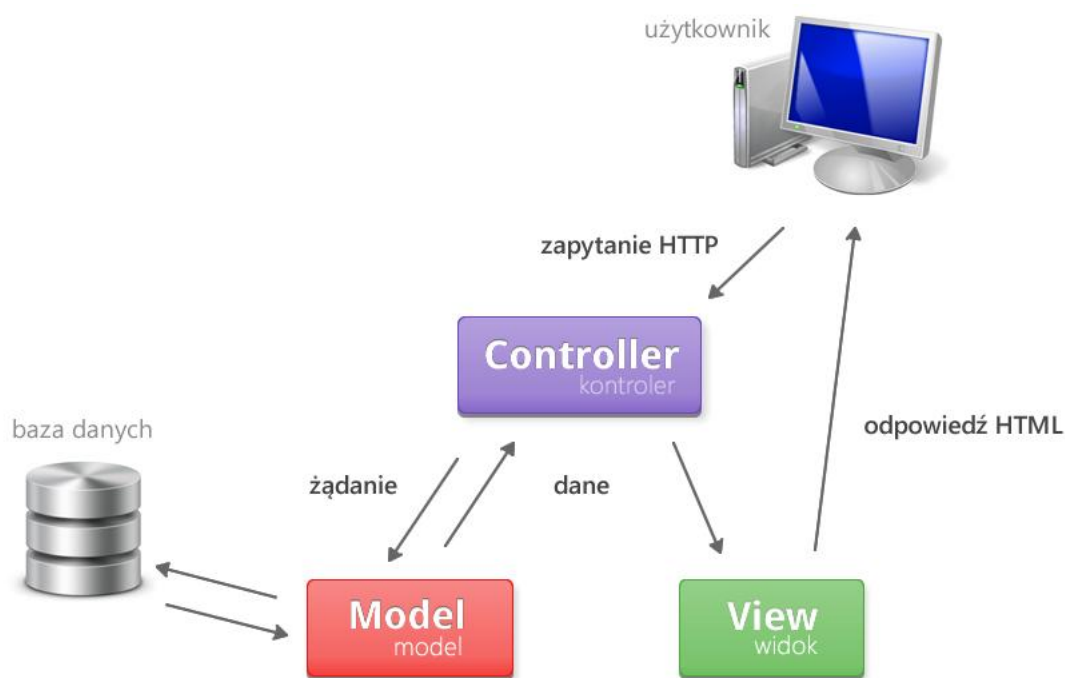
8.2.2 Diagram UML klas programu klienta



Rys. 8-4 Diagram UML klas programu klienta

8.3 Panel zarządzania

Panel zarządzania to aplikacja internetowa w całości napisana w języku PHP, komunikująca się z bazą danych MySQL. Architekturą przypomina standardowy wzorzec projektowy MVC (Model–View–Controller) – model, widok i kontroler.



Rys. 8-5 Schemat ideowy wzorca MVC

Rys. 8-5 przedstawia ideowy opis wzorca projektowego MVC. Głównym założeniem wzorca MVC jest oddzielenie warstwy danych (model) od warstwy prezentacji (widok) i kontrolowanie ich za pomocą osobnego, niezależnego modułu.

Kontroler, czyli centrum dowodzenia/szkielet aplikacji, przeznaczony jest do przetwarzania zapytań HTTP od użytkownika i odpowiednie pokierowanie modelem i widokiem, aby najpierw wydobyć potrzebne dane z bazy, a następnie je zaprezentować pożądanym przez użytkownika widoku.

Model odpowiada wyłącznie za komunikację ze źródłem danych (niekoniecznie musi być to baza danych) i przekazywanie żądanych porcji danych do kontrolera. Warstwa widoku dba o format, strukturę i wygląd dokumentu, który wróci do użytkownika jako odpowiedź.

8.3.1 Model

Rolę modelu pełni klasa *Admin* w pliku *admin.functions.php*. Zawiera ona definicję metod odpowiadających za przetwarzanie i wysyłanie zapytań do bazy danych, a następnie zwracanie zinterpretowanej odpowiedzi lub zestawu poszukiwanych danych.

