

PROJEKT ZESPOŁOWY

Inteligentny zamek

Autorzy:

MACIEJ MARCINIĄK

nr indeksu: 121996

e mail:

maciej.r.marcniak@student.put.poznan.pl

DAMIAN FILIPOWICZ

nr indeksu: 122002

e mail:

Damian.Filipowicz@student.put.poznan.pl

Spis treści

1	Aktorzy systemu	4
2	Opis składowych systemu	5
2.1	Urządzenie sterujące	5
2.2	Aplikacja mobilna	7
2.3	Aplikacja serwerowa obsługująca bazę danych	9
3	Diagram przypadków użycia	11
4	Diagramy sekwencji	13
5	Diagramy Klas	22
6	Komunikaty HTTP Request	26
7	Projekt bazy danych	32
8	Schemat elektryczny systemu	35
9	Zabezpieczenia systemu	36
10	Widok graficzny systemu	37
11	Testowanie aplikacji	52
12	Możliwe zagrożenia systemu	54
13	Dalsze perspektywy rozwoju projektu	55
14	Podział prac projektowych	56
15	Implementacja	57
16	Podsumowanie projektu	58

Wstęp

Inteligentny zamek powinien być systemem, który ma na celu zastąpienie starego modelu zabezpieczeń różnego rodzaju drzwi i skrytek, w którym używano tradycyjnych kluczy, czy szyfrów na klucze cyfrowe, którymi będzie można posługiwać się przy pomocy smartfonów z funkcją bluetooth. Celem tego rodzaju usprawnień będzie wyeliminowanie z życia codziennego sytuacji, w których użytkownik musi posiadać pęki kluczy. Zamiast tego dzięki temu systemowi może wszystkie klucze przechowywać w jednym miejscu (smartfonie). System jest dedykowany przede wszystkim budynkom zawierającym wiele pomieszczeń, które należy chronić, lecz wykorzystany może być również w użytku domowym.

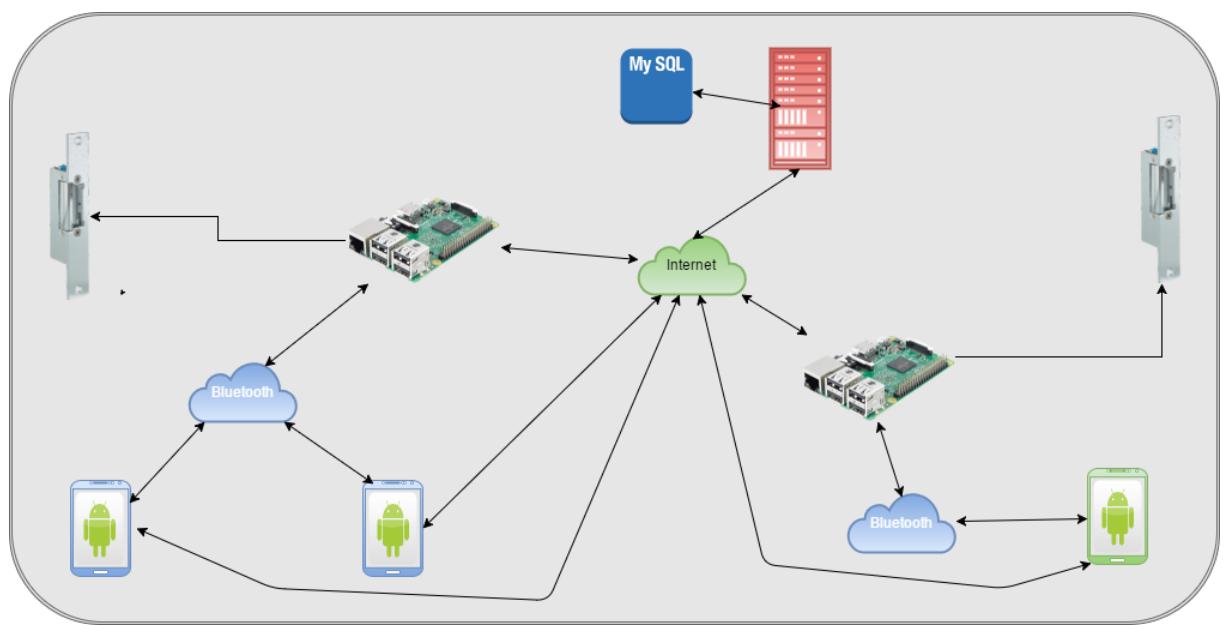
System składać się będzie z:

- urządzenia sterującego:
 - mikrokomputera Raspberry Pi 3,
 - serwomechanizmu / elektronicznego zamka,
- aplikacji mobilnej,
- aplikacji serwerowej obsługującej bazę danych.

System będzie spełniał wymagania dotyczące bezpieczeństwa poprzez zastosowanie szeregu funkcji kryptograficznych przy procesie uwierzytelniania jak i przy generowaniu kluczy takich jak np. funkcje skrótu, SSH, algorytmów szyfrowania asymetrycznego, systemu zarządzaniem kluczem publicznym (podpisu cyfrowego).

Używane klucze będą posiadały podpis cyfrowy, który jednoznacznie będzie definiował właściciela oraz stempel czasowy do określania ważności. Klucze będą mogły mieć w zależności od przeznaczenia różne okresy przedawnienia, np. pracownik zatrudniony w danym budynku posiadać będzie klucz o długim terminie ważności, a goście klucz jednorazowy bądź kilku godzinny bez możliwości odnowienia. Wszelkie dane dostępowe będą generowane i dystrybuowane na serwerze systemu, z możliwością zdalnej prośby o utworzenie kluczy tylko przez uprawnione przez administratora osoby.

Ogólny schemat systemu znajduje się na Rysunku 1.



Rysunek 1: Diagram wdrożeń

Rozdział 1

Aktorzy systemu

W systemie Inteligentnego zamka wyróżniamy następujących aktorów:

- **RaspberryPi** - jest to mikrokomputer Raspberry Pi 3 sterujący zamkiem,
- **serwomechanizm/elektrozamek** - jest to urządzenie służące do odblokowania/zablokowania zamka,
- **urządzenie mobilne** - jest to urządzenie posiadające system operacyjny, z funkcją bluetooth oraz posiadające możliwość instalacji aplikacji,
- **aplikacja serwerowa** - jest to program znajdujący się na serwerze z dostępem globalnym poprzez Internet,
- **użytkownik** - jest to osoba fizyczna operująca urządzeniem mobilnym, chcącą uzyskać dostęp do zamka.

Użytkowników dodatkowo dzieli się na grupy ze względu na uprawnienia w systemie:

- **gość** - posiada najniższe uprawniania, może jedynie posiadać klucze o krótkim okresie ważności, nie może generować nowych kluczy ani udostępniać ich,
- **użytkownik zalogowany** - posiada uprawnienia gościa, dodatkowo przechowywać może klucze o stałym dostępie do zamka (np. dostęp przez cały dzień),
- **administrator** - może wykonywać wszystkie czynności związane z uprawnieniami gościa i użytkownika zalogowanego, dodatkowo posiada dostęp do statystyk historii zamka, decyduje o rejestracji użytkowników zalogowanych.

Rozdział 2

Opis składowych systemu

2.1 Urządzenie sterujące

Zadaniem urządzenia sterującego, w którego skład wchodzić będą Raspberry Pi 3 oraz serwomechanizm/zamek elektroniczny, jest weryfikacja klucza cyfrowego przesyłanego przez urządzenie mobilne oraz otwieranie zamka przy pozytywnym wyniku weryfikacji.

Oprogramowanie mikrokomputera obejmuje system Linux raspbian-jessie oraz szereg podprogramów napisanych w języku Python. Skrypty programów łączą się do serwera w celu pobrania informacji o poprawności i dacie ważności certyfikatu dostępu. Jeśli dane będą poprawne to zostaje wysterowany serwomechanizm (lub wysłany impuls do elektrozamka), który otwiera zamek, w przeciwnym przypadku użytkownik zostanie poinformowany o odmowie dostępu, a nieudana próba dostania się do systemu zarejestrowana zostanie w bazie danych wraz z danymi właściciela klucza.

Funkcjonalność urządzenia sterującego przedstawiona została w Tabeli 2.1.

Tabela 2.1: Tabela wymagań funkcjonalnych urządzenia sterującego

Funkcja	Opis	Aktorzy
Parowanie urządzeń bluetooth	Parowanie bluetooth urządzenia mobilnego z Raspberry Pi	RaspberryPi, Urządzenie mobilne
Nasłuchiwanie połączenia bluetooth	Oczekuje na przychodzące połączenia bluetooth	RaspberryPi
Nawiązanie połączenia bluetooth	Każda próba nawiązania połączenia bluetooth zostanie zaakceptowana	RaspberryPi, Urządzenie mobilne
Pobranie pliku z kluczem cyfrowym przez bluetooth	Przesyłanie pliku z kluczem dostępowym z urządzenia mobilnego do Raspberry Pi poprzez bluetooth	RaspberryPi, Urządzenie mobilne, Gość
Weryfikacja poprawności klucza cyfrowego	Porównanie danych z bazy danych z otrzymanymi w kluczu dostępowym	RaspberryPi, Aplikacja serwerowa

Otwarcie zamka	Otwarcie zamka poprzez wysłanie sygnału PWM do serwomechanizmu lub zezwolenie zamka elektronicznego	RaspberryPi, Serwomechanizm/ elektrozamek
Zamknięcie zamka	Zamknięcie zamka poprzez wysłanie sygnału PWM do serwomechanizmu lub zezwolenie zamka elektronicznego	RaspberryPi, Serwomechanizm/ elektrozamek
Rejestracja próby dostępu	Zapis każdej pozytywnej i negatywnej próby weryfikacji klucza cyfrowego w tabeli bazy danych	RaspberryPi
Deszyfracja certyfikatu użytkownika	Deszyfracja pliku z certyfikatem dostępowym używając klucza publicznego użytkownika	RaspberryPi
Wysłanie żądania do serwera	Wysłanie zapytania do aplikacji serwerowej, w celu pobrania informacji	RaspberryPi, Aplikacja serwerowa
Pobranie awaryjnego klucza dostępu	Umożliwia wczytanie specjalnego hasła 512-bitowego do Raspberry Pi, który otwiera zamek bez konieczności dostępu do Internetu	RaspberryPi, Aplikacja mobilna, Administrator

Wymagania pozafunkcjonalne:

- jednocześnie może być weryfikowany tylko jeden użytkownik,
- zasięg połączenia bluetooth to maksymalnie 15m,
- niezbędny dostęp do Internetu do połączenia z aplikacją serwerową przy weryfikacji kluczy,
- narzut czasowy związany z weryfikacją poprawności klucza cyfrowego zależny od parametrów serwera i sieci,
- zainstalowany interpreter języka Python 2.7 z pakietem bibliotek (RSA, Bluetooth, Lightblue, RPi.GPIO, PyCrypto, Request),
- niezbędny ciągły dostęp do zasilania 5V o prądzie co najmniej 2.5A,
- ograniczenia prądowe dla serwomechanizmu lub zamka elektronicznego,
- narzut czasowy związany z uruchomieniem urządzenia - maksymalnie 20 sekund.

2.2 Aplikacja mobilna

Aplikacja mobilna w języku Java na platformę Android ma na celu przechowywanie w pamięci smartfona kluczy cyfrowych użytkownika oraz możliwość komunikacji człowiek-zamek-serwer. Program posiadać powinien interfejs graficzny, dzięki któremu będzie można wybrać, który zamek chce się otworzyć w danej chwili. Klucz cyfrowy przesyłany będzie bezprzewodowo do komputera sterującego zamkiem za pomocą sieci bluetooth. Aplikacja powinna posiadać również funkcję generowania kluczy tymczasowych, które można udostępniać osobom postronnym z ustalonym okresem ważności (jednorazowy, godzinny, od poniedziałku do piątku w godzinach od 8 do 16 itp.). W tym celu zostaje wysłana prośba do serwera poprzez Internet o wygenerowanie klucza o określonych parametrach.

Klucze przesyłane przez bluetooth powinny być szyfrowane algorytmem RSA (kluczem prywatnym). Klucz prywatny otrzymywany jest wraz z pierwszym certyfikatem oraz systematycznie odnawiany w celach bezpieczeństwa. Przypadku potrzeby przeniesienia klucza dostępowego na inne urządzenie należy wyeksportować klucz, a następnie zainportować do nowego urządzenia.

Funkcjonalność aplikacji przedstawiona została w Tabeli 2.2.

Tabela 2.2: Tabela wymagań funkcjonalnych aplikacji mobilnej

Funkcja	Opis	Aktorzy
Parowanie urządzeń bluetooth	Parowanie bluetooth urządzenia mobilnego z Raspberry Pi	Urządzenie mobilne, RaspberryPi
Nawiązywanie połączenia bluetooth	Nawiązanie połączenia bluetooth z konkretnym zamkiem identyfikując go jednoznacznie adresem MAC	Urządzenie mobilne, RaspberryPi
Przesłanie pliku klucza cyfrowego	Przesłanie pliku zawierającego klucz cyfrowy do urządzenia sterującego zamkiem. Komputer sterujący odsyła wynik weryfikacji (pozytywny lub negatywny)	Urządzenie mobilne, RaspberryPi
Utworzenie klucza cyfrowego dla gości	Utworzenie specjalnego klucza cyfrowego o ograniczonym dostępie oraz krótkim terminie ważności do użytku dla gości. Każde żądanie generowania klucza wymaga wpisania klucza bezpieczeństwa	Urządzenie mobilne, Aplikacja serwerowa, Użytkownik zalogowany
Udostępnianie klucza cyfrowego dla gości	Udostępnienie specjalnego klucza cyfrowego o ograniczonym dostępie oraz krótkim terminie ważności poprzez np. wiadomość MMS, bluetooth	Urządzenie mobilne, Użytkownik zalogowany
Wczytanie klucza cyfrowego z pliku	Umożliwia wczytanie do listy dostępnych zamków pliku klucza cyfrowego	Urządzenie mobilne, Gość
Pobranie z serwera nowego klucza cyfrowego	Umożliwia pobranie z serwera klucza cyfrowego i dodanie go do listy dostępnych zamków	Urządzenie mobilne, Aplikacja serwerowa, Użytkownik zalogowany

Prośba o przedłużenie ważności klucza	W celu przedłużenia ważności certyfikatu zostaje wysłana prośba do administratora systemu	Urządzenie mobilne, Aplikacja serwerowa, Użytkownik zalogowany
Listowanie dostępnych kluczy	Wyświetlenie na ekranie telefonu listy dostępnych kluczy do danych drzwi	Urządzenie mobilne, Gość
Modyfikacja danych kluczy cyfrowych	Modyfikacja nazw użytkownika, zamków. Pozwala spersonalizować opis zamków	Urządzenie mobilne, Aplikacja serwerowa, Użytkownik zalogowany
Szyfrowanie pliku klucza cyfrowego	Szyfrowanie algorytmem RSA klucza cyfrowego z wykorzystaniem klucza prywatnego	Urządzenie mobilne
Przechowywanie kluczy cyfrowych	Przechowywanie kluczy cyfrowych (szyfrowanych) w pamięci telefonu	Urządzenie mobilne
Podgląd do historii akcji zamków	Umożliwia przeglądanie historii akcji zamka, tzn. daty otwarcia przez kogo, daty zamknięcia	Urządzenie mobilne, Aplikacja serwerowa, Administrator
Autoryzacja użytkownika do aplikacji	Logowanie użytkownika poprzez podanie hasła i loginu do odblokowania aplikacji	Urządzenie mobilne, Aplikacja serwerowa, Użytkownik zalogowany
Rejestracja użytkownika	Założenie nowego konta użytkownika w systemie	Urządzenie mobilne, Aplikacja serwerowa, Gość
Logowanie użytkownika	Zalogowanie do systemu poprzez podanie adresu IP serwera, loginu oraz hasła	Urządzenie mobilne, Aplikacja serwerowa, Gość
Akceptacja przez administratora nowego użytkownika	Administrator systemu może zaakceptować i nadać uprawniania użytkownika	Urządzenie mobilne, Aplikacja serwerowa, Administrator
Zarządzanie ważnością certyfikatów dostępu	Dodawanie, usuwanie ważności certyfikatów dostępowych. Usunięcie praw użytkownika nie skutkuje unieważnieniem wygenerowanych przez niego certyfikatów	Urządzenie mobilne, Aplikacja serwerowa, Administrator
Przesłanie awaryjnego klucza dostępu	Umożliwia wczytanie specjalnego hasła 512-bitowego do Raspberry Pi, który otwiera zamek bez konieczności dostępu do Internetu	Urządzenie mobilne, RaspberryPi, Administrator
Tryb otwierania zamka	Komunikacja z Raspberry może odbywać się automatycznie lub na żądanie wyzwalane przyciskiem otwierania zamka z poziomu aplikacji	Urządzenie mobilne, RaspberryPi, Użytkownik zalogowany
Importowanie z pliku klucza prywatnego	Pobranie z pliku klucza prywatnego klucza dostępowego	Urządzenie mobilne
Eksportowanie do pliku klucza prywatnego	Przesłanie do pliku klucza prywatnego potrzebnego do szyfrowania kluczy dostępowych	Urządzenie mobilne

Wymagania pozafunkcjonalne:

- narzut czasowy związany z procesem szyfrowania kluczy cyfrowych (zależny od parametrów urządzenia mobilnego),
- zabezpieczenie transmisji danych poprzez szyfrowanie przy pomocy asymetrycznych kluczy cyfrowych,
- wymagany dostęp do Internetu do zarządzania kluczami, czy logowania,
- przyznanie uprawnień aplikacji do modułu bluetooth, wysyłania wiadomości MMS, Internetu,
- język aplikacji Polski,
- wersja androida minimalna 4.4, docelowa 5.0,

2.3 Aplikacja serwerowa obsługująca bazę danych

Rolą serwera w tym systemie będzie przechowywanie danych dostępowych w bazie danych MySQL oraz generowanie nowych kluczy cyfrowych poprzez program w języku Python. Aplikacja serwerowa oparta powinna być o technologię Python oraz serwera http Apache2. Serwer postawiony powinien być na odrębnym urządzeniu od instalacji zamka, lecz dopuszcza się ze względów ekonomicznych również postawienie serwera na wybranym (jeśli w systemie znajduje się wiele zamków) urządzeniu Raspberry Pi.

Funkcjonalność aplikacji serwerowej przedstawiona została w Tabeli 2.3.

