**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОСТРОЗЬКА АКАДЕМІЯ»**

**ЕКОНОМІЧНИЙ ФАКУЛЬТЕТ**

**КАФЕДРА ЕКОНОМІКО-МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ ТА**

**ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Довгострокове завдання**

з предмету «Основи програмування»

на тему: «Месенджер You.NET»

Підготував:

студент групи ЕК-21

Чухілевич Василь Вікторович

Викладач:

Клебан Ю.В.

**Острог – 2020 р.**

# Опис задачі

**Завдання.** Створення месенджера, придатного до повноцінного листування; можливості використовувати емоджі у переписці, видаляти повідомлення. У роботі з чатами надати користувачу можливість створювати приватні, групові чати; можливість перейменовувати їх для себе та для всіх учасників, покидати окремі чати. Реалізувати можливості реєстрації та авторизації користувача. Повідомлення, що були надіслані у наступну добу, повинні мати у заголовку дату відправки.

# Файловий вміст проєкта

|  |  |
| --- | --- |
| **Forms**  **Користувацькі форми**   1. AddChat 2. Auth 3. Client   **Форми для введення**   1. GropNameEditInput 2. InputForm   **Classes**   1. DB 2. Program | **User controls**  **Для повідомлень**   1. Messages 2. ChatInfo 3. Inbox 4. Outbox   **Інші**   1. Chat 2. Users   **MySQL Server** |

# Forms

## Користувацькі форми

### AddChat

Дана форма призначена для створення нового чату(групового або приватного) та задання кола його учасників, назви.

**Код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace msg

{

public partial class AddChat : Form

{

List<int> selectedUsers = new List<int>();

private DB db;

public AddChat(DB db)

{

this.db = db;

InitializeComponent();

LoadUsers();

selectedUsers.Add(db.UserId);

}

private void LoadUsers()

{

Users[] users = db.LoadUsers(search.Text, selectedUsers.ToArray());

usersFlow.Controls.AddRange(users);

foreach (var c in users)

c.add.Click += AddUser;

}

private void search\_TextChanged(object sender, EventArgs e)

{

usersFlow.Controls.Clear();

LoadUsers();

}

private void AddUser(object sender, EventArgs e)

{

var btn = (Button)sender;

if(btn.Text == "+")

{

selectedUsers.Add((int)btn.Tag);

btn.Text = "-";

}

else

{

selectedUsers.Remove((int)btn.Tag);

btn.Text = "+";

}

}

private void cancel\_Click(object sender, EventArgs e)

{

this.Close();

}

private void create\_Click(object sender, EventArgs e)

{

if(selectedUsers.Count == 1)

MessageBox.Show("Виберіть принаймні одного учасника чату", "You.NET messager", MessageBoxButtons.OK, MessageBoxIcon.Information);

else if(selectedUsers.Count == 2 )

{

if(db.CreatePrivateChat(selectedUsers[1]))

{

DialogResult = DialogResult.OK;

this.Close();

}

else

MessageBox.Show("Помилка з'єднання", "You.NET messager", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

var f = new InputForm("Enter group chat name", "create", "");

f.ShowDialog();

if(f.DialogResult == DialogResult.OK)

{

if (db.CreatePublicChat(selectedUsers.ToArray(), f.input.Text))

{

DialogResult = DialogResult.OK;

this.Close();

}

else

MessageBox.Show("Помилка з'єднання", "You.NET messager", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

}

}

}

### Auth

Дана форма призначена авторизції користувача. Цю операцію може бути здійснено як вхід уже існуючого користувача або реєстрацію нового.

**Код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace msg

{

public partial class Auth : Form

{

private DB db = new DB();

public Auth()

{

InitializeComponent();

}

private void loginBtn\_Click(object sender, EventArgs e)

{

if (CheckFill())

{

if (db.CheckIfLoginExist(loginTb.Text))

{

if (db.Login(loginTb.Text, passwordTb.Text))

{

passwordTb.Text = "";

(new Client(db, this)).ShowDialog();

}

else

Say("Пароль неправильний");

}

else

{

Say("Логін не існує");

}

}

}

private void register\_Click(object sender, EventArgs e)

{

if (CheckFill())

{

if (!db.CheckIfLoginExist(loginTb.Text))