Klasa *Admin* zawiera jedno pole, którym jest uchwyt do obiektu PDO. Jedyny, parametryczny konstruktor klasy *Admin* wymusza, aby uchwyt do utworzonego obiektu został od razu podany. Obiekt PDO natomiast jest tworzony w pliku *database.php*, który oprócz definicji stałych, tworzy także obiekt PDO i łączy go z bazą MySQL. To właśnie uchwyt do tego konkretnego obiektu jest przekazywany do konstruktora w części kodu kontrolera.

Metody klasy *Admin* przyjmują za parametr zwykle klucze główne tabel, na których chcemy operować, a w przypadku metod dodających dane, argumentem jest cały wiersz danych. Wartości zwracane także mogą być różnego typu. Funkcje usuwające i dodające rekord zwracają wartość boolowską, która oznacza poprawność wykonanej operacji. Innym razem metoda może zwrócić cały wiersz, którego klucz podany został w argumencie funkcji. Poniższy wykaz zawiera opis wszystkich metod, ich argumentów i zwracanych wartości.

Tabela 1 Metody klasy *Admin*

Action_TerminIstnieje(\$id)

Funkcja zwraca `true`, jeżeli termin o podanym numerze id istnieje.

Action_DodajTermin(\$d)

Funkcja dodaje termin. Jeżeli operacja przebiegła poprawnie, zwracana jest wartość `true`.

Action_UsunTermin(\$id)

Funkcja usuwa termin o podanym numerze id. Jeżeli usuwanie przebiegło poprawnie, zwracana jest wartość `true`. Metoda usuwa także wszystkie obecności z tego terminu.

Action_AktualizujTermin(\$d)

Funkcja aktualizuje termin. W argumencie podany jest wiersz rekordu, gdzie zawarty jest numer id aktualizowanego terminu i nowe wartości kolumn.

Action_PojedynczyTermin(\$id)

Funkcja zwraca rekord o podanym numerze id. Jeżeli nie odnaleziono rekordu, zwracana jest wartość `false`.

Action_ZajeciaIstnieje(\$id)

Funkcja zwraca `true`, jeżeli wiersz o podanym numerze id istnieje.

Action_DodajZajecia(\$d)

Funkcja dodaje wiersz. Jeżeli operacja przebiegła poprawnie, zwracana jest wartość true.

Action_UsunZajecia(\$id)

Funkcja usuwa wiersz o podanym numerze id. Jeżeli usuwanie przebiegło poprawnie, zwracana jest wartość true. Oprócz usunięcia wiersza zajęć, usuwane są także powiązane terminy i obecności do tych zajęć.

Action_AktualizujZajecia(\$d)

Funkcja aktualizuje pojedynczy rekord. W argumencie podany jest wiersz rekordu, gdzie zawarty jest numer id aktualizowanego rekordu i nowe wartości kolumn.

Action_PojedynczeZajecia(\$id)

Funkcja zwraca rekord o podanym numerze id. Jeżeli nie odnaleziono rekordu, zwracana jest wartość false.

Action_UsunObecnosc(\$NrIndeksu,\$IdTermin)

Funkcja usuwa obecność studenta o zadanym numerze indeksu z zajęć o podanym numerze id.

Action_DodajObecnosc(\$NrIndeksu,\$IdTermin,\$Typ)

Funkcja wpisuje studentowi o zadanym numerze indeksu obecność o zadanym typie na wybranych zajęć.

Action_SprawdzObecnosc(\$NrIndeksu,\$IdTermin)

Funkcja zwraca typ obecności studenta o danym numerze indeksu na zajęciach o podanym numerze id. Brak obecności jest sygnalizowany zwracaną wartością „0”.

Action_KursIstnieje(\$id)

Funkcja zwraca true, jeżeli wiersz o podanym numerze id istnieje.

Action_DodajKurs(\$d)

Funkcja dodaje wiersz. Jeżeli operacja przebiegła poprawnie, zwracana jest wartość true.

Action_UsunKurs(\$id)

Funkcja usuwa wiersz o podanym numerze id. Jeżeli usuwanie przebiegło poprawnie, zwracana jest wartość true. Wraz z kursem usuwane są także wszystkie zajęcia, terminy z tym kursem związane.

Action_AktualizujKurs(\$d)

Funkcja aktualizuje pojedynczy rekord. W argumencie podany jest wiersz rekordu, gdzie zawarty jest numer id aktualizowanego rekordu i nowe wartości kolumn.