Tabela 2.3: Tabela wymagań funkcjonalnych aplikacji serwerowej

Funkcja	Opis	Aktorzy
Utworzenie klucza cyfrowego użytkownika	Utworzenie pseudolosowego 128-bitowego klucza prywatnego i publicznego użytkownika potrzebnych do uwierzytelnienia kluczy dostępowych	Aplikacja serwerowa, Użytkownik zalogowany
Kontrola uprawnień użytkownika	Weryfikacja uprawnień użytkownika do wykonania danej czynności	Aplikacja serwerowa
Modyfikowanie wpisów w bazie danych	Pośredniczenie w modyfikacji danych zawartych w bazie danych	Aplikacja serwerowa
Przekazywanie wpisów z bazy danych	Pośredniczenie w przekazywaniu danych pobieranych z bazy danych	Aplikacja serwerowa
Rejestrowanie żądań dostępu	Zapisywanie danych użytkownika ubiegającego się o dostęp do serwera	Aplikacja serwerowa
Pobranie historii dostępu zamka	Pobranie statystyk związanych z historią dostępu do zamka	Aplikacja serwerowa, Administrator
Zablokowanie dostępu użytkownika	Zablokowanie certyfikatu dostępowego, np. w przypadku kradzieży telefonu	Aplikacja serwerowa, Administrator

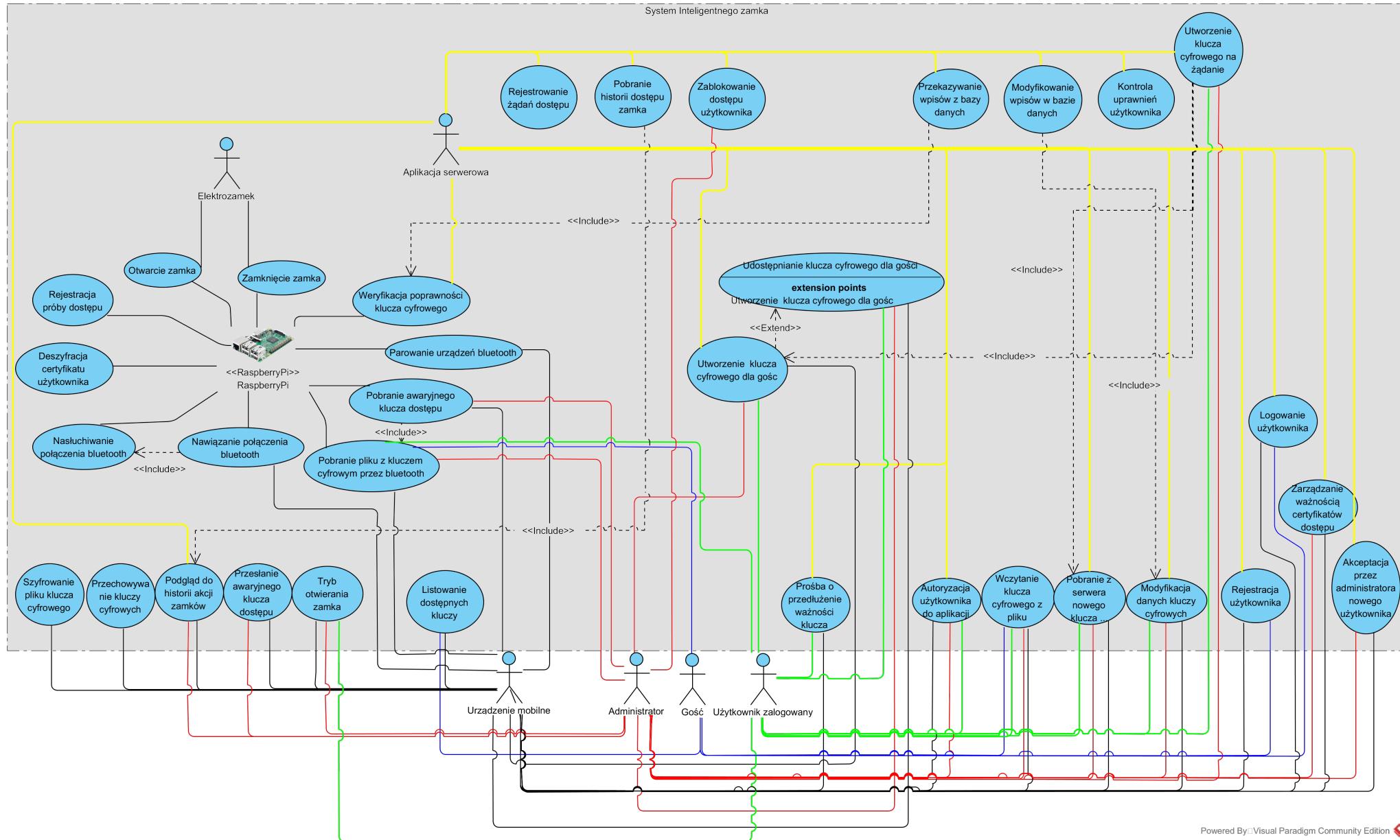
Wymagania niefunkcjonalne:

- ograniczenie pamięci dostępnej dla bazy danych (32Gb - pamięć niezbędna dla systemu operacyjnego i oprogramowania),
- ograniczenie liczby obsługiwanych zamków zależna od wielkości dostępnej pamięci i liczby użytkowników,
- narzut czasowy związany z generowaniem nowych kluczy,
- ograniczenie liczby użytkowników wykonujących jednocześnie żądania do serwera - 9 urządzeń,
- wymagany system operacyjny Linux,
- zainstalowany interpreter języka Python 2.7 wraz z pakietem bibliotek (MySQL, RSA, Django, PyCrypto),
- zainstalowany i skonfigurowany serwer Apache2,
- dostęp do Internetu do połączenia z zamkami i urządzeniami mobilnymi,
- zabezpieczenie bazy danych hasłem generowanym losowo.

Rozdział 3

Diagram przypadków użycia

Diagram przypadków użycia (funkcjonalności) systemu wraz z opowiadającymi aktorami przedstawiono na Rysunku 3.1.



Rysunek 3.1: Diagram przypadków użycia

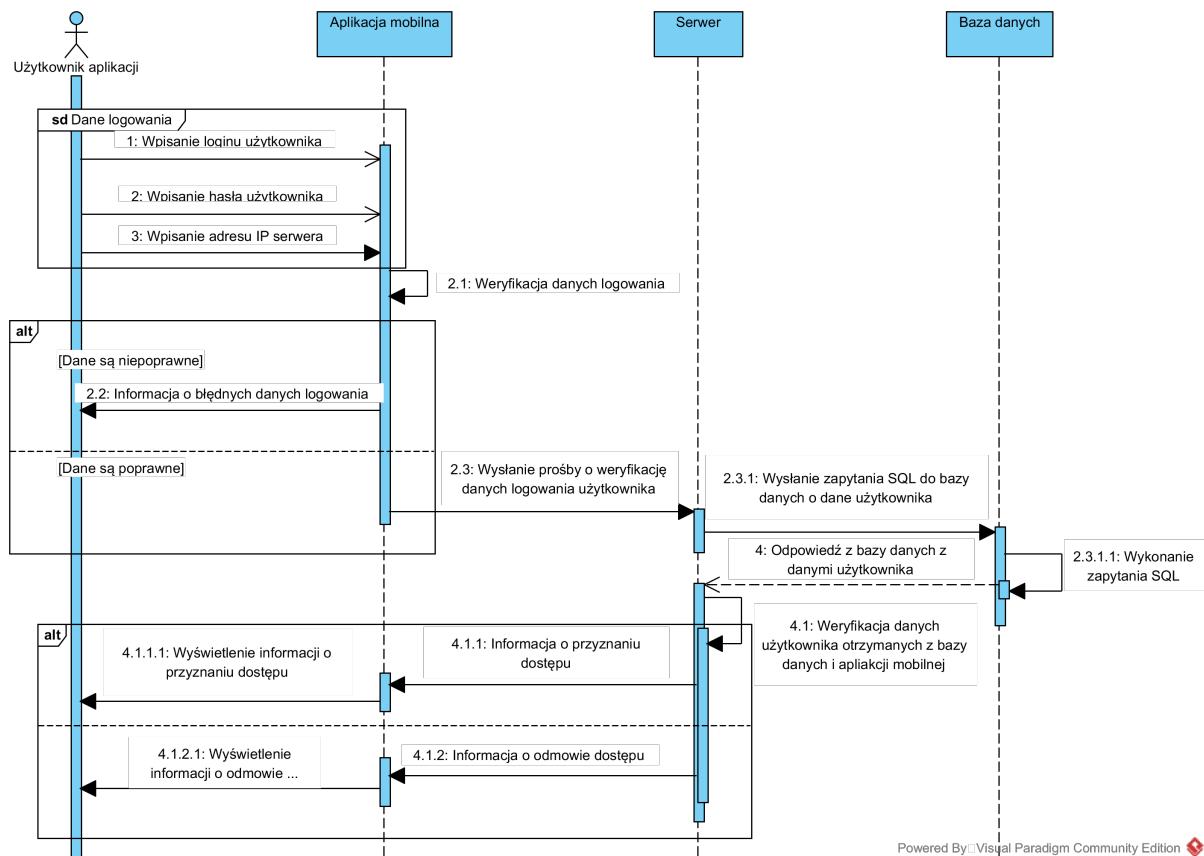
Rozdział 4

Diagramy sekwencji

Diagramy poniżej przedstawione mają mieć na celu przybliżenie ogólnego działania systemu. Schematy nie są odzwierciedleniem poszczególnych przypadków użycia, mogą zawierać nawet szczegółowe odniesienia do wielu z nich.

Logowanie użytkownika

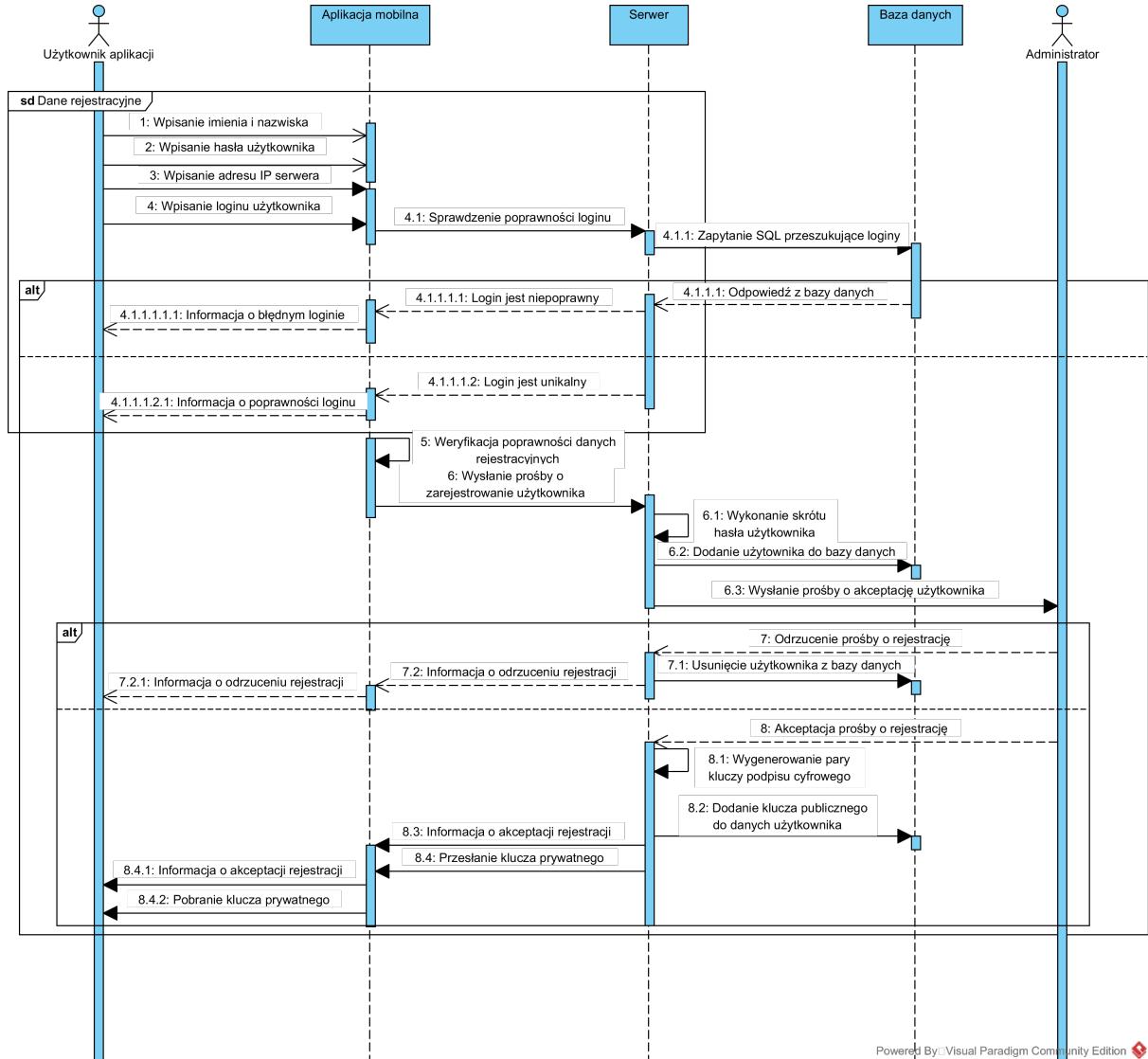
Diagram 4.1 opisuje proces logowania użytkownika do aplikacji mobilnej z weryfikacją danych na serwerze.



Rysunek 4.1: Diagram sekwencji logowanie użytkownika

Rejestracja użytkownika

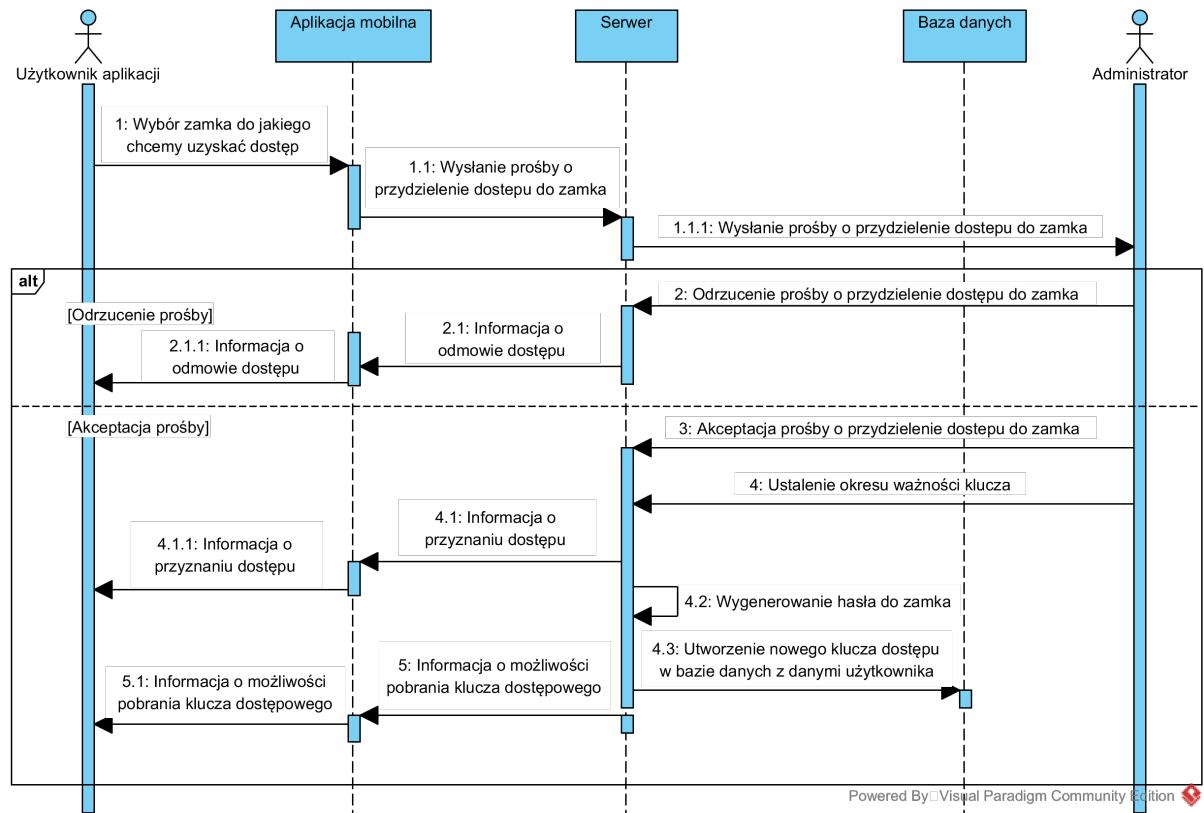
Diagram 4.2 opisuje proces rejestracji użytkownika do systemu z weryfikacją danych na serwerze oraz podjęciem decyzji przez administratora o przydzielaniu dostępu.



Rysunek 4.2: Diagram sekwencji rejestracja użytkownika

Generowanie certyfikatu użytkownika

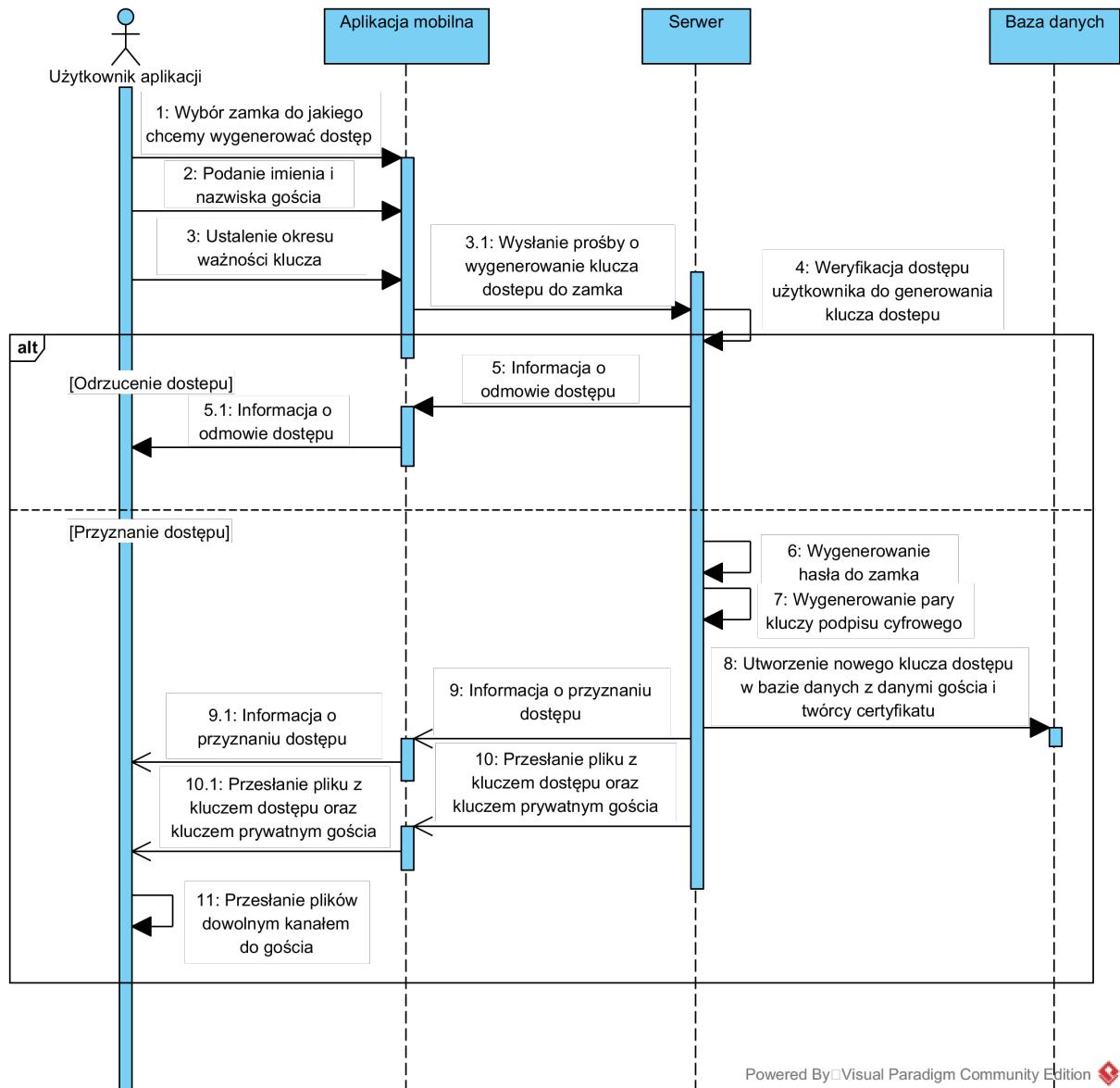
Diagram 4.3 przedstawia sekwencję operacji podczas generowania certyfikatu użytkownika oraz podjęcia decyzji przez administratora o przydzieleniu dostępu.



