|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sterowniki PLC** | | | | | | |
| Rok akademicki | Termin | Rodzaj studiów | Kierunek | Prowadzący | Grupa | Sekcja |
| 2017/2018 | *Czwartek* | SSI | Informatyka | dr inż. Michał Maćkowski | PSI | 4 |
| *16:15 – 17:45* |
| Ocena: |  | Komentarz: |  | | | |

Dokumentacja projektu

Data wykonania: *yyyy-mm-dd*

Temat projektu:

System dostępu do pomieszczeń

|  |
| --- |
| Skład sekcji: |
| Łukasz Błasiak |
| Katarzyna Urbasik |
| Jakub Czachor |
|  |
|  |

1. **Wstęp**
2. **Serwer TCP/IP – specyfikacja funkcjonalna**

Główną częścią projektu jest serwer TCP/IP obsługujący żądania od klientów, sterujący dostępem do pomieszczeń oraz komunikujący się z baza danych. Całość została zaimplementowana w języku obiektowym Java z wykorzystaniem pakietu Java.net do obsługi komunikacji sieciowej.

Do prawidłowej pracy serwera, w pierwszej kolejności należy poprawnie skonfigurować plik *config.properties* znajdujący się folderze *cfg* w głównym katalogu projektu. Zawiera on podstawowe ustawienia konfiguracyjne takie jak m.in. adres IP urządzenia na którym pracuje serwer oraz port. W przypadku braku lub niepoprawnie wypełnionego pliku konfiguracyjnego, aplikacja poinformuje o tym fakcie stosownym komunikatem.

Serwer został zaimplementowany z wykorzystaniem wielowątkowości, co pozwala na obsługę każdego klienta w osobnym wątku. Powyższe rozwiązanie skraca kolejkę oczekiwania na obsługę żądania. Dodatkowo, ze względu na niską cykliczność zapytań, na każde żądanie tworzone jest pojedyncze połączenie, zamykane zaraz po jego obsłużeniu. Redukuje to liczbę otwartych połączeń do liczby aktualnych żądań, a nie liczby uruchomionych aplikacji (które mogą być aktywne w tle).

Na potrzeby projektu zaimplementowany został protokół komunikacyjny, znajdujący się w klasie *Protocol*. Posiada on m.in. listę dozwolonych komend nadesłanych przez klienta. Odpowiada on głównie za walidację żądania (poprawność polecenia oraz odpowiednia liczba argumentów) oraz oddzielenie argumentów do odpowiedniego kontenera. W protokole zostały zaimplementowane odpowiednie gettery umożliwiające odczytanie powyższych wartości. Serwer obsługuje polecenia służące do logowania, otwierania drzwi oraz zwracania listy pomieszczeń w formacie JSON. Poniżej znajduje się tabela zawierająca opis poszczególnych poleceń:

|  |  |  |
| --- | --- | --- |
| **POLECENIE** | **ARGUMENTY** | **ZWRACANA WARTOŚĆ** |
| OPEN | <email> <hasło> <nr\_pokoju> | OPENED ROOM NR: … |
| NO PRIVILEGES TO OPEN ROOM NR: … |
| AUTHORIZATION FAILED |
| WRONG ROOM NUMBER |
| WRONG COMMAND FORMAT |
| LOGIN | <email> <hasło> | LOGGED |
| AUTHORIZATION FAILED |
| WRONG COMMAND FORMAT |
| GET\_DOORS\_LIST | <email> <hasło> | <JSON zawierający listę pomieszczeń> |
| AUTHORIZATION FAILED |
| WRONG COMMAND FORMAT |

Do poprawnej pracy wymagana jest również baza danych MySQL wraz z poprawną konfiguracją. Do obsługi komunikacji z bazą danych, wykorzystany został pakiet Java.sql oraz oficjalny MySQL connector w wersji 5.1.46. W pliku konfiguracyjnym serwera muszą zostać podane takie ustawienia jak: adres do bazy wraz z portem, nazwa użytkownika oraz hasło. W przypadku błędnej konfiguracji serwer zasygnalizuje powyższy fakt stosownym komunikatem.

W bazie znajdują się dwie tabele: *user* oraz *privilege*. Pierwsza przechowuje informacje o użytkownikach takie jak id, e-mail, hasło oraz dwie flagi: ze stanem aktywności konta oraz czy jest to konto z uprawnieniami administratora. Druga tabela przechowuje natomiast informację związane z dostępem do poszczególnych pomieszczeń: id uprawnienia, numer pokoju oraz id użytkownika, który posiada dane uprawnienie.

W serwerze został dodatkowo zaimplementowany logger archiwizujący wszystkie aktywności zachodzące po stronie serwera. Pojedynczy wpis związany z zapytaniem przyjmuje następujący format:

<data> <godzina> <email> <adres IP> <rezultat polecenia>

Logger zapisuje również połączenia i rozłączenia klienta z serwerem. Pełną historię logów można znaleźć w pliku *logs.log* znajdującym się folderze *cfg* w głównym katalogu projektu. Dodatkowo każda aktywność jest wypisywana na konsoli serwera.

Dodatkowo w kodzie źródłowym, do każdej metody oraz pola zostały zamieszczone odpowiednio sformatowane komentarze. Z wykorzystaniem narzędzia *Javadoc,* umożliwiają one generację dokumentacji w postaci strony WWW na podstawie zamieszczonych w kodzie źródłowym znaczników w komentarzach.

**X. Wnioski**