Action_StudentIstnieje(\$id)

Funkcja zwraca true, jeżeli wiersz o podanym numerze id istnieje.

Action_StudentIstnieje(\$id,\$uid)

Funkcja zwraca true, jeżeli student o podanym numerze indeksu i numerze UID legitymacji istnieje.

Action_DodajStudenta(\$d)

Funkcja dodaje studenta. Jeżeli operacja przebiegła poprawnie, zwracana jest wartość true. Jeżeli użytkownik próbuje dodać studenta o numerze indeksu, który już znajduje się w bazie, zwracana jest wartość false.

Action_UsunStudenta(\$id)

Funkcja usuwa wiersz o podanym numerze id. Jeżeli usuwanie przebiegło poprawnie, zwracana jest wartość true. Wraz z danymi o studencie usuwane są także wszystkie informacje o jego obecnościach i zapisach.

Action_AktualizujStudenta(\$d)

Funkcja aktualizuje pojedynczy rekord. W argumencie podany jest wiersz rekordu, gdzie zawarty jest numer id aktualizowanego rekordu i nowe wartości kolumn.

Action_PojedynczyStudent(\$id)

Funkcja zwraca pojedynczy rekord o podanym numerze id. Jeżeli nie odnaleziono rekordu, zwracana jest wartość false.

Action_PojedynczyStudentPoUid(\$id)

Funkcja zwraca pojedynczy rekord o podanym numerze UID. Jeżeli nie odnaleziono rekordu, zwracana jest wartość false.

Action_PojedynczyStudentPoNazwiskuLubImieniu(\$q)

Funkcja szuka studenta, którego nazwisko lub imię zawierają w sobie ciąg znaków \$q. Znalezione wiersze są zwracane w postaci tabeli asocjacyjnej, gdzie kluczem jest numer indeksu studenta, a wartością jest jego imię i nazwisko.

Action_ProwadzacyIstnieje(\$id)

Funkcja zwraca true, jeżeli wiersz o podanym numerze id istnieje.

Action_ProwadzacyIstnieje(\$id,\$uid)

Funkcja zwraca true, jeżeli prowadzący o podanym numerze id i uid karty istnieje.

Action_DodajProwadzacego(\$d)

Funkcja dodaje prowadzącego. Jeżeli operacja przebiegła poprawnie, zwracana jest wartość true.

Action_UsunProwadzacego(\$id)

Funkcja usuwa wiersz o podanym numerze id. Jeżeli usuwanie przebiegło poprawnie, zwracana jest wartość true.

Action_AktualizujProwadzacego(\$d)

Funkcja aktualizuje pojedynczy rekord. W argumencie podany jest wiersz rekordu, gdzie zawarty jest numer id aktualizowanego rekordu i nowe wartości kolumn.

Action_PojedynczyProwadzacy(\$id)

Funkcja zwraca pojedynczy rekord o podanym numerze id. Jeżeli nie odnaleziono rekordu, zwracana jest wartość false.

Action_PojedynczyProwadzacyPoUid(\$id)

Funkcja zwraca pojedynczy rekord o podanym numerze UID. Jeżeli nie odnaleziono rekordu, zwracana jest wartość false.

Action_PojedynczyProwadzacyPoNazwiskuLubImieniu(\$q)

Funkcja szuka prowadzącego, którego nazwisko lub imię zawierają w sobie ciąg znaków \$q. Znalezione wiersze są zwracane w postaci tabeli asocjacyjnej, gdzie kluczem jest numer id prowadzącego, a wartością jest jego tytuł, imię i nazwisko.

Action_StudentZapisany(\$NrIndeksu,\$IdZajecia)

Funkcja zwraca wartość true, jeżeli na zajęcia o podanym numerze id jest zapisany student o podanym numerze indeksu.

Action_ZapiszStudenta(\$NrIndeksu,\$IdZajecia)

Funkcja zapisuje studenta o podanym numerze indeksu na zajęcia o podanym id.

Action_WypiszStudenta(\$NrIndeksu,\$IdZajecia)

Funkcja wypisuje studenta o podanym numerze indeksu z zajęć o podanym id.

Action_SalaIstnieje(\$id)

Funkcja zwraca true, jeżeli termin o podanym numerze id istnieje.

Action_DodajSale(\$d)

Funkcja dodaje rekord. Jeżeli operacja przebiegła poprawnie, zwracana jest wartość true.