Rysunek 4.3: Diagram sekwencji generowanie certyfikatu użytkownika

Generowanie certyfikatu dla gościa

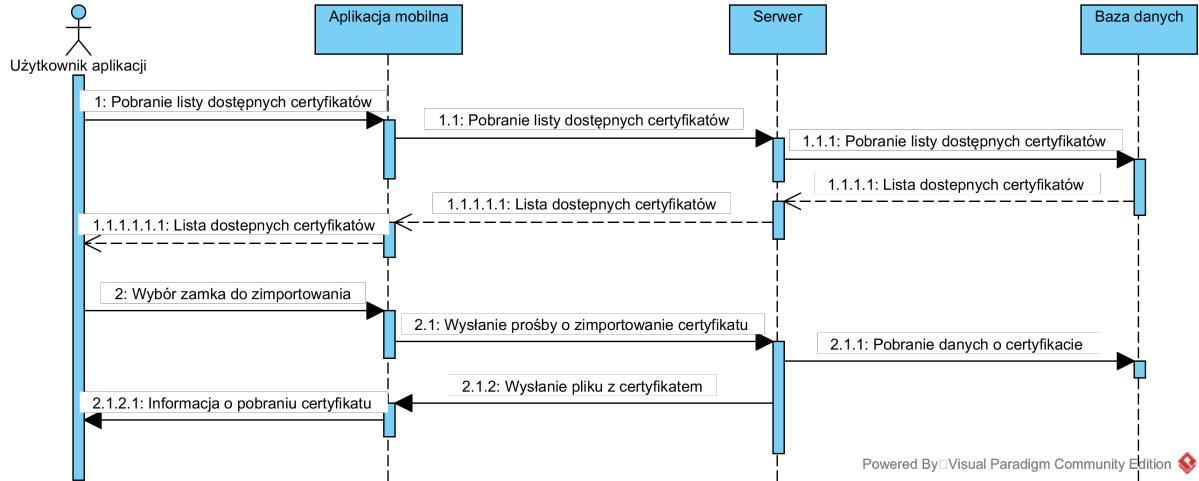
Diagram 4.4 przedstawia sekwencję operacji podczas generowania certyfikatu dla gościa przez użytkownika zalogowanego wraz z wyborem okresu ważności klucza oraz udostępnienie certyfikatu gościowi.



Rysunek 4.4: Diagram sekwencji generowanie certyfikatu gościa

Importowanie certyfikatu z serwera

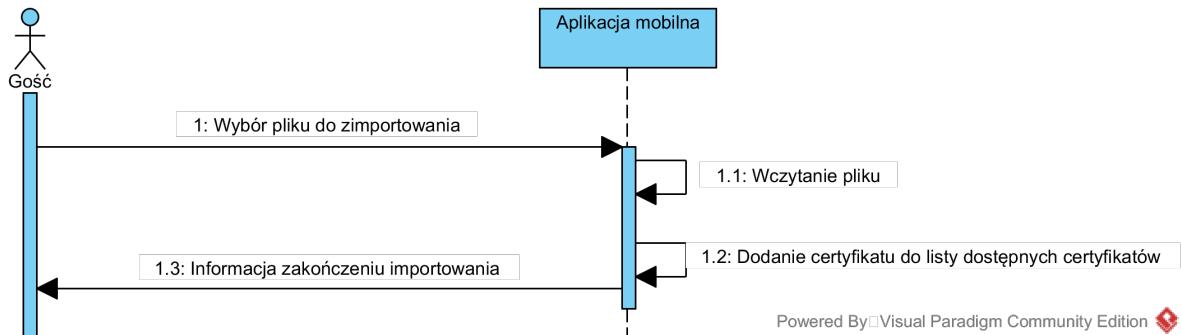
Diagram 4.5 opisuje operację importowania certyfikatu z serwera.



Rysunek 4.5: Diagram sekwencji generowanie importowanie certyfikatu z serwera

Importowanie certyfikatu z pliku

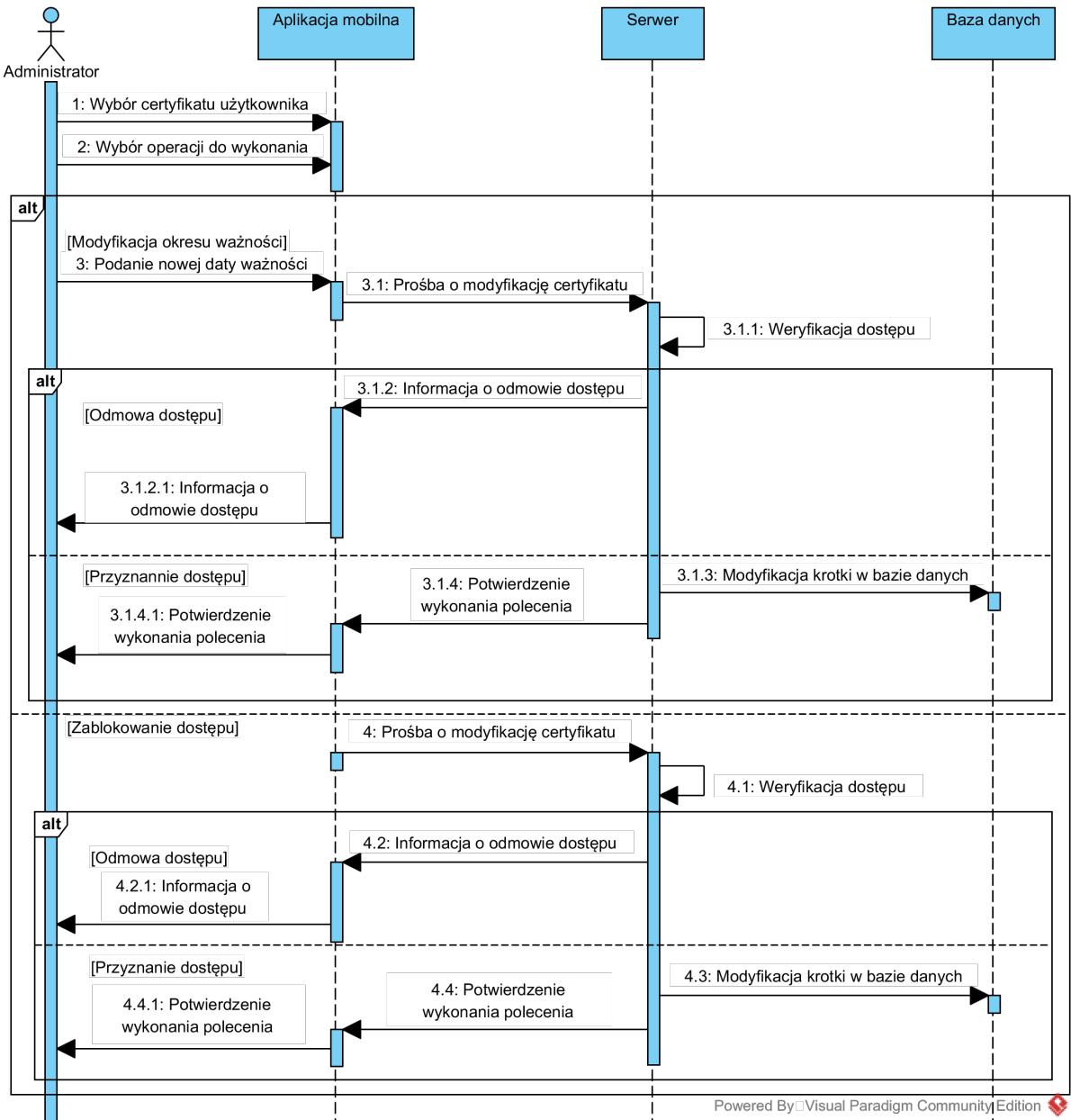
Diagram 4.6 opisuje operację importowania certyfikatu z pliku.



Rysunek 4.6: Diagram sekwencji generowanie importowanie certyfikatu z pliku

Modyfikacja certyfikatu

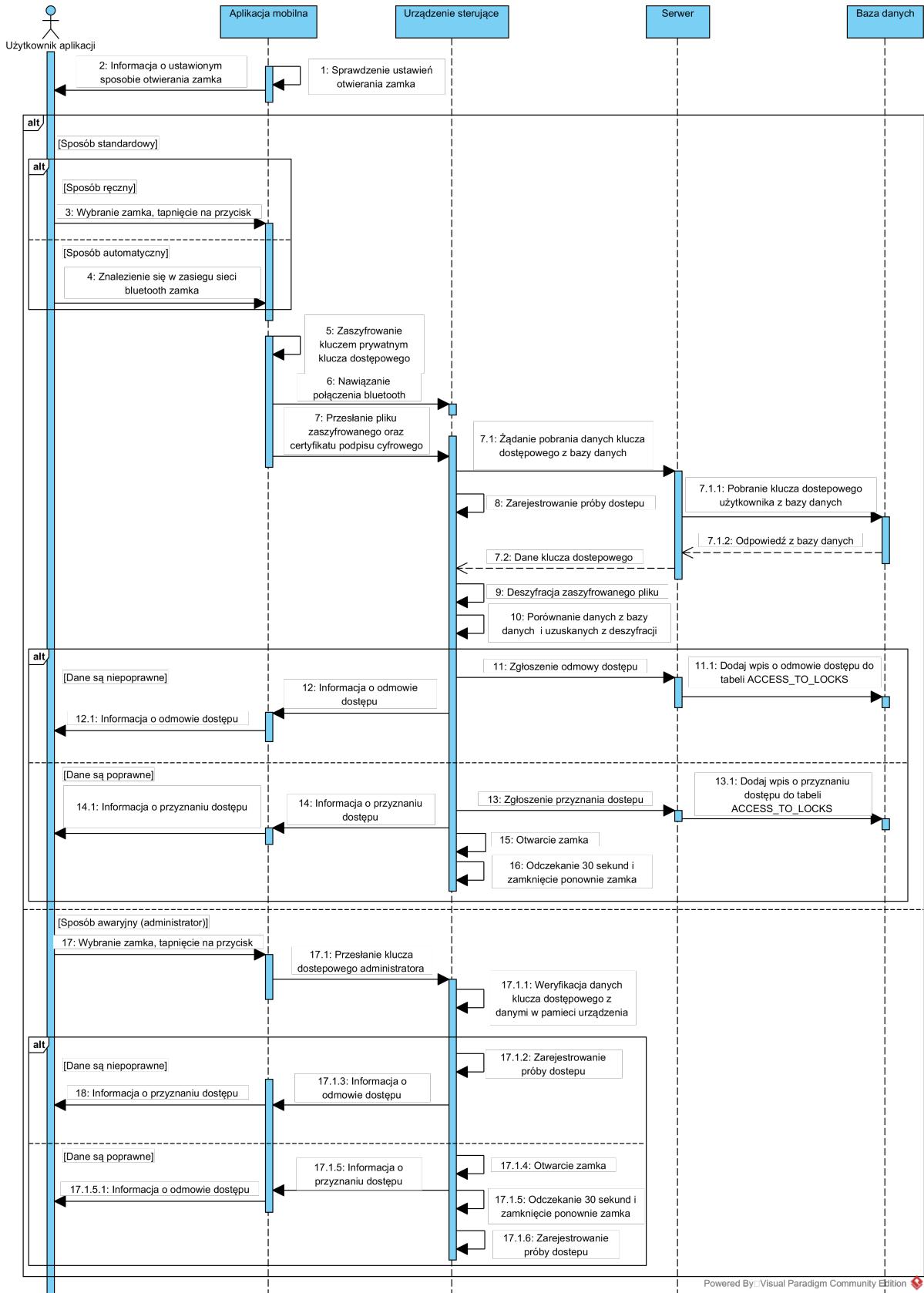
Diagram 4.7 opisuje proces modyfikacji, przez administratora, certyfikatów znajdujących się w bazie danych.



Rysunek 4.7: Diagram sekwencji modyfikacji certyfikatu

Otwieranie zamka

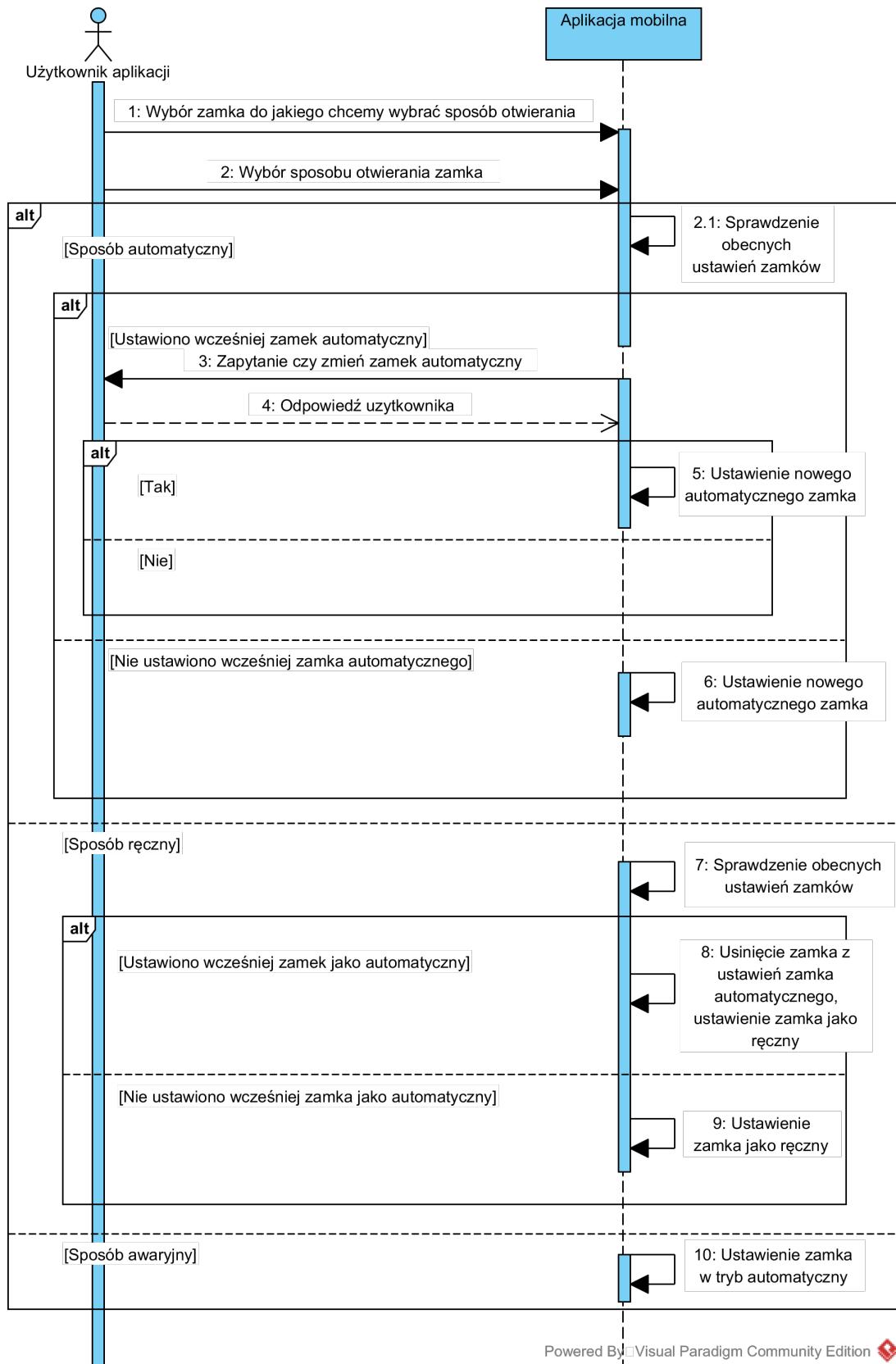
Diagram 4.8 przedstawia operacje wykonywane podczas otwierania zamka wraz z jego automatycznym zamknięciem.



Rysunek 4.8: Diagram sekwencji otwieranie zamka

Wybór sposobu otwierania zamka

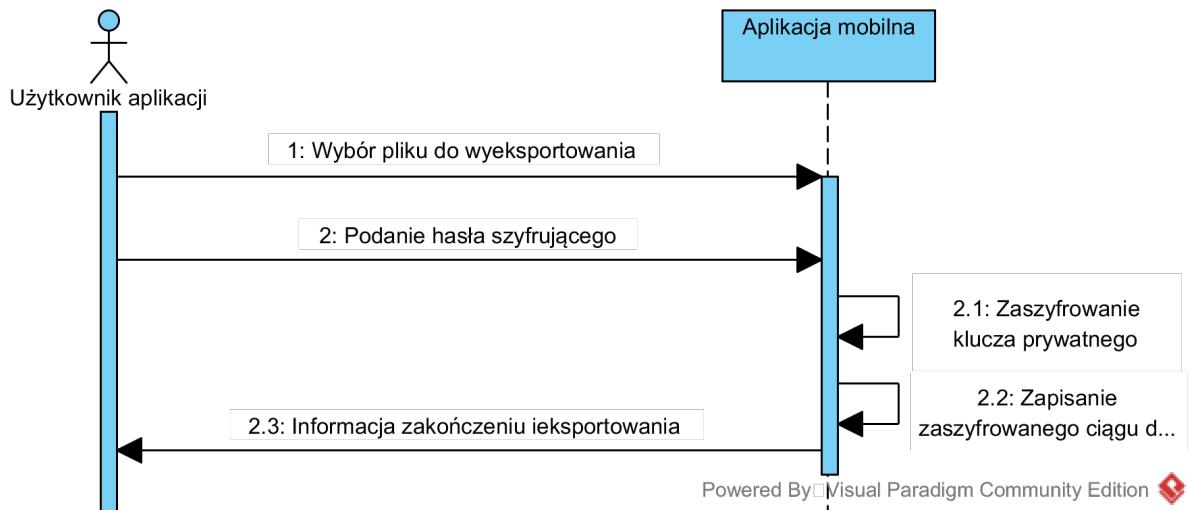
Diagram 4.9 przedstawia operacje wykonywane podczas wyboru sposobu otwierania zamka.



