Міністерство освіти і науки України Національний університет «Львівська політехніка»

Кафедра ЕОМ



**Лабораторна робота №3**

з дисципліни: «Інженерія програмного забезпечення»

на тему: “UML діаграми варіантів використання та UML діаграми активності..”

Виконав:

ст. гр. КІ-31

Басюк А.А.

Прийняв:

Цигилик Л.О.

Львів – 2022

**Тема:** Розробка серверної частини. Розробка комунікації за протоколом TCP. Підключення серверного модуля до БД.

**Мета:** Розробити консольну аплікацію що буде підтримувати зв’язок по протоколу TCP/IP, отримувати дані та записувати у БД. Також, згідно деякої команди, вичитувати з БД необхідну інформацію та передавати по TCP протоколу на клієнтську частину.

**Постаноска задачі:**

Розробка серверної частини.

**Варіант 1** UML діаграми варіантів використання та UML діаграми активності.

**Програма серверної частини**



Рис 1. Вікно програми сервера

**Опис API функцій сервера**

Функція **LogProc**(QTcpSocket\* socket) призначена для виконання процедури входу в систему.

Функція **RegProc(**QTcpSocket\* socket**)** призначена для виконання процедури реєстрації нового користувача в системі.

Функція **GetCity(**QTcpSocket\* socket**)** призначена для отримання списку областей. Функція **GetUniver (**QTcpSocket\* socket**)** призначена для отримання списку університетів.

Функція **GetSpecialty (**QTcpSocket\* socket**)** призначена для отримання списку спеціальностей. Функція **GetCourse (**QTcpSocket\* socket**)** призначена для отримання списку курсів.

Функція **GetRating (**QTcpSocket\* socket**)** призначена для отримання рейтингового списку студентів.

Функція **GroupSearch (**QTcpSocket\* socket**)** призначена для отримання рейтингового списку студентів у потрібній користувачу групі.

Функція **NameSearch (**QTcpSocket\* socket**)** призначена для отримання рейтингового списку студентів з потрібним користувачу ім’ям.



Рис 2. База даних для областей

Рис 3. База даних університетів



Рис 4. База даних для рейтингових списків

Рис 5. База даних зареєстрованих користувачів

**Приклад коду для запису у базу даних**

void Server::**RegProc**(QTcpSocket \*socket )

{

*// find log in database*

query->prepare("select \* from UserInfo where UserLog = :log"); query->bindValue(":log", obj->value("log").toString());

*if* (query->exec())

{

*// if qry is not empty (if log exist) if* (query->next())

{

*// send user already exist password respond to client* Data="{\"operation\":\"register\", \"resp\":\"bad\", \"err\":\"User already exist\"}"; DataSend(socket, Data);

socket->*waitForBytesWritten*(1500);

}

*// qry is empty so login does not exist else*

{

*// insert login and password to database*

query->prepare("insert into UserInfo (UserLog, UserPass) values (:log, :pass)"); query->bindValue(":log", obj->value("log").toString());

query->bindValue(":pass", obj->value("pass").toString());

*// if registration successful if* (query->exec())

{

*// send good respond* Data="{\"operation\":\"register\", \"resp\":\"ok\"}"; DataSend(socket, Data);

socket->*waitForBytesWritten*(1500);

}

*else*

{

*// set smt went wrong respond*

Data="{\"operation\":\"register\", \"resp\":\"bad\", \"err\":\"Something went wrong when transfering data. Please check your internet connection\"}"; DataSend(socket, Data);

socket->*waitForBytesWritten*(1500);

}

}

}

*else*

{

*// handle bad query execution*

Data="{\"operation\":\"register\", \"resp\":\"bad\", \"err\":\"Something went wrong when transfering data. Please check your internet connection\"}"; DataSend(socket, Data);

socket->*waitForBytesWritten*(1500);

}

}

**Приклад коду для зчитування з бази даних**

void Server::**LogProc**(QTcpSocket\* socket)

{

*// find log in database*

query->prepare("select \* from UserInfo where UserLog = :log"); query->bindValue(":log", obj->value("log").toString());

*// if qry is not empty (if log exist) if* (query->exec())

{

*if* (query->next())

{

*// if password is correct*

*if* (query->value(1).toString() == obj->value("pass").toString())

{

*// return validation is ok respond to client* Data="{\"operation\":\"login\", \"resp\":\"ok\"}"; DataSend(socket, Data);

socket->*waitForBytesWritten*(1500);

}

*// if password is not correct else*

{

*// send invalid password respond to client*

Data="{\"operation\":\"login\", \"resp\":\"bad\", \"err\":\"Invalid password\"}"; DataSend(socket, Data);

socket->*waitForBytesWritten*(1500);

}

}

*// qry is empty so login does not exist else*

{

*// send no such login respond to client*

Data="{\"operation\":\"login\", \"resp\":\"bad\", \"err\":\"Login doesn't exist\"}"; DataSend(socket, Data);

socket->*waitForBytesWritten*(1500);

}

}

*else*

{

*// handle bad query execution*

Data="{\"operation\":\"login\", \"resp\":\"bad\", \"err\":\"Something went wrong when transfering data. Please check your internet connection\"}"; DataSend(socket, Data);

socket->*waitForBytesWritten*(1500);

}

}

**Приклад пакету даних що передається**

Усі дані передаються у форматі JSON.Це спрощує їх розшифрування, полегшує процес виявлення помилок та збільшує надійність передачі даних.

**Висновок**

Під час виконання цієї лабораторної роботи я створив програму-сервер, яка обробляє та виконує запити клієнтської частини шляхом взаємодії з базою даних.