Action_UsunSale(\$id)

Funkcja usuwa salę o podanym numerze id. Jeżeli usuwanie przebiegło poprawnie, zwracana jest wartość true. Metoda usuwa także wszystkie obecności z tego terminu.

Action_AktualizujSale(\$d)

Funkcja aktualizuje rekord. W argumencie podany jest wiersz rekordu, gdzie zawarty jest numer id aktualizowanego terminu i nowe wartości kolumn.

Action_PojedynczaSala(\$id)

Funkcja zwraca rekord o podanym numerze id. Jeżeli nie odnaleziono rekordu, zwracana jest wartość false.

Action_TerminyStempel(\$d)

Funkcja za argument przyjmuje numer id terminu lub wektor numerów id terminów i zwraca datę obecności, która została wpisana w tym terminie najpóźniej.

8.3.2 Widok

Widok – w przypadku systemu obecności – nieco odbiega od podstawowej definicji widoku z wzorca MVC. Warstwa widoku zlewa się nieco z warstwą modelu i kontrolera, bowiem w plikach odpowiadających za poszczególne widoki następuje np. przetworzenie żądania zmiany sposobu wyświetlania czy sortowania rekordów, a także bezpośrednie zapytanie do bazy danych o wyświetlenie listy rekordów z uwzględnieniem wyszukiwania/filtrowania wyników. Widoki systemu obecności także częściowo przejmują funkcję kontrolera przy przetwarzaniu parametrów \$_GET i \$_POST w przypadku próby dodawania, modyfikacji czy usuwania rekordów, bowiem sam reaguje na akcję użytkownika i realizuje sprzężenie zwrotne, czyli interakcje z użytkownikiem.

Do widoku dostarczany jest zazwyczaj zestaw danych pobranych z bazy, a widok sam już dba o ułożenie rekordów w tabelę i udostępnienie użytkownikowi interfejsu, którym może manipulować treścią. Wszystkie widoki znajdują się w katalogu *admin/*.

W widokach, o ile warstwa modelu i widoku nie jest zupełnie odseparowana, o tyle warstwa treści i prezentacji są zupełnie oddzielone. Kod HTML generowany przez widok jest semantyczny, przez co za pomocą pliku *main.css* w katalogu *admin/css/* określona jest prezentacja wszystkich widoków systemu. Warstwa prezentacji widoku może zostać bez ingerencji w warstwę treści, czyli kod widoków. Umożliwia to stworzenie kilku różnych sposobów prezentacji widoków, np. dla tabletów, urządzeń mobilnych, różnych systemów operacyjnych i rozdzielczości dla dokładnie tego samego wygenerowanego kodu HTML.

8.3.3 Kontroler

Rolę kontrolera pełni w pierwszej fazie program napisany w pliku *admin.php*, gdzie rozpoczyna się cykl życia aplikacji. Najpierw załadowane są pliki nagłówkowe i konfiguracyjne do bazy danych. Dalej inicjalizowany jest model, czyli łączy z bazą danych. W dalszej kolejności uruchomiony zostaje mechanizm sesji, na którym opiera się autoryzacja dostępu do systemu. Jeżeli do komputera użytkownika, który obecnie próbuje się dostać do systemu, jest przypisana sesja z flagą „zalogowany” ustawioną na true, zostaje wydane zezwolenie na wyświetlenie zawartości interfejsu. W innym razie wybrany zostaje widok formularza autoryzacji.

Jeżeli klient wpisze poprawne dane przy logowaniu i zmieści się w dopuszczalnej liczbie prób, flaga zalogowania zostaje ustawiona na wartość true i uruchamiany jest skrypt *main.php*, który w dalszej kolejności przejmuje rolę kontrolera.

Za pomocą funkcji `switch` zostaje przetworzony parametr `$_GET['p']`, który niesie nazwę żądanego widoku. Jeżeli użytkownik próbuje dostać się do widoku, który nie istnieje, zostaje przekierowany na stronę główną. W innym razie określona zostaje ścieżka do widoku i zostaje on wyświetlony. Pozostałe parametry są przetwarzane już przez widoki.