Rysunek 4.9: Diagram sekwencji ustawiania sposobu otwierania zamka

Eksportowanie klucza prywatnego użytkownika

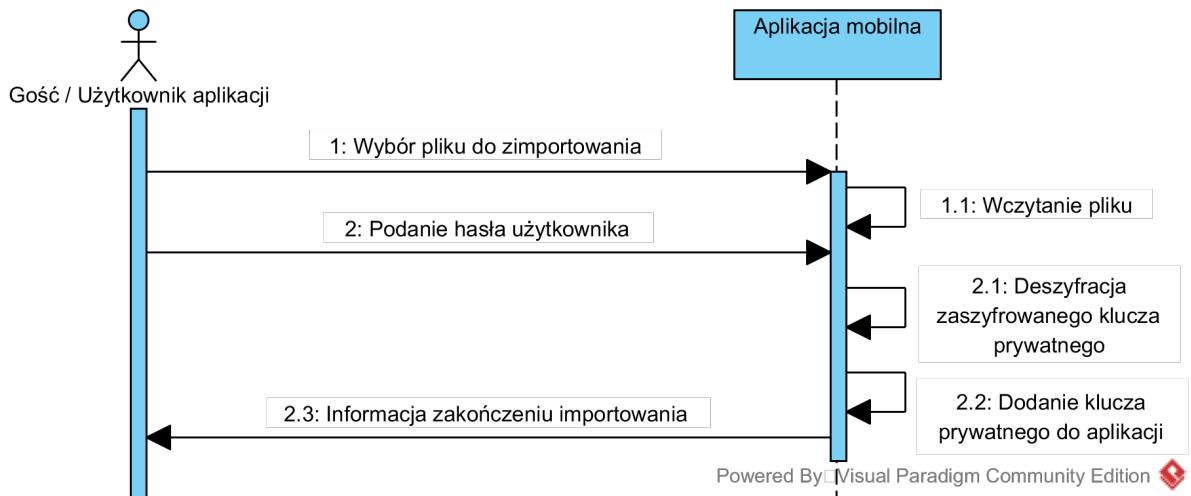
Diagram 4.10 opisuje operację eksportowania klucza prywatnego użytkownika.



Rysunek 4.10: Diagram sekwencji eksportowania klucza prywatnego użytkownika

Importowanie klucza prywatnego użytkownika

Diagram 4.11 opisuje operację importowania klucza prywatnego użytkownika.



Rysunek 4.11: Diagram sekwencji importowania klucza prywatnego użytkownika

Rozdział 5

Diagramy Klas

Urządzenie sterujące

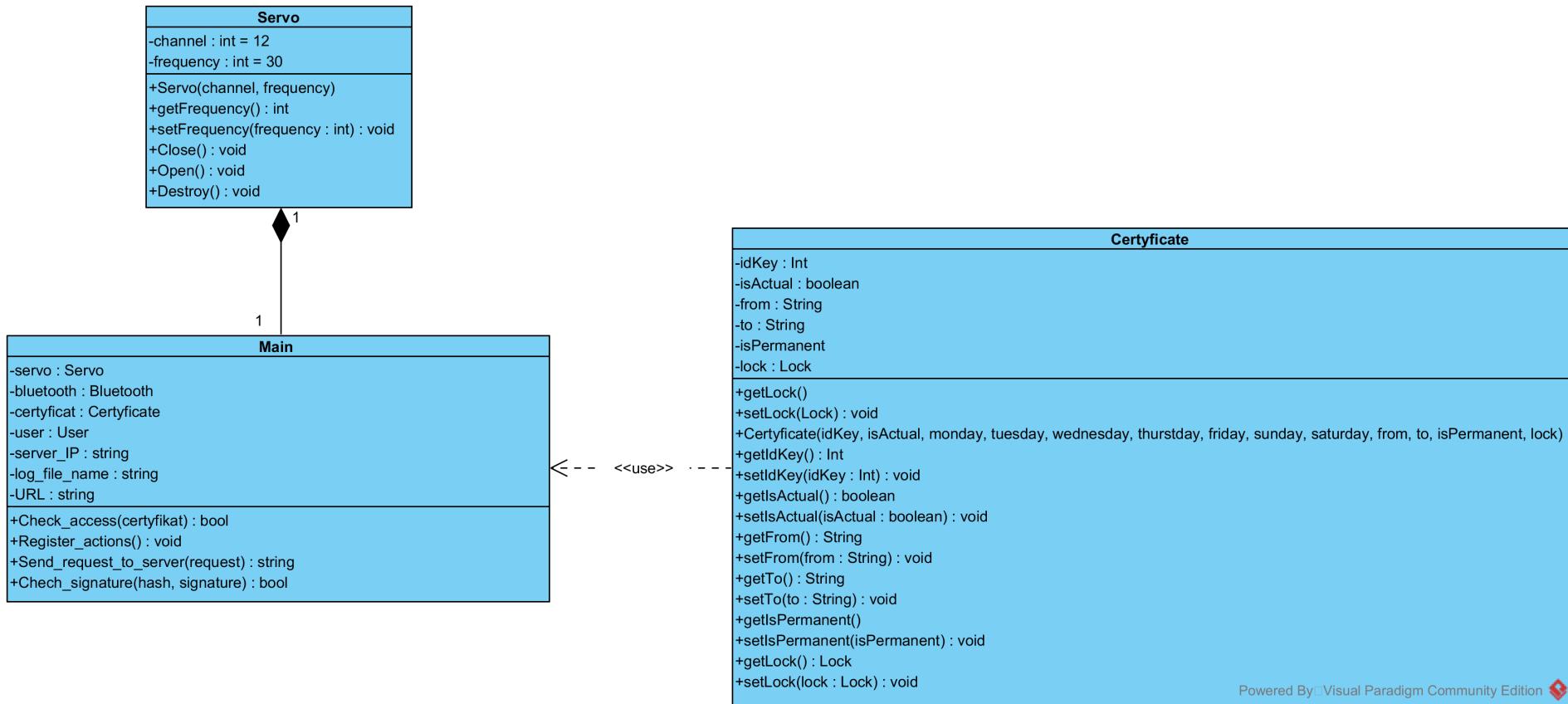
Diagram 5.1 przedstawia opis klas znajdujących się w urządzeniu sterującym.

Aplikacja mobilna

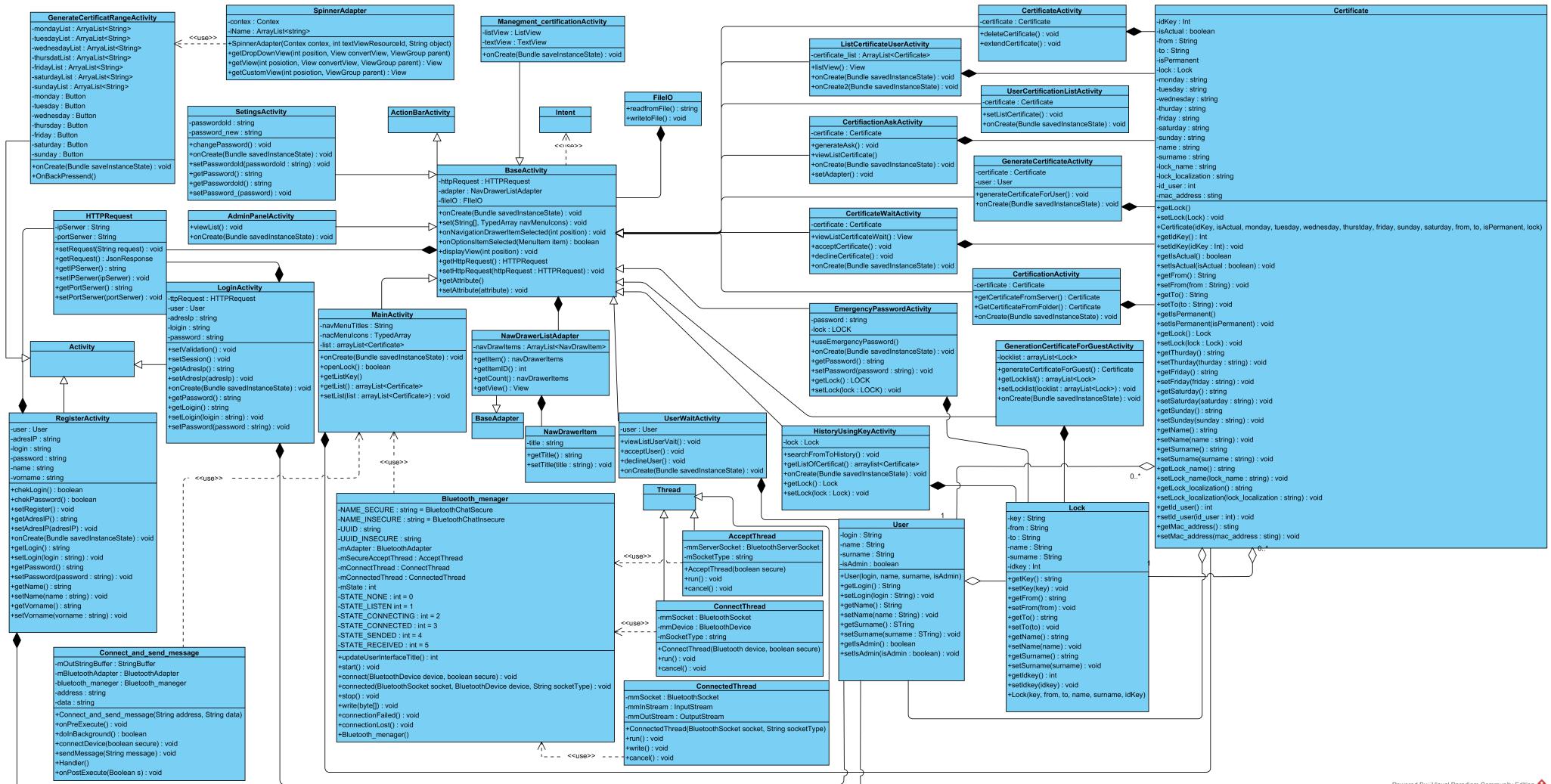
Diagram 5.2 przedstawia opis klas znajdujących się w aplikacji mobilnej.

Aplikacja serwerowa

Diagram 5.3 przedstawia opis klas znajdujących się w aplikacji serwerowej.



Rysunek 5.1: Diagram klas urządzenia sterującego



Views
<pre> -username : string -userpassword : string -databasename : string -databaseaddress : string +api_login(username, password) : JsonResponse +api_refister(login, password, name, surname, publickey) : JsonResponse +api_logout(login, token) : JsonResponse +api_download_all_certificate(login, token) : JsonResponse +api_download_all_locks(login, token) : JsonResponse +api_download_all_user(login, token) : JsonResponse +api_RPI_download_certificate(certificate_id, RPI_MAC, login_user) : JsonResponse +api_RPI_access_decision(decision, certificate_id) : JsonResponse +api_deactivation(login, token) : JsonResponse +api_request_new_certificate(login, token, lock_id) : JsonResponse +api_admin_generate_new_certificate(login, token, user_id, lock_id, from_date, to_date, monday, tuesday, wednesday, thursday, friday, saturday, sunday, is_pernament, name, surname) : JsonResponse +api_admin_history(login, token) : JsonResponse +api_admin_download_all_certificate(login, token) : JsonResponse +api_admin_deactivation(login, token, certificate_id) : JsonResponse +api_generate_new_quest_certificate(login, token, lock_id, quest_name, quest_surname, period) : JsonResponse +api_change_password(login, token password, newpassword) : JsonResponse </pre>



URLs
<pre> +api_login(views, name) +api_register(views, name) +api_logout(views, name) +api_download_all_certificate(views, name) +api_download_all_locks(views, name) +api_deactivations(views, name) +api_request_new_certificate(views, name) +api_generate_new_quest_certificate(views, name) +api_admin_history(views, name) +api_admin_generate_new_certificate(views, name) +api_admin_download_all_certificate(views, name) +api_admin_deactivation(views, name) +api_admin_register_waiting(views, name) +api_admin_register_decision(views, name) +api_admin_certificate_waiting(views, name) +api_admin_certificate_decision(views, name) +api_RPI_download_certificate(views, name) +api_RPI_access_decision(views, name) +api_download_all_users(views, name) +api_change_password(views, name) </pre>

Rysunek 5.3: Diagram klas aplikacji serwerowej

Rozdział 6

Komunikaty HTTP Request

Moduły systemu komunikują się z serwerem za pomocą komunikatów HTTP. Tabela 6.1 przedstawia komunikaty serwera przeznaczone dla urządzenia Raspberry Pi.

Tabela 6.2 zawiera komunikaty pomiędzy serwerem, a urządzeniem mobilnym.

Tabela 6.1: Tabela komunikatów HTTP dla Raspberry Pi

Adres URL	Parametry	Odpowiedź	Opis
/api/RPI/download/cerificate/	certificate_id — identyfikator certyfikatu RPI_MAC — adres MAC urządzenia login_user — login użytkownika	"data": dict_all_certificate, "public_key": public_key "data": "invalid"	Pobranie certyfikatu użytkownika
/api/RPI/access_decision/	certificate_id — identyfikator certyfikatu decision — informacja o odmowie/akceptacji dostępu	"status": "ok" "data": "invalid"	Informacja do serwera o statusie otwierania zamka

Tabela 6.2: Tabela komunikatów HTTP dla urządzenia mobilnego

Adres URL	Parametry	Odpowiedź	Opis
/api/login/	username — login użytkownika password — hasło użytkownika	"status": "ok", "token": token "status": "ERROR PASSWORD", "token": "invalid" "status": "not activated", "token": "invalid"	Logowanie użytkownika do aplikacji
/api/register/	login — login użytkownika password — hasła użytkownika name — imię użytkownika surname — nazwisko użytkownika publickey — klucz publiczny użytkownika	"status": "ERROR" "status": "REGISTER OK"	Rejestracja nowego użytkownika do aplikacji
/api/logout/	login — login użytkownika token — klucz sesji logowania	"status": "logout" "status": "invalid"	Wylogowanie użytkownika z aplikacji
/api/download/all_certificate/	login — login użytkownika token — klucz sesji logowania	"data": dict_all_certificate "status": "invalid"	Pobranie wszystkich dostępnych certyfikatów użytkownika
/api/download/all_locks/	login — login użytkownika token — klucz sesji logowania	"data": dict_all_locks "status": "invalid"	Pobranie listy wszystkich dostępnych zamków w systemie

/api/download/all_user/	login — login użytkownika token — klucz sesji logowania	"data": dict_all_users ————— "status": "invalid"	Pobranie listy wszystkich użytkowników systemu
/api/deactivation/	login — login użytkownika token — klucz sesji logowania certificate_id — identyfikator certyfikatu do usunięcia	"status": "ok" ————— "status": "invalid"	Usunięcie dostępu do certyfikatu
/api/request_new_certificate/	login — login użytkownika token — klucz sesji logowania lock_id — identyfikator zamka	"status": "ok" ————— "status": "invalid"	Wnioskowanie o nowy certyfikat
/api/generate_new_guest_certificate/	login — login użytkownika token — klucz sesji logowania lock_id — identyfikator zamka quest_name — imię gościa guest_surname — nazwisko gościa from_to — data i godzina od której ważny jest certyfikat period — okres ważności certyfikatu	"status": "ok" ————— "status": "invalid" ————— "status": "denied"	Generowanie nowego certyfikatu gościa
/api/change_password/	login — login użytkownika token — klucz sesji logowania newpasswd — nowe hasło użytkownika	"status": "ok" ————— "status": "invalid"	Zmiana hasła użytkownika
/api/admin/history/	login — login użytkownika token — klucz sesji logowania	"data": dict_history ————— "status": "invalid"	Pobranie historii użycia zamków w systemie (administrator)

/api/admin/download/ all_certificate/	login — login użytkownika token — klucz sesji logowania	"data": dict_all_certificate "status": "invalid"	Pobranie wszystkich certyfikatów z systemu (administrator)
/api/admin/deactivation/	login — login użytkownika token — klucz sesji logowania certificate_id — identyfikator certyfikatu do usunięcia	"status": "ok" "status": "invalid"	Usunięcie dostępu do certyfikatu (administrator)
/api/admin/register_waiting/	login — login użytkownika token — klucz sesji logowania	"status": "ok" "status": "invalid"	Pobranie listy oczekujących użytkowników na zaakceptowanie rejestracji
/api/admin/register_decision/	login — login użytkownika token — klucz sesji logowania user_login — login użytkownika, którego dotyczy decyzja decision — decyzja True/False dotycząca akceptacji rejestracji	"status": "ok" "status": "invalid"	Podjęcie decyzji przez administratora dotyczącej rejestracji użytkownika o danym loginie
/api/admin/certificate_waiting/	login — login użytkownika token — klucz sesji logowania	"status": "ok" "status": "invalid"	Pobranie listy oczekujących certyfikatów na zaakceptowanie
/api/admin/certificate_decision/	login — login użytkownika token — klucz sesji logowania certificate_id — identyfikator certyfikatu, którego dotyczy decyzja	"status": "ok" "status": "invalid"	Podjęcie decyzji przez administratora dotyczącej przyjęcia

<p>/api/admin/ generate_new_certificate/</p>	<p>login — login użytkownika token — klucz sesji logowania user_id — identyfikator użytkownika lock_id — identyfikator zamka from_date — data od której obowiązuje certyfikat to_date — data do której obowiązuje certyfikat monday...sunday — 7 parametrów oznaczających zakresy godzin w poszczególnych dniach tygodnia is_persistent — czy dostęp jest ciągły name — imię użytkownika certyfikatu surname — nazwisko użytkownika certyfikatu</p>	<p>"status": "ok" ————— "status": "invalid"</p>	<p>Generowanie nowego certyfikatu (administrator)</p>
--	---	---	---

Rozdział 7

Projekt bazy danych

Baza danych będzie składać się z pięciu tabel:

- **USERS** — przechowuje dane użytkowników oraz dane niezbędne przy weryfikacji logowania,
- **LOCKS** — zawiera informacje na temat dostępnych w systemie zamków,
- **ACCESS_TO_LOCKS** — archiwizuje próby użycia certyfikatów,
- **LOCKS_KEYS** — zawiera wszystkie klucze dostępowe użytkowników,
- **WAIT_LOCKS_KEYS** — przetrzymuje klucze dostępowe oczekujące na zatwierdzenie przez administratora.

Wiersz tabeli USERS zawierać musi:

- **ID_USER** — unikalny identyfikator (klucz główny) użytkownika składający się z 10 cyfr,
- **LOGIN** — unikalna nazwa użytkownika niezbędna podczas logowania, zawierająca nie więcej niż 255 znaków,
- **PASSWORD** — hasło zapisane w postaci skrótu, potrzebne do autoryzacji dostępu użytkownikowi,
- **PUBLIC_KEY** — klucz publiczny użytkownika potrzebny do podpisu cyfrowego,
- **NAME** - imię użytkownika,
- **SURNAME** — nazwisko użytkownika,
- **IS_ADMIN** — pole boolowskie wskazujące czy dany użytkownik jest administratorem czy nie,
- **TOKEN** — generowany ciąg pseudolosowy klucz sesji logowania,
- **ISACTIVATED** — pole boolowskie oznaczające, czy dane konto jest zaakceptowane (aktywowane) przez administratora.

Zamek opisywany jest poprzez kolumny:

- **ID_LOCK** — unikalny identyfikator (klucz główny) zamka składający się z 10 cyfr,
- **NAME** — unikalna nazwa zamka,
- **MAC_ADDRESS** — adres fizyczny urządzenia sterującego zamkiem,
- **LOCALIZATION** — nieobowiązkowe pole opisujące fizyczne położenie zamka,
- **ADMIN_KEY** — wartość klucza awaryjnego dla administratora.

Klucz dostępowy składa się z:

- **ID_KEY** — unikalny identyfikator (klucz główny) klucza dostępowego składający się z 10 cyfr,
- **ID_LOCK** — klucz obcy do tabeli przechowującej dostępne zamki,
- **ID_USER** — klucz obcy do tabeli przechowującej dane użytkownika, jest to pole służące do określenia kto utworzył klucz dostępu,
- **KEY** — unikalna wartość certyfikatu dostępu,
- **FROM** — data od której obowiązuje klucz,
- **TO** — data do której obowiązuje klucz,
- **ISACTUAL** — data wygaśnięcia klucza, jeśli równa TO, oznacza to że klucz utracił ważność z powodu czasu, jeśli różna oznacza, to że zablokowano z innego powodu ważność,
- **MONDAY** — słowne określenie, w których godzinach zostanie przyznany dostęp w poniedziałki,
- **TUESDAY** — słowne określenie, w których godzinach zostanie przyznany dostęp we wtorki,
- **WEDNESDAY** — słowne określenie, w których godzinach zostanie przyznany dostęp w środy,
- **THURSDAY** — słowne określenie, w których godzinach zostanie przyznany dostęp w czwartki,
- **FRIDAY** — słowne określenie, w których godzinach zostanie przyznany dostęp w piątki,
- **SATURDAY** — słowne określenie, w których godzinach zostanie przyznany dostęp w soboty,
- **SUNDAY** — słowne określenie, w których godzinach zostanie przyznany dostęp w niedziele,
- **IS_PERNAMENT** — zmienna boolowska oznaczająca czy dostęp jest zawsze,
- **NAME** - imię osoby, której dotyczy certyfikat,
- **SURNAME** — nazwisko osoby, której dotyczy certyfikat.

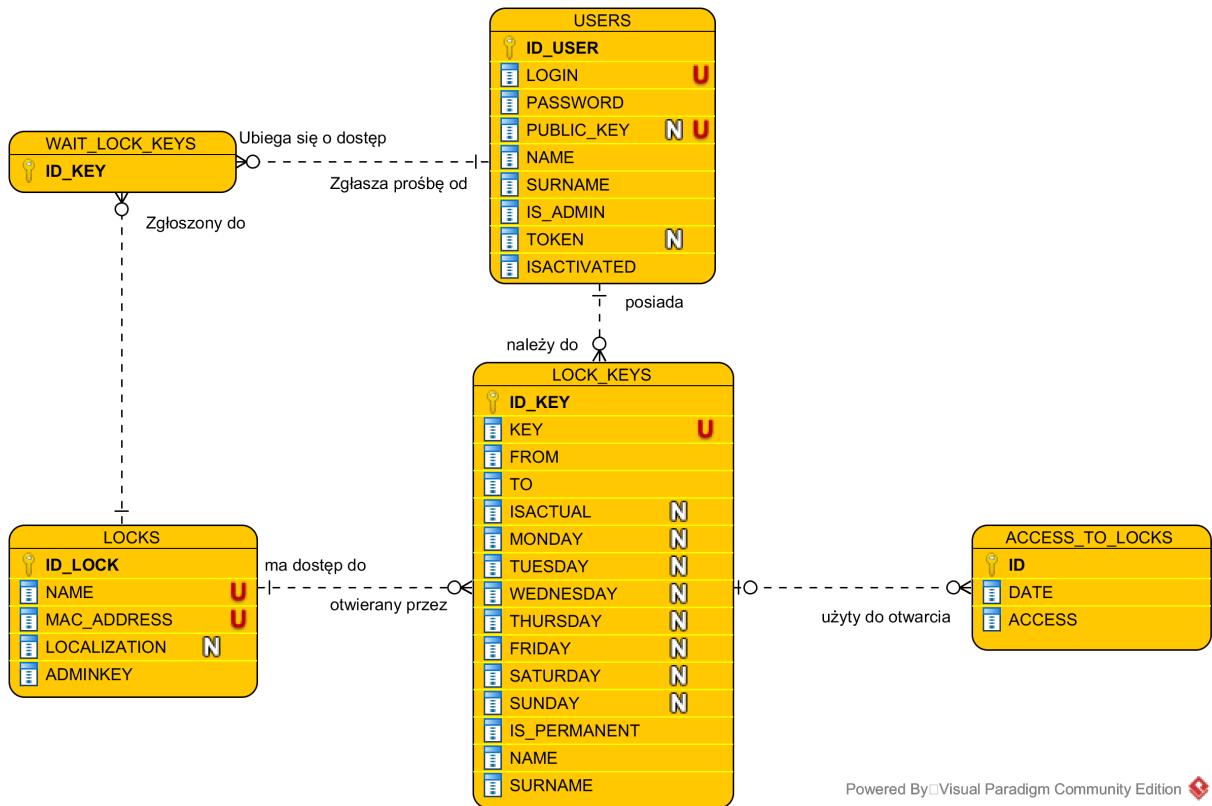
W tabeli archiwizującej akcje na zamku znajdują się takie dane jak:

- **ID** — unikalny identyfikator (klucz główny) akcji wykonanej na certyfikacie składający się z 10 cyfr,
- **ID_KEY** — klucz obcy do tabeli przechowującej klucze dostępowe, dzięki tej informacji możemy uzyskać dane o zamku, który został otwierany jak również do kogo należał klucz,
- **DATE** — dokładna data z godziną użycia klucza dostępowego,
- **ACCESS** — binarna flaga informująca czy dostęp został przyznany czy odmówiony.

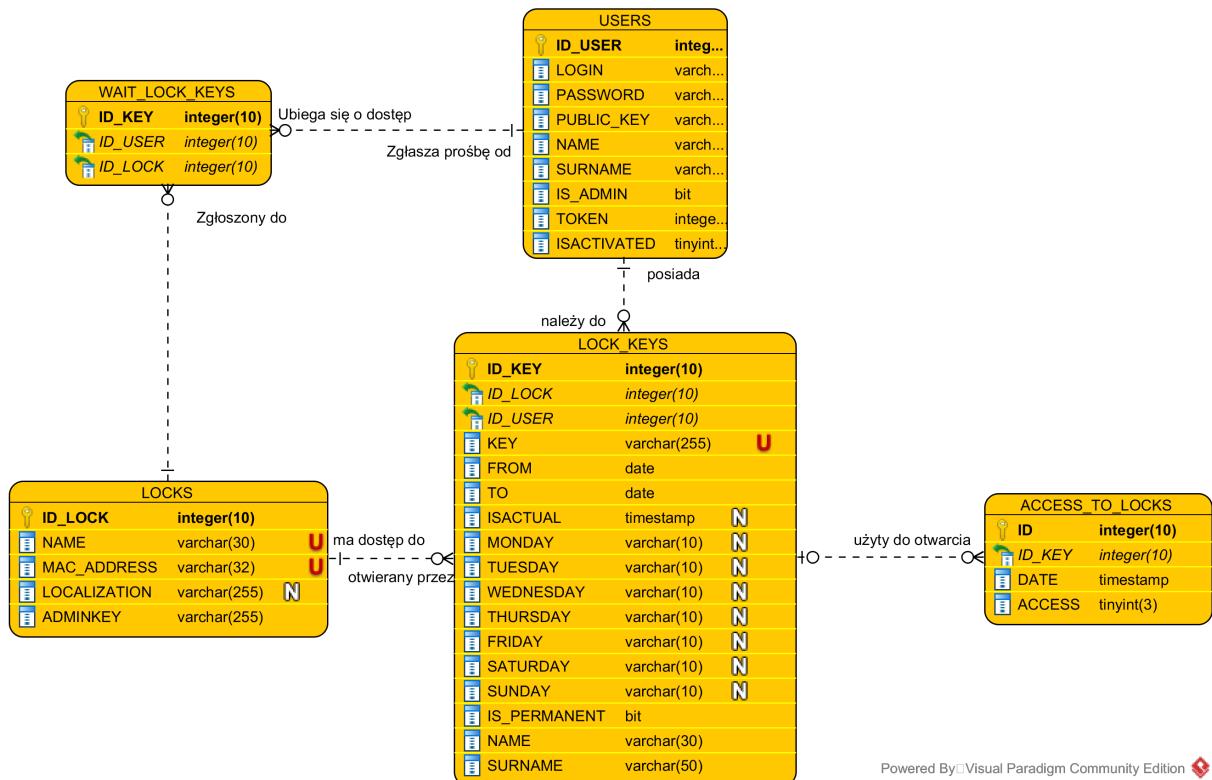
Tabela WAIT_LOCKS_KEYS składa się z:

- **ID_KEY** — unikalny identyfikator (klucz główny) oczekującego certyfikatu,
- **ID_LOCK** — klucz obcy do tabeli LOCKS, oznacza zamek do którego jest zgłoszana prośba dostępu,
- **ID_USER** — klucz obcy do tabeli USERS, oznacza użytkownika który zgłasza prośbę o dostęp do zamka.

Diagramy bazy danych odpowiednio encji i relacji przedstawione zostały na Rysunkach 7.1 i 7.2.



Rysunek 7.1: Diagram encji bazy danych

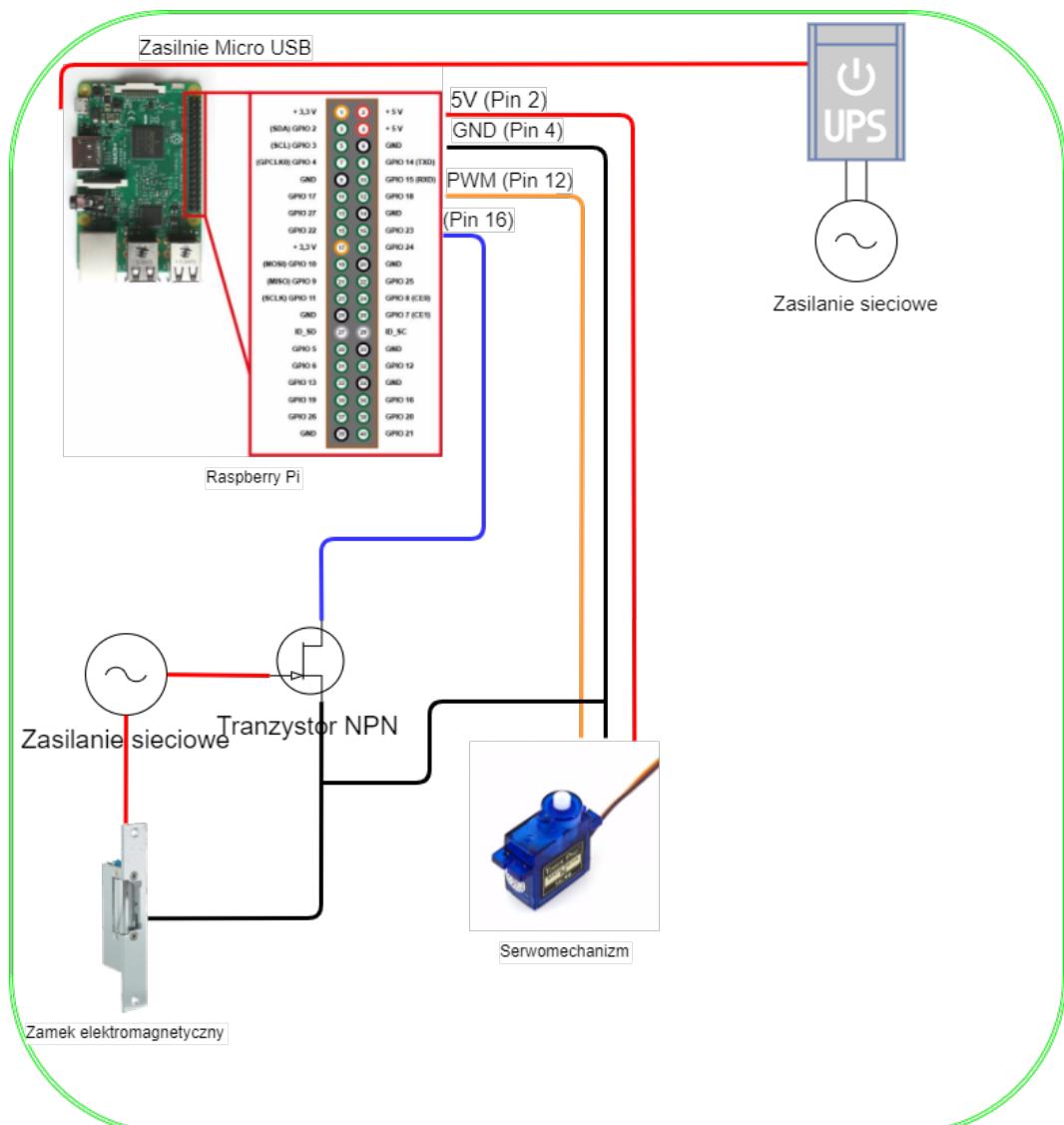


Rysunek 7.2: Diagram relacji bazy danych

Rozdział 8

Schemat elektryczny systemu

Ze względu na sprzętowy charakter projektu oraz liczne dostępne piny GPIO urządzania sterującego Raspberry, należy przedstawić schemat elektryczny systemu. Przykładowe podłączenie zgodne z przeznaczeniem pinów przedstawiono na Rysunku 8.1.



Rysunek 8.1: Schemat elektryczny systemu

Rozdział 9

Zabezpieczenia systemu

Inteligentny zamek ma na celu zabezpieczanie wejścia do pomieszczeń, dlatego sam system powinien być również bezpieczny. Podstawowymi elementami jakie należy chronić są:

- hasła,
- dane użytkowników,
- ważniejsze funkcje systemu (tj. generowanie nowych kluczy),
- transmisję danych,
- zasilanie urządzenia,
- dostępność do instalacji.

Hasła przechowywane powinny być w bazie danych w postaci skrótu funkcji SHA256. Każde hasło składać powinno się z przynajmniej 6 znaków i zawierać co najmniej jedną cyfrę.

Dane użytkowników znajdować się będą w bazie danych, do której jedynie bezpośredni dostęp będzie z aplikacji serwera lub z konta administratora. Hasło dostępu generowane powinny być za pomocą generatora liczb pseudolosowych i być o długości co najmniej 400bitów oraz przechodzić podstawowe testy losowości.

Zabezpieczeniem ważniejszych (wrażliwych) funkcji systemu odbywać się będzie po stronie aplikacji serwerowej. Serwer będzie zezwalał tylko na operacje dozwolone dla danego typu użytkownika. Do wrażliwych funkcji należy przede wszystkim pobieranie danych z bazy danych oraz ich modyfikacje. Inną formą zabezpieczenia operacji będzie podawanie hasła, mimo iż użytkownik będzie zalogowany. Weryfikacja hasła ma na celu ochronę przed chwilowym przejęciem telefonu i wygenerowaniem np. sobie dostępu do zamka.

Każde połączenie internetowe pomiędzy mikrokomputerem Raspberry Pi i serwerem będzie zabezpieczone protokołem IPSec. Komunikacja bluetooth nie będzie szyfrowana, lecz certyfikaty przesyłane będą podpisane cyfrowo kluczem asymetrycznym RSA oraz aplikacje posiadać będą klucze programowe, które muszą być identyczne po jednej i po drugiej stronie połączenia bluetooth. Pary kluczy (prywatny i publiczny) będą przypisane do danego użytkownika.

Zabezpieczeniem fizycznym systemu przed utratą zasilania z sieci energetycznej, będzie zamontowanie równolegle zasilania baterijnego załączanego w momencie zaniku głównego. Ochroną systemu przed ingerencją w sprzęt będzie umieszczenie zabudowanego mikrokomputera po wewnętrznej stronie pomieszczenia, tak żeby dostęp miały tylko osoby które miały pozwolenie przejść przez drzwi.

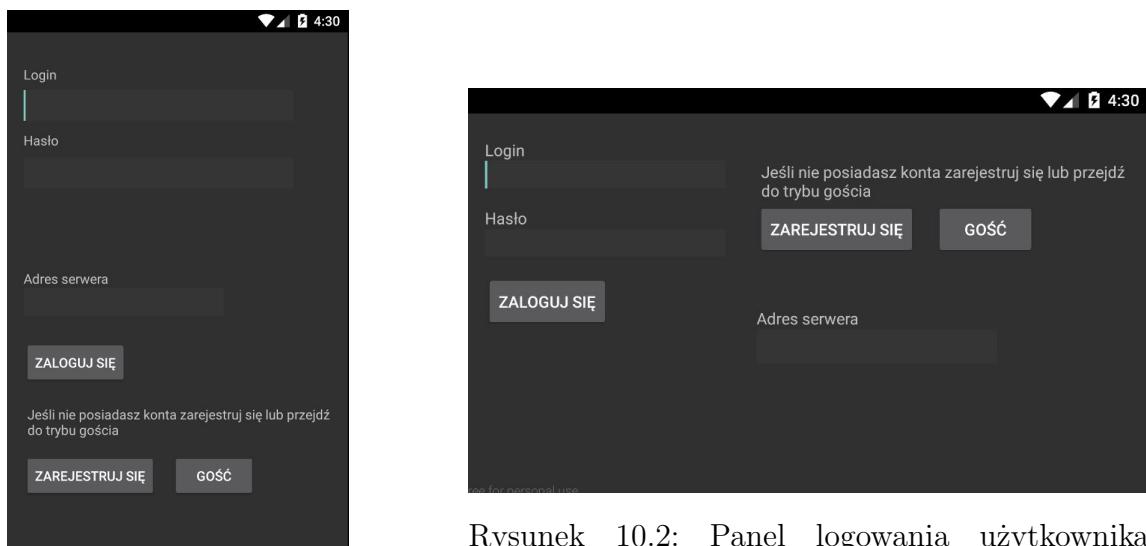
Rozdział 10

Widok graficzny systemu

Jednym elementem graficznym systemu Inteligentnego Zamka jest aplikacja znajdująca się na urządzeniach mobilnych. Poniżej opisano, krótko poszczególne widoki wykonane w środowisku Android Studio.