Do systemu obecności udostępnione jest także *API*, za pomocą którego możemy z poziomu JavaScriptu, bądź jakichkolwiek innych źródeł zapytać `$_GET`, `$_POST` lub `$_REQUEST` utrzymać interakcję z bazą danych. Aby skorzystać z *API*, wymagana jest autoryzacja użytkownika poprzez panel logowania na stronie głównej aplikacji. Następnie,

aby wykonać zapytanie, należy wywołać plik *admin.api.php* z określonym parametrem *action* i dodatkowymi niezbędnymi parametrami w formacie *admin.api.php?action=<nazwa akcji>&<nazwa parametru>=<wartość>*.

Szczególnym komponentem, o którym należy wspomnieć przy kontrolerze, jest plik JavaScript *script.admin.js*, który znajduje się w katalogu *scripts/*. Wprowadza on bowiem elementy AJAX-u, czyli asynchronicznego JavaScriptu, celem zwiększenia interaktywności, użyteczności i łatwości obsługi aplikacji. W pliku tym deklarowane są polecenia walidacji pól formularzy, a także ustawienia autouzupełniania i funkcje sprawdzające obecność niewymagające przeładowania strony internetowej. Plik ten jest przypisywany do części kontrolera przede wszystkim dlatego, że z jednej strony – komunikuje się z modelem, a z drugiej strony – modyfikuje widok.

Najważniejszym elementem tego pliku JavaScript jest funkcja odpowiedzialna za asynchroniczne dodawanie i aktualizowanie obecności. Interaktywne ikony, za pomocą których można dodawać i modyfikować obecności, mają taki o to kod (Pogrubioną czcionką zaznaczono metadane, czyli parametry przechowywane w atrybucie *class* znacznika):

```
<a href="#" class="dynamic-obecnosc {NrIndeksu: '<indeks>',IdTermin: '<id terminu>',action: 'change'}">
```

Następnie JavaScript, za pomocą biblioteki jQuery z wtyczkami metadata i jQuery UI, przechwytuje zdarzenie naciśnięcia elementu z klasą „dynamic-obecnosc”, a następnie przetwarza metadane i wyświetlenia odpowiedzi użytkownikowi.

```
$(".dynamic-obecnosc").click(function(){
    var elem=$(this);
    var akcja=-1;
    var obecnosc_data=$(this).metadata();

    $( "#dialog-obecnosc" ).dialog({
        buttons: {
            "Obecność": function() {
                $.ajax({
                    type: "GET",
                    url: "admin/admin.api.php",
                    data: { action: "Action_DodajObecnosc",
                        NrIndeksu:obecnosc_data.NrIndeksu,IdTermin:obecnosc_data.IdTermin }
                }).done(function( msg ) {
                    sprawdzObecnosc(obecnosc_data.NrIndeksu,obecnosc_data.IdTermin,elem);
                });
                $( this ).dialog( "close" );
            }
        }
    });
});
```

```
    },  
    "Spóźnienie": function() {  
    $.ajax({  
        type: "GET",  
        url: "admin/admin.api.php",  
        data: { action: "Action_DodajSpoznienie",  
NrIndeksu:obecosc_data.NrIndeksu,IdTermin:obecosc_data.IdTermin }  
    }).done(function( msg ) {  
        sprawdzObecnosc(obecnosc_data.NrIndeksu,obecnosc_data.IdTermin,elem);  
    });  
    $( this ).dialog( "close" );  
    },  
    "Nieobecność": function() {  
    $.ajax({  
        type: "GET",  
        url: "admin/admin.api.php",  
        data: { action: "Action_UsunObecnosc",  
NrIndeksu:obecosc_data.NrIndeksu,IdTermin:obecosc_data.IdTermin }  
    }).done(function( msg ) {  
        sprawdzObecnosc(obecnosc_data.NrIndeksu,obecnosc_data.IdTermin,elem);  
    });  
    $( this ).dialog( "close" );  
    }  
    }  
    });  
    return false;  
    });  
});
```

Metoda *sprawdzObecnosc()* w argumencie przyjmuje numer indeksu studenta, id terminu oraz referencję na przycisk, który użytkownik nacisnął. Działanie tej funkcji polega na tym, że także przy pomocy *API*, wczytuje z bazy danych obecność studenta o podanym w parametrze numerze albumu w terminie o zadanym id, i modyfikuje ikonę i wartość przycisku tak, aby wskazywał na nowy typ obecności (obecność, nieobecność lub spóźnienie).