Panel logowania użytkownika

Widok umożliwia zalogowanie się użytkownika do systemu poprzez podanie loginu, hasła oraz adres IP serwera w odpowiednie pola, a następnie kliknięcie w przycisk “ZALOGUJ SIĘ”. Jeśli nie posiada się konta, można je utworzyć poprzez przycisk “ZAREJESTRUJ SIĘ” lub przejść do panelu Gościa przyciskiem “GOŚĆ”. (Rysunek 10.1 i 10.2)

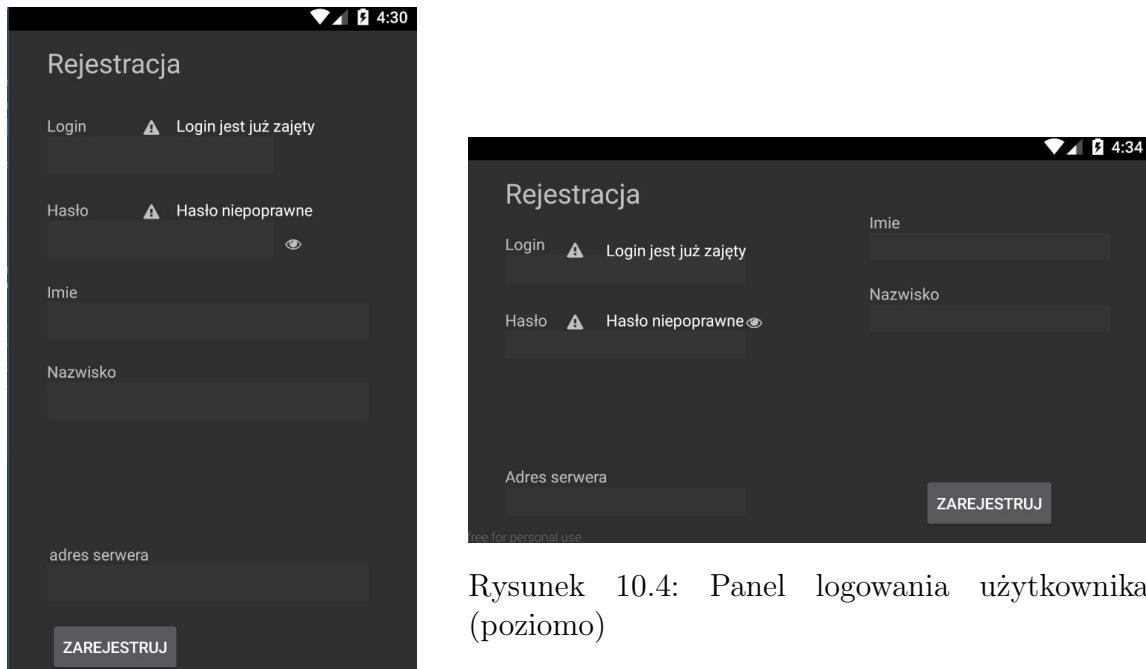


Rysunek 10.2: Panel logowania użytkownika (poziomo)

Rysunek 10.1: Panel logowania użytkownika (pionowo)

Panel rejestracji użytkownika

Panel rejestracji służy do utworzenie nowego użytkownika poprzez podanie loginu, hasła, imienia i nazwiska użytkownika oraz adresu IP serwera do którego chcemy się zarejestrować. Po upewnieniu się, że wszystkie dane są poprawne, aby zakończyć proces rejestracji, klikamy przycisk “ZAREJESTRUJ”. (Rysunek 10.3 i 10.4)

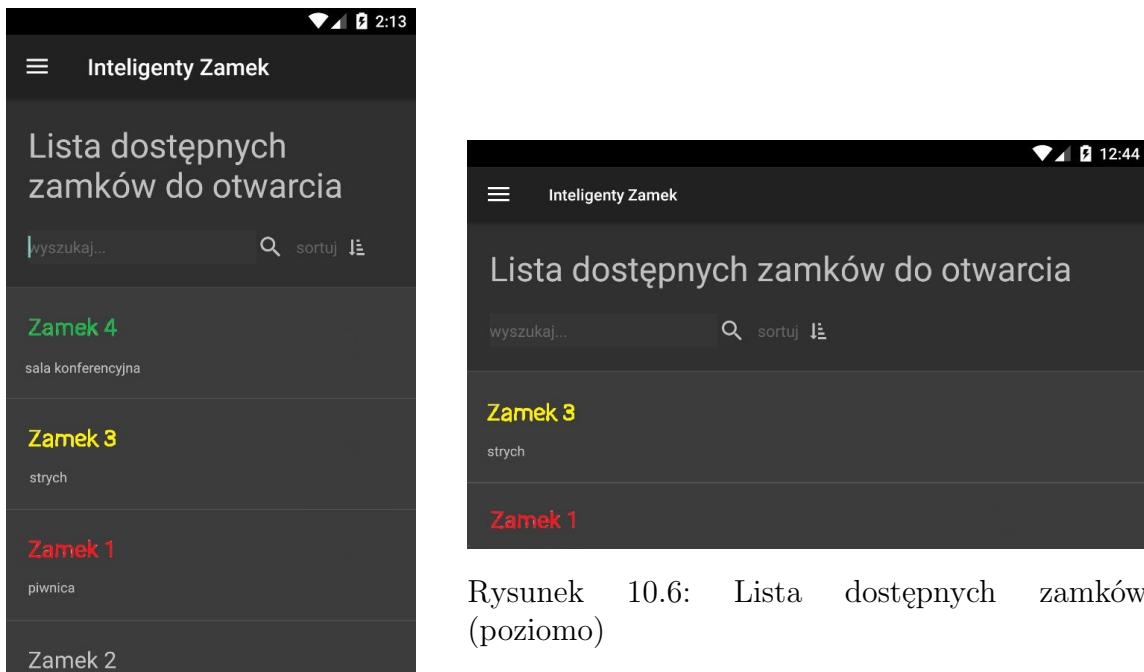


Rysunek 10.3: Panel logowania użytkownika (pionowo)

Rysunek 10.4: Panel logowania użytkownika (poziomo)

Panel listy zamków

Widok listy dostępnych zamków przedstawia listę nazw zamków do jakich dany użytkownik ma dostęp. Ułatwieniem jest możliwość sortowania wyników i wyszukiwanie po nazwach. Kliknięcie w nazwę zamka powoduje otwarcie zamka. Ustawić można również zamek, który ma być otwierany automatycznie gdy jest się w pobliżu zamka. Zmiana koloru nazw zamków sygnalizować ma status zamka. Zielona napis oznacza zamek otwarty, żółty w trakcie weryfikacji, biały zamkniętą, zaś czerwony odmowę dostępu. (Rysunek 10.5 i 10.6)

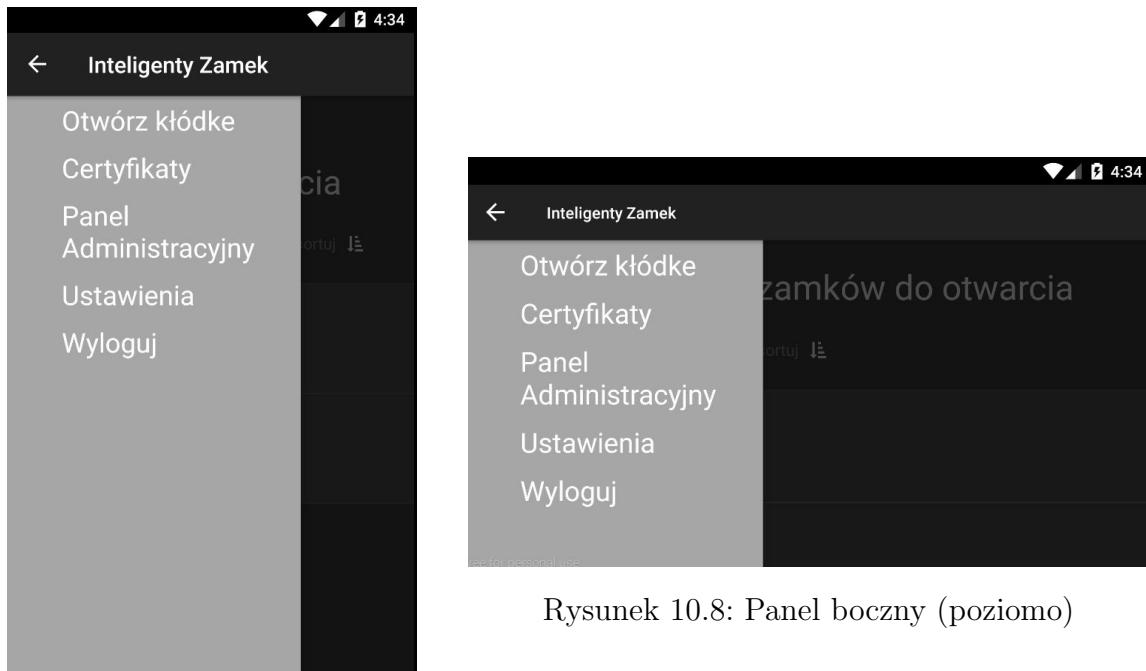


Rysunek 10.5: Lista dostępnych zamków (pionowo)

Rysunek 10.6: Lista dostępnych zamków (poziomo)

Panel boczny

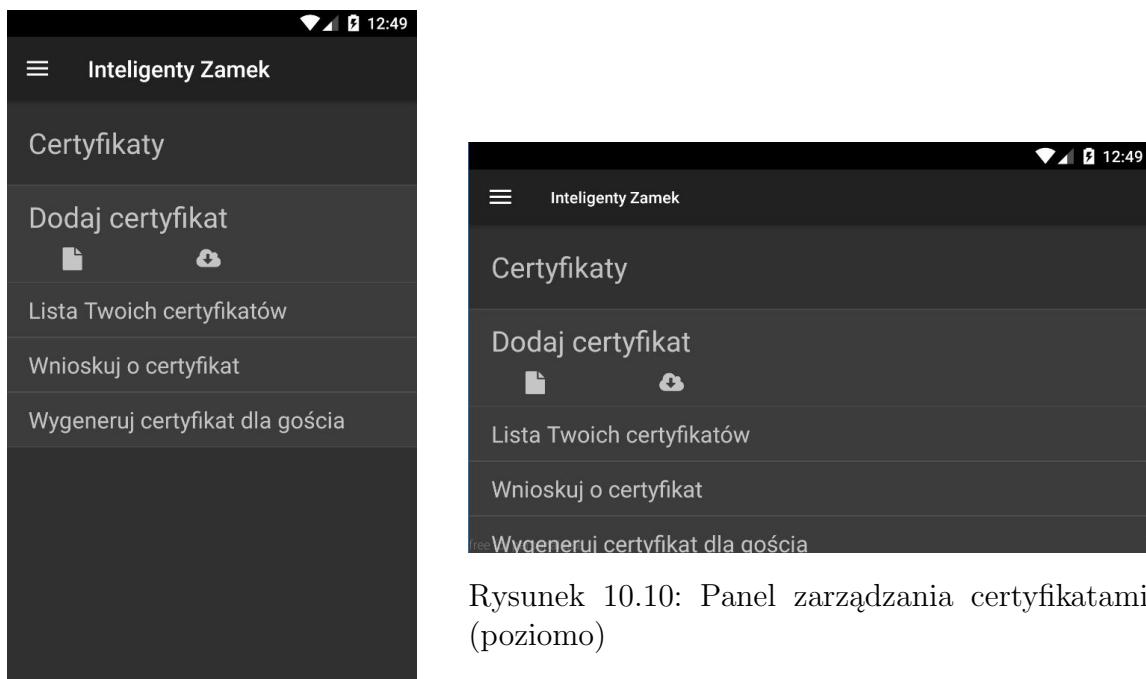
Panel boczny pozwala na szybkie przełączanie pomiędzy widokami. Chowany jest po lewej stronie ekranu. Umożliwia przechodzenie odpowiednio do listy zamków, zarządzania certyfikatami, panelu administracyjnego oraz ustawień. Ostatnia pozycja powoduje wylogowanie z aplikacji. (Rysunek 10.7 i 10.8)



Rysunek 10.7: Panel boczny (pionowo)

Panel zarządzania certyfikatami

Panel zarządzania certyfikatami umożliwia wybór funkcji dodania certyfikatu w dwóch wariantach (rozwijana lista) - dodania z pliku lub ściągnięcia z serwera. Kolejne pozycje to lista posiadanych certyfikatów, wysłanie wniosku o utworzenie nowego certyfikatu oraz ostatnia opcja to wygenerowanie certyfikatu dla gościa. (Rysunek 10.9 i 10.10)

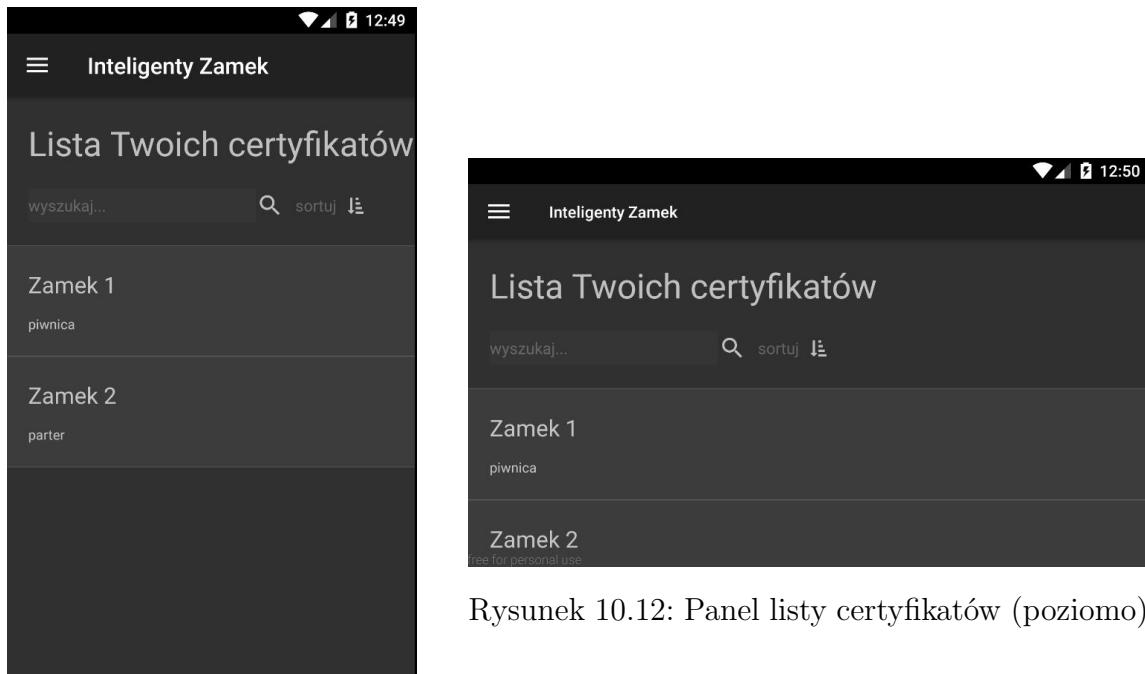


Rysunek 10.9: Panel zarządzania certyfikatami (pionowo)

Rysunek 10.10: Panel zarządzania certyfikatami (poziomo)

Panel listy certyfikatów

Panel listy certyfikatów, jest listą aktualnych certyfikatów należących do użytkownika. Kliknięcie w dany certyfikat przenosi do widoku szczegółowego związanego z operacjami na tym certyfikacie. (Rysunek 10.11 i 10.12)

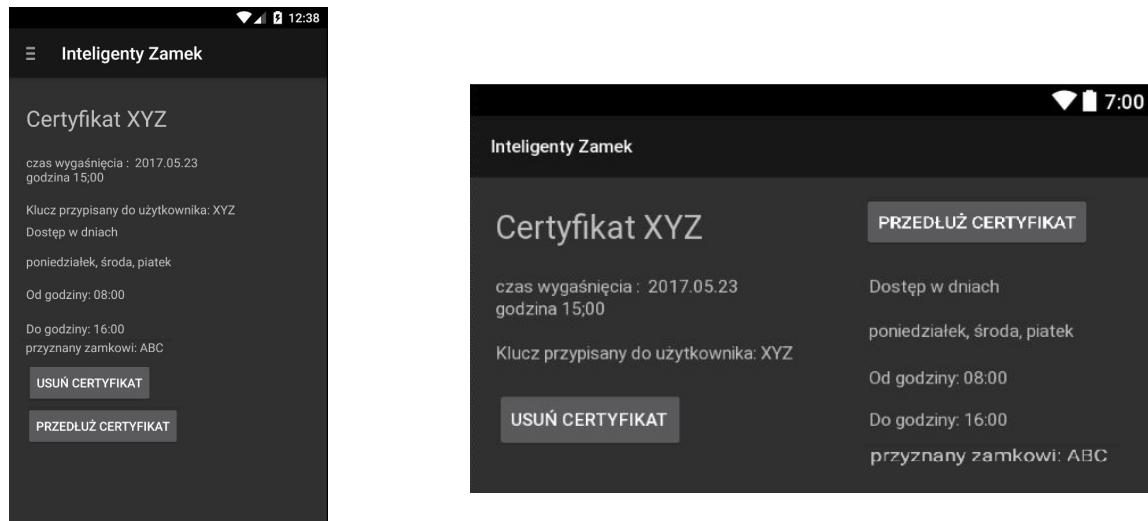


Rysunek 10.12: Panel listy certyfikatów (poziomo)

Rysunek 10.11: Panel listy certyfikatów (pionowo)

Panel certyfikatu

Panel certyfikatu u góry zawiera nazwę certyfikatu, poniżej informacje o dacie wygaśnięcia, którego zamku dotyczy oraz w jakim czasie przyznaje dostęp. Na dole dostępne są dwa przyciski pozwalające usunąć certyfikat lub wysłać prośbę o przedłużenie ważności. (Rysunek 10.13 i 10.14)

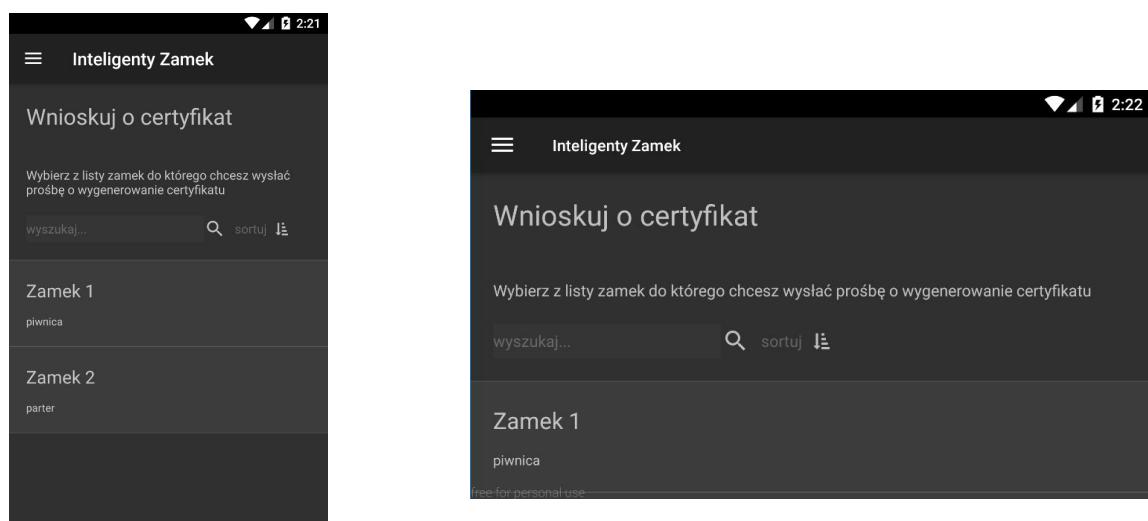


Rysunek 10.14: Panel certyfikatu (poziomo)

Rysunek 10.13: Panel certyfikatu (pionowo)

Panel wnioskowania o certyfikat

Panel wnioskowania o certyfikat polega na wybraniu z listy wszystkich zamków, konkretnego do którego chcemy uzyskać dostęp i wysłać wniosek o przydzielanie dostępu. (Rysunek 10.15 i 10.16)

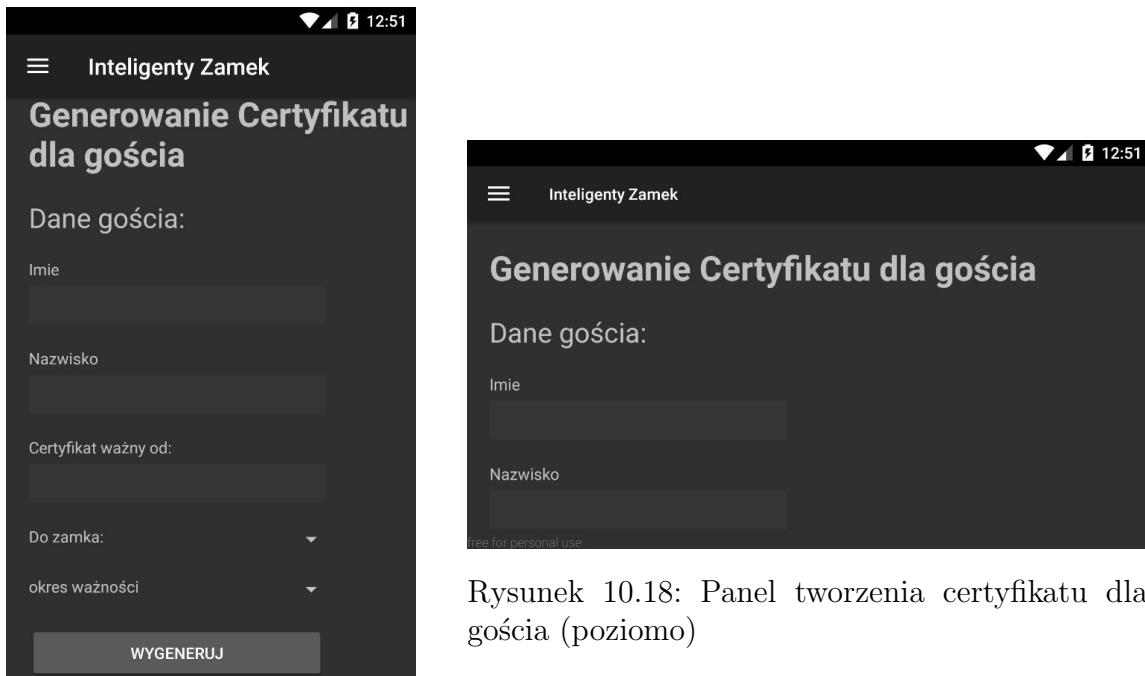


Rysunek 10.16: Panel wnioskowania o certyfikat (poziomo)

Rysunek 10.15:
Panel wnioskowania o certyfikat
(pionowo)

Panel tworzenia certyfikatu dla gościa

Panel tworzenia certyfikatu gościa składa się z danych gościa (imię i nazwisko), daty, od której obowiązuje certyfikat, zamka którego dotyczy oraz jak długo pozostanie ważny (wybór z rozwijanej listy). Aby zakończyć proces generowania należy kliknąć przycisk “WYGENERUJ”. (Rysunek 10.17 i 10.18)



Rysunek 10.17: Panel tworzenia certyfikatu dla gościa (pionowo)

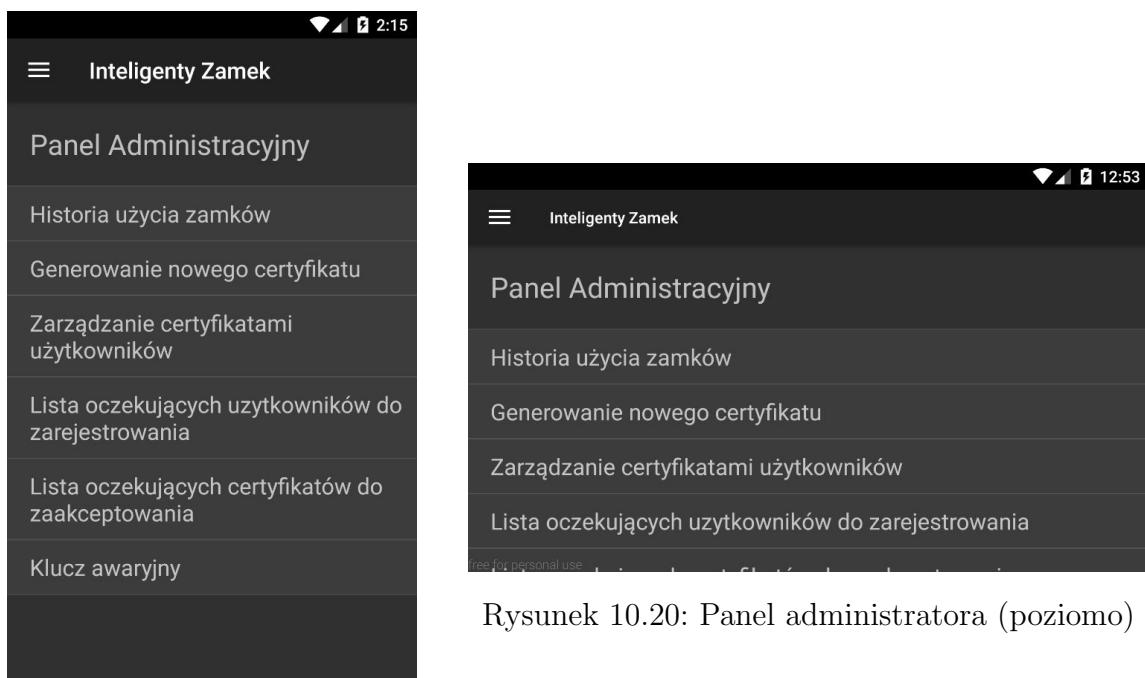
Rysunek 10.18: Panel tworzenia certyfikatu dla gościa (poziomo)

Panel administratora

W panelu administratora znajdują się 6 przycisków do administrowania systemem zamków:

- „Historia użycia zamków”
- „Generowanie nowego certyfikatu” ,
- „Zarządzanie certyfikatami użytkowników” ,
- „Lista oczekujących użytkowników do zarejestrowania” ,
- „Lista oczekujących certyfikatów do zaakceptowania” ,
- „Klucz awaryjny” .

Po kliknięciu każdego przycisku przechodzi się do nowego odpowiadającego widoku. (Rysunek 10.19 i 10.20)

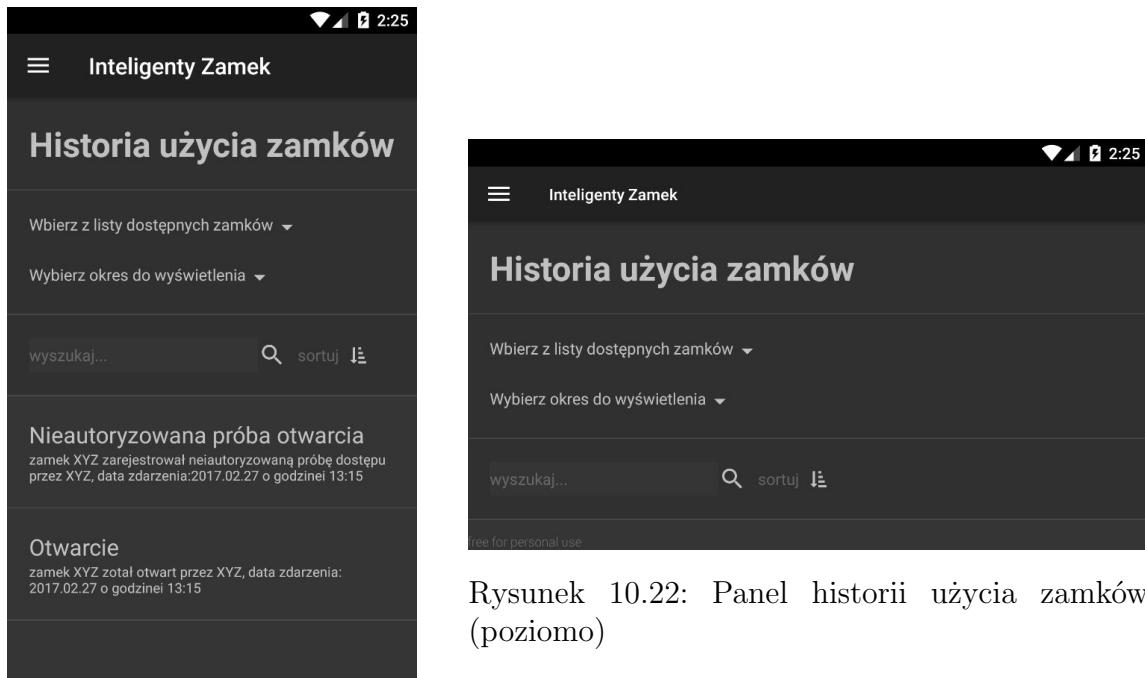


Rysunek 10.19: Panel administratora (pionowo)

Rysunek 10.20: Panel administratora (poziomo)

Panel historii użycia zamków

Panel historii użycia zamków składa się z rozwijanej listy wszystkich zamków znajdujących się w systemie, służącej do filtrowania wyników. Dodatkowo wybrać można okres z jakiego wyniki są prezentowane. Pozycje opisane kolorem czerwonym dotyczą operacji odrzucanych, białe przyznających dostęp do zamka. (Rysunek 10.21 i 10.22)

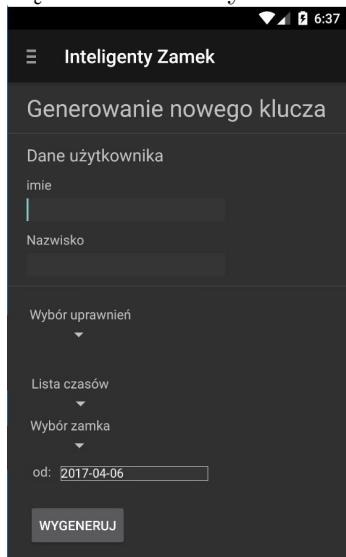


Rysunek 10.22: Panel historii użycia zamków (poziomo)

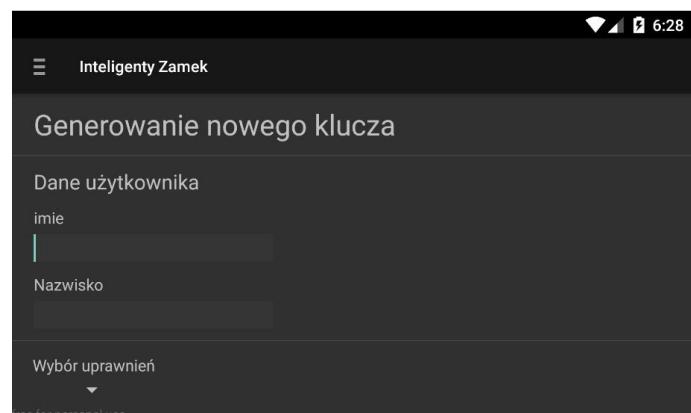
Rysunek 10.21: Panel historii użycia zamków (pionowo)

Panel generowania nowego certyfikatu (administrator)

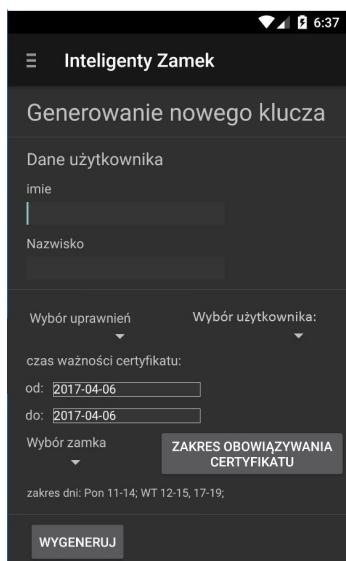
Panel generowanie nowego certyfikatu (administrator) służy do tworzenia nowych certyfikatów przez administratora. W pierwszych polach podaje się imię i nazwisko kogo dotyczy certyfikat. W następnych administrator wybiera typ certyfikatu (gość / użytkownik zalogowany). W zależności od wyboru dostępne są określone pola. Wspólne to od kiedy jest dostęp i jakiego zamka dotyczy. Dla gościa należy wybrać na jak długo jest dostęp, a dla zalogowanego użytkownika do kiedy jest ważność i w jakich godzinach danych dni tygodni ma dostęp (po kliknięciu przycisku "ZAKRES OBOWIĄZYWANIA CERTYFIKATU" przechodzi się do widoku wyboru zakresu godziny). (Rysunek 10.23, 10.24, 10.25 i 10.26)



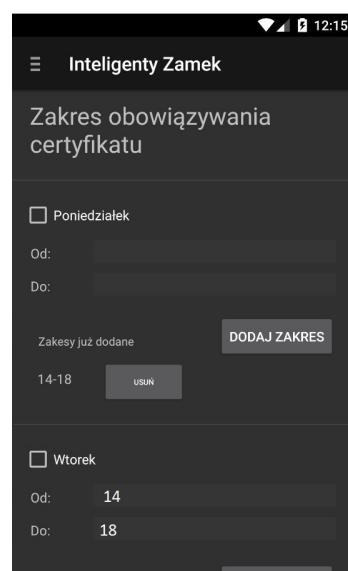
Rysunek 10.23: Panel generowania nowego klucza dla gościa (administrator) (pionowo)



Rysunek 10.24: Panel generowania nowego klucza (administrator)(poziomo)



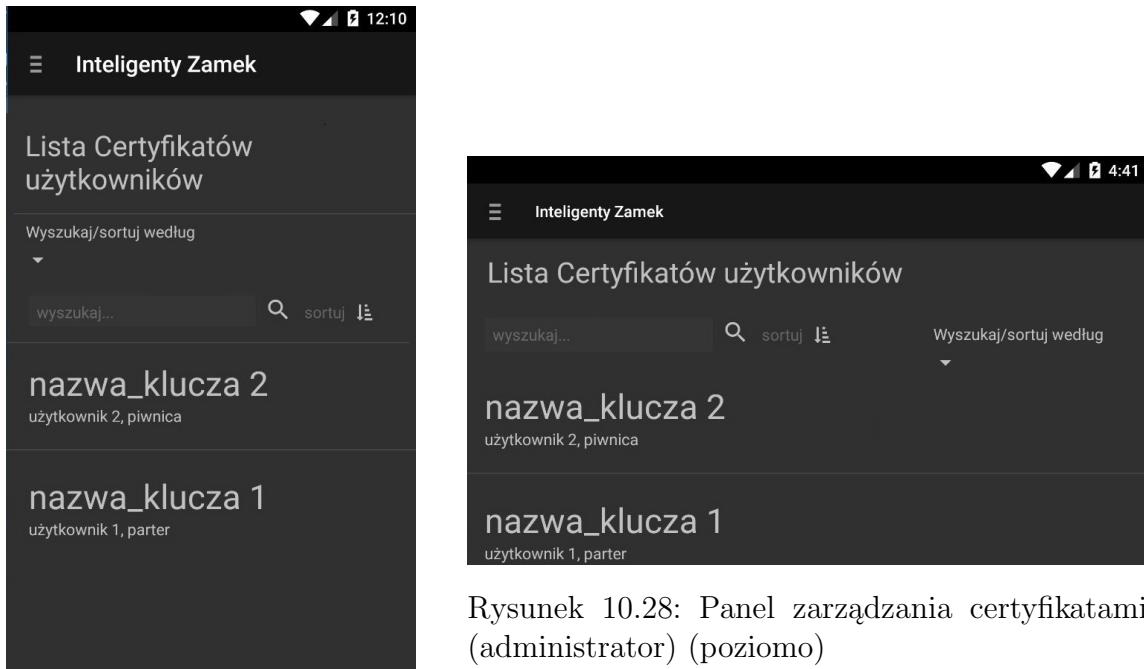
Rysunek 10.25: Panel generowania nowego klucza dla użytkownika zalogowanego (administrator)(poziomo)



Rysunek 10.26: Panel panel wyboru zakresu certyfikatu

Panel zarządzania certyfikatami (administrator)

Panel zarządzania certyfikatami użytkowników (administrator) jest widokiem tylko wszystkich aktywnych certyfikatów w systemie. Administrator klikając na pozycję przechodzi do panelu certyfikatu opisanego wyżej. Tam może usunąć dostęp lub go przedłużyć. Ułatwieniem jest możliwość wyboru typu sortowania. (Rysunek 10.27 i 10.28)

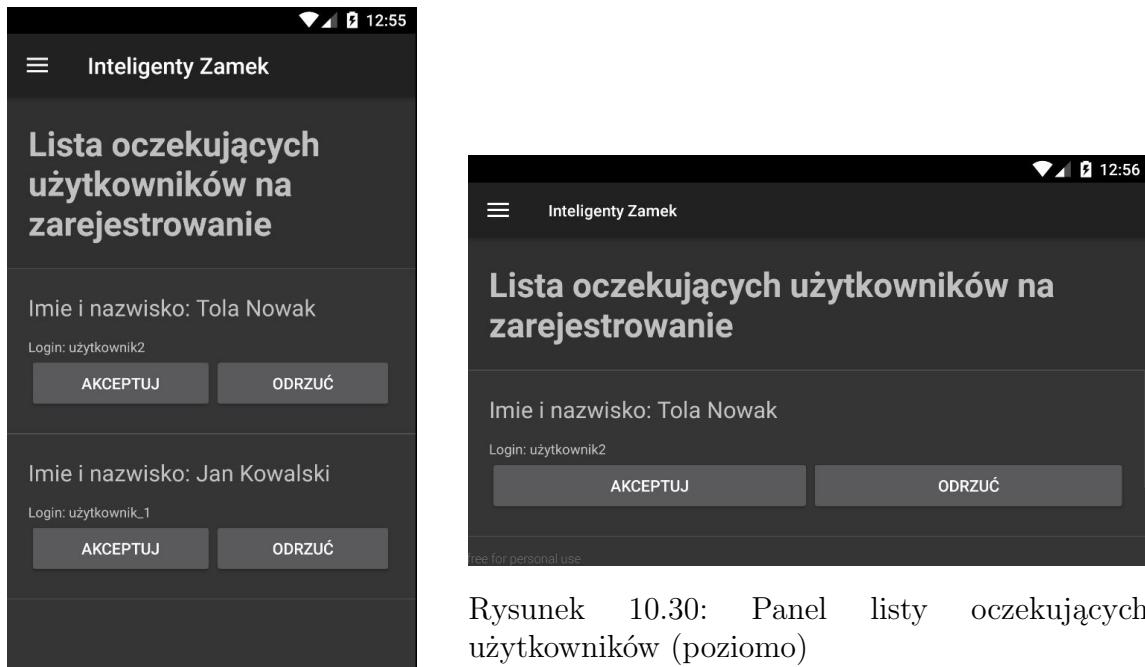


Rysunek 10.27: Panel zarządzania certyfikatami (administrator) (pionowo)

Rysunek 10.28: Panel zarządzania certyfikatami (administrator) (poziomo)

Panel listy oczekujących użytkowników do rejestracji

Panel listy oczekujących użytkowników jest listą wszystkich gości, którzy ubiegają się o zarejestrowanie. PO kliknięciu w odpowiednią pozycję pojawiają się dwie opcje: “AKCEPTUJ” lub “ODRZUĆ”. (Rysunek 10.29 i 10.30)

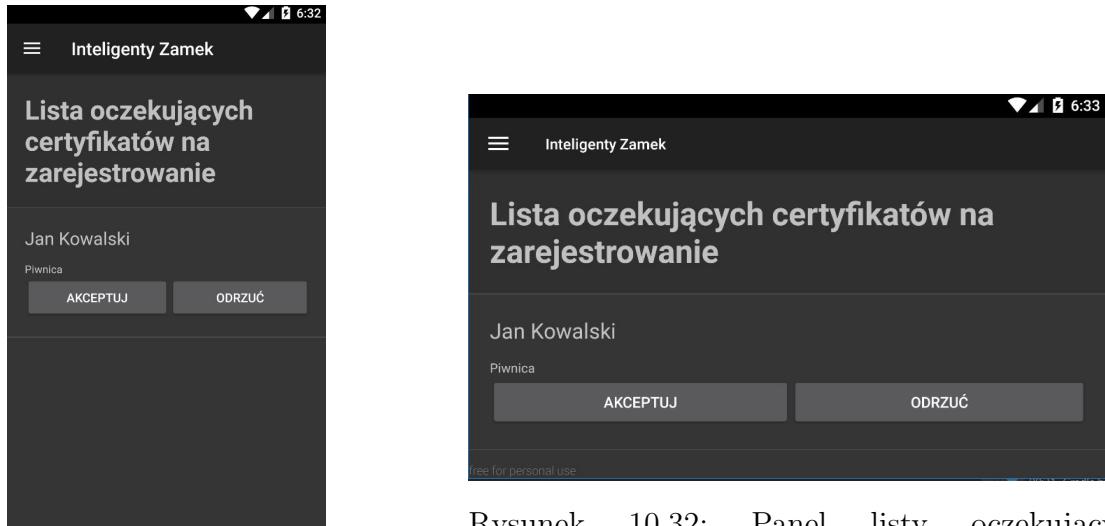


Rysunek 10.29: Panel listy oczekujących użytkowników (pionowo)

Rysunek 10.30: Panel listy oczekujących użytkowników (poziomo)

Panel listy oczekujących certyfikatów do wygenerowania

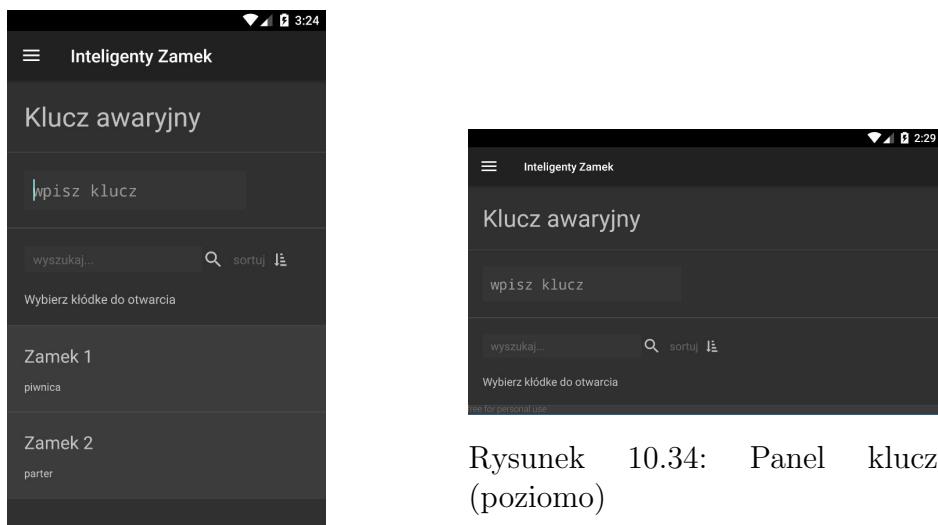
Panel listy oczekujących certyfikatów jest listą wszystkich certyfikatów, które ubiegają się o akceptację administratora. Po kliknięciu w odpowiednią pozycję pojawiają się dwie opcje: "AKCEPTUJ" lub "ODRZUĆ". (Rysunek 10.31 i 10.32)



Rysunek 10.31: Panel listy oczekujących certyfikatów (pionowo)
Rysunek 10.32: Panel listy oczekujących certyfikatów (poziomo)

Panel klucza awaryjnego

Panel klucza awaryjnego służy do otwierania zamków w sytuacji awaryjnej, np. gdy zamek nie ma dostępu do Internetu lub innej awarii. (Rysunek 10.33 i 10.34)

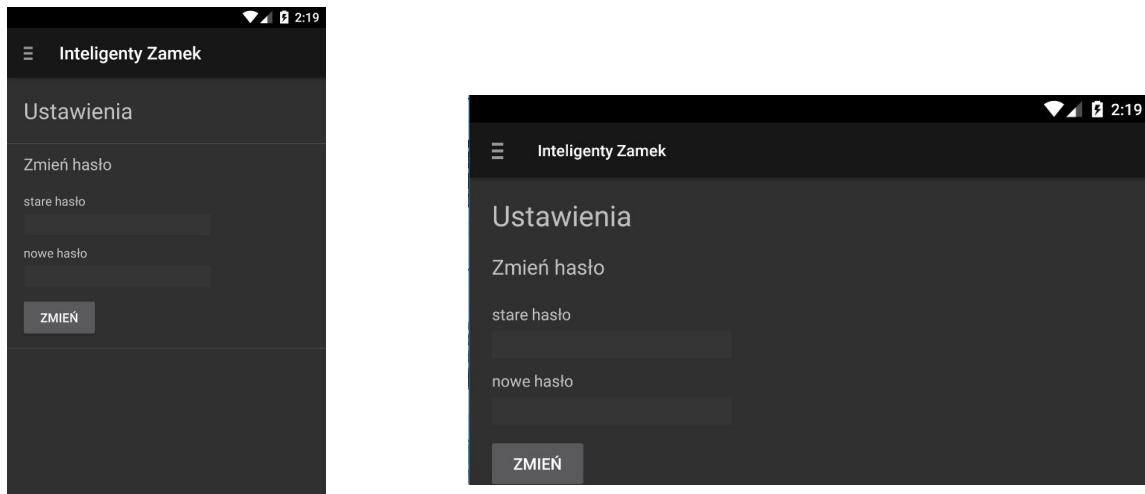


Rysunek 10.33: Panel klucza awaryjnego (pionowo)

Rysunek 10.34: Panel klucza awaryjnego (poziomo)

Panel ustawień konta

W panelu ustawień użytkownik może zmienić hasło do swojego konta. Wymagane jest podanie starego hasła, a następnie nowego. Aby zakończyć czynność klikamy na “ZMIEN”. (Rysunek 10.35 i 10.36)



Rysunek 10.36: Panel ustawień konta (poziomo)

Rysunek 10.35: Panel ustawień konta (pionowo)

Rozdział 11

Testowanie aplikacji

Główne zagadnienia które należy przetestować:

- generator liczb pseudolosowych,
- zasilanie urządzeń,
- wydajność systemu,
- komunikatywność aplikacji mobilnej (człowiek-system oraz system-człowiek),
- główne funkcje systemu podczas operacji otwierania/zamykania zamków oraz dystrybucji kluczy,

Generator liczb pseudolosowych powinien być przetestowany pod kontem testów losowości, takie jak test pojedynczych bitów, serii, długiej serii, pokerowy oraz dwójkowy. Próby wykonywane powinny być wielokrotnie, dla wielu utworzonych próbek.

System oparty jest o poprawne działanie sprzętu, należy więc przetestować reakcję urządzeń na spadki napięcia podczas spoczynku, pracy zwykłej oraz przy dużym obciążeniu. Raspberry Pi w razie awarii zasilania sieciowego, przełączy się na zasilanie baterijne. Poprzez częste przełączanie sposobu zasiania, np. przy użyciu tranzystora należy sprawdzić, czy w trakcie przejścia urządzenia nie wyłączy się z powodu krótkiego całkowitego zaniku napięcia. Test zasilania zweryfikować powinien również czas działania urządzeń pod zasilaniem baterijnym.

Testem wydajności systemu jest sprawdzenie odporności na przeciążenia systemu, choćby przy wielu żądaniach do serwera. System zakłada obsługę wielu zamków przy jednym serwerze, dlatego należy zweryfikować działanie bazy danych oraz aplikacji serwerowej pod kątem wielu zapytań jednocześnie. Wykonać taki test można przy pomocy aplikacji Apache JMeter. Program zbada serwer pod kątem szybkości oraz odpowiedzi httprequest.

Jedynym elementem kontaktującym się z użytkownikiem jest aplikacja mobilna. Program powinien w pierwszej fazie testów zostać sprawdzony automatycznie przy pomocy narzędzia Robotium, pozwalającym na utworzeniu sekwencji operacji jakie mają po sobie nastąpić, aby osiągnąć zamierzony efekt. Scenariusz testu obejmować powinien reakcję na:

- podanie błędnych kluczy dostępowych podczas procesu otwierania zamka,
- zlego lub przedawnionego klucza prywatnego,
- braku połączenia z internetem,
- braku połączenia bluetooth,

- błędnego logowania użytkownika,
- niewłaściwych danych przy rejestracji użytkownika.

Po przejściu aplikacji przez testy automatyczne, należy przekazać aplikację użytkownikom, w pierwszej kolejności posiadającym wiedzę informatyczną oraz znającym założenia systemu. Użytkownicy wyrażają opinie na temat działania, czytelności aplikacji podczas ankiety. Następnym krokiem jest testowanie aplikacji przez osoby nie posiadające wiedzy informatycznej, lecz zaznajomione z tematyką systemu. Ostatnim krokiem testów aplikacji są testy przez użytkowników bez wiedzy informatycznej oraz bez szczegółowej wiedzy o tematyce projektu. Każdy etap musi uzyskać pewien próg pozytywnych ocen, przy braku zdania testu należy wrócić do początku. Ankieta zawierać powinna pytania na temat estetyki aplikacji, komunikatywności z użytkownikiem.

Testowanie głównej funkcji systemu, tzn. otwierania/zamykania zamka oraz dystrybucji kluczy dostępowych zrealizowane zostanie podczas wcześniej wymienionych testów. Dodatkowo, również przy użyciu narzędzia Robotium wykonane zostaną scenariusze przykładowych użyć systemu. Przykładowy scenariusz:

Użytkownik zalogowany (UZ) wnioskuje o klucz dostępowy do zamka A, B i C na okres tygodnia. Administrator (AD) akceptuje wniosek dla zamka B i C, dla zamka A odrzuca. UZ tworzy klucz gościa do zamka B jednorazowy (zał. powodzenie). UZ próbuje otworzyć zamek B (zał. powodzenie). AD blokuje dostęp UZ do zamka B. Gość próbuje otworzyć zamek B (zał. powodzenie). UZ próbuje otworzyć ponownie zamek B (zał. niepowodzenie). Gość próbuje otworzyć zamek B (zał. niepowodzenie). UZ próbuje utworzyć klucz dostępu dla gościa do zamka B na 10 minut (zał. niepowodzenie). UZ próbuje utworzyć klucz dostępu dla gościa do zamka C na 10 minut (zał. powodzenie). Gość próbuje otworzyć zamek C (zał. powodzenie). Gość próbuje ponownie otworzyć zamek C po upływie 10 minut (zał. niepowodzenie). UZ próbuje otworzyć zamek C po upływie tygodnia od utworzenia klucza dostępu (zał. niepowodzenie).

Każda operacja wykonana w scenariuszy będzie mierzona ile czasu zajmuje, pozwoli to na określenie czasu oczekiwania aplikacji na odpowiedzi, w celu optymalizacji działania.

Rozdział 12

Możliwe zagrożenia systemu

Głównym celem ataków na system Inteligentnego Zamka może być uzyskanie klucza prywatnego użytkownika. Umożliwi to podszywanie się pod inną osobę, próbując otrzymać dostęp do nowych zamków. Ograniczeniem tej możliwości jest decyzja administratora o przydzielenie pozwolenia oraz jeśli właściciel spostrzeże się o naruszeniu, to może zablokować konto.

System nie przewiduje fizycznych zabezpieczeń urządzenia sterującego, zatem mając bezpośredni kontakt z Raspberry można wówczas w łatwy sposób wywołać otwarcie zamka.

Aplikacja mobilna mimo zabezpieczenia hasłem oraz ponownego uwierzytelniania przy generowaniu nowych kluczy, wciąż jest narażona na przejęcie sesji lub urządzenia użytkownika.

System wymaga w dużej mierze dostępu do Internetu, przy braku połączenia prawidłowe działanie jest praktycznie niemożliwe. Tylko administrator jest w stanie otworzyć zamek w takiej sytuacji.

Rozdział 13

Dalsze perspektywy rozwoju projektu

Projekt rozwinać można o zabezpieczenia wizyjne, to znaczy podłączyć kamerę komunikującą się z Raspberry, która przy nieudanej próbie otwarcia zamka wykonywała by zdjęcie. W ten sposób każda próba naruszanie bezpieczeństwa byłaby udokumentowana.

Innym pomysłem rozwoju systemu, również z użyciem kamer, może być monitorowanie ruchu wchodzącego jak i wychodzącego na podstawie mechanizmu wykrywania twarzy. Pozwoli to zabezpieczyć system na sytuacje, że pozostaną w pomieszczeniu / budynku osoby po godzinach urzędowania. Bilansu ruchu użyć można także w sytuacjach awaryjnych, kiedy ważna jest informacja o liczbie osób znajdujących się w pomieszczeniu.

Wykrywanie twarzy mogłoby również posłużyć jako alternatywa dla połączenia bluetooth. Dodając algorytmy rozpoznajace twarzy można w ten sposób decydować czy dana osoba ma aktualnie dostęp czy nie.

System odpowiednio zabezpieczony mógłby być zarządzany nie tylko z poziomu aplikacji telefonu, a również z strony internetowej.

Sposobem na uatrakcyjnienie systemu jest przygotowanie różnych wersji urządzeń sterujących w zależności od stopnia zaawansowania. Mikrokomputer Raspberry Pi jest stosunkowo drogim narzędziem, można zastąpić go słabszym kontrolerem z ograniczonymi funkcjami. Poszerza to elastyczność systemu w zależności od zapotrzebowania i ceny produktu.

Rozdział 14

Podział prac projektowych

Zadania wykonane przez **Macieja Marciniak**:

- pełne oprogramowanie Raspberry Pi,
- aplikacja mobilna:
 - parowanie urządzeń bluetooth,
 - transmisja danych poprzez bluetooth,
 - uruchamianie i sprawdzanie aktywności modułu bluetooth,
 - konsultacje projektowe widoków aplikacji,
- serwer systemu:
 - tworzenie API serwerowego,
 - tworzenie bazy danych,

Zadania wykonane przez **Damiana Filipowicz**:

- aplikacja mobilna — cała aplikacja, za wyjątkiem obsługi bluetooth,
- serwer systemu:
 - tworzenie API serwerowego,
 - konfiguracja serwera.

Rozdział 15

Implementacja

Postępy prac oraz kod źródłowy znajduje się w repozytorium GitHub pod adresem:

[Inteligentny Zamek](#)

Rozdział 16

Podsumowanie projektu

Projekt nie został w pełni zrealizowany według zakładanej dokumentacji. Dużym problemem okazało się generowanie certyfikatu dla gości. Tworząc dostęp trudno było utworzyć parę kluczy prywatny — publiczny, tak aby każdy użytkownik miał inny, a przy tym nie naruszał bezpieczeństwa kluczy twórcy dostępu. Został zastosowany w takim wypadku uniwersalny klucz dla gości, to znaczy każdy z nich ma taki sam klucz. Jest to zagrożenie dla systemu o tyle, że nie jest w stanie potwierdzić kto faktycznie wysłał dany certyfikat.

Nie została zrealizowana funkcja związana z kluczem awaryjnym należącym do administratora ze względu na wymagane przy tym zmiany w innych elementach systemu. Nie zostało to w pełni przemyślane na odpowiednim etapie projektowania.

Ze względów ekonomicznych nie zastosowano w projekcie alternatywnego zasilania UPS, lecz nie jest to czynność wymagająca szczególnej uwagi podczas realizacji zadania projektowego.

Podsumowując całościowo projekt, system działa prawidłowo, lecz nie posiada pełnej funkcjonalności zakładającej na początku. Podczas tworzenia projektu nauczyliśmy się przede wszystkim dokumentować system. Jest to bardzo przydatna umiejętność, pozwalająca wykryć błędy zanim przystąpi się do implementacji. Dzięki pracy nad zadaniem wzmacniliśmy umiejętności związane z pracą zespołową, mimo że zespół składał się tylko z dwóch osób, jak również mogliśmy poszerzyć znajomość na temat pracy ze sprzętem (tu Raspberry Pi), programowania aplikacji mobilnych oraz tworzenia systemów serwerowych.