{

if (db.Register(loginTb.Text, passwordTb.Text))

{

passwordTb.Text = "";

(new Client(db, this)).ShowDialog();

}

else

Say("Перевірте підключення");

}

else

{

Say("Логін зайнято");

}

}

}

private bool CheckFill()

{

loginTb.Text = loginTb.Text.Replace("'", "’");

passwordTb.Text = passwordTb.Text.Replace("'", "’");

if (loginTb.Text.Length < 3)

{

Say("Логін повинен містити принаймні 3 символи");

return false;

}

else if (passwordTb.Text.Length < 3)

{

Say("Пароль повинен містити принаймні 3 символи");

return false;

}

return true;

}

private void Say(String text)

{

MessageBox.Show(text, "You.NET messager", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

private void passwordTb\_KeyUp(object sender, KeyEventArgs e)

{

if (e.KeyData == Keys.Enter)

{

loginBtn\_Click(new object(), new EventArgs());

}

}

}

}

### Client

Дана форма призначення для листування і також є панеллю керування для здійснення інших дій: виходу, створення чату, перемикання між чатами та їх редаугвання.

**Код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace msg

{

public partial class Client : Form

{

private DB db;

private Thread notificationChecker;

private Auth auth;

public Client(DB db, Auth auth)

{

this.db = db;

InitializeComponent();

login.Text = db.UserLogin;

ChatsInnit(false);

notificationChecker = new Thread(new ThreadStart(CheckForNotifications));

notificationChecker.Start();

this.auth = auth;

auth.Hide();

}

private void CheckForNotifications()

{

while (true)

{

try

{

if (db.CheckForNotifications())

{

(new System.Media.SoundPlayer(Properties.Resources.Notification)).Play();

Invoke(new MethodInvoker(() => ChatsInnit(false)));

//Invoke(new MethodInvoker(ChatsInnit));

}

}

catch { };

Thread.Sleep(333);

}

}

private void ChatsInnit(bool isChatNew)

{

Chat[] chat = db.LoadChats();

foreach (Chat ch in chat)

{

ch.Click += new EventHandler(clearChat) + new EventHandler(openChat);

if (ch.Id == db.ChatId && ch.isUnreaded)

{

openChat(ch, new EventArgs());

if (!isChatNew)

return;

}

}

chatsLayout.Controls.Clear();

chatsLayout.Controls.AddRange(chat);

}

private void clearChat(object sender, EventArgs e)

{

msgsLayout.Controls.Clear();

msgBox.Clear();

}

private void openChat(object sender, EventArgs e)

{

// local info introduction

setExtraChatInfo((Chat)sender);

//fill layout

Messages[] msg = db.LoadMsgs((sender as Chat).Id);

int newMsgAm = db.GetNewMessagesAmount();

for(int i = msgsLayout.Controls.Count; i < msg.Length; i++)

{

msgsLayout.Controls.Add(msg[i]);

}

// set scroll to last unreaded

msgsLayout.ScrollControlIntoView(msgsLayout.Controls[msgsLayout.Controls.Count - newMsgAm - (newMsgAm == 0 ? 1 : 0)]);

msgBox.Focus();

}

private void send\_Click(object sender, EventArgs e)

{

msgBox.Text = msgBox.Text.Trim('\n');

if (msgBox.Text.Replace(" ","").Length > 0)

{

if (db.SendMsg(msgBox.Text.Replace("'", "’")))

{

msgsLayout.Controls.Add(db.displayNewDateIfRequired(DateTime.Now));

var m = new Outbox(msgBox.Text, DateTime.Now.ToString("HH:mm:ss"), db.GetSentMsgId(), db);

msgsLayout.Controls.Add(m);

msgBox.Clear();

msgsLayout.ScrollControlIntoView(m);

}

else

MessageBox.Show("Перевірте з'єднання");

}

msgBox.Text = msgBox.Text.Trim('\n');

}

private void emoji\_Click(object sender, EventArgs e)

{

msgBox.AppendText(((Label)sender).Text);

}

private void setExtraChatInfo(Chat c)

{

if(db.SetMessagesRead())

c.unreaded.Text = "";

chatName.Text = c.user.Text;

int participantsAmount = db.GetParticipantsAmountById(c.Id);

if (participantsAmount == 2)

participants.Text = "";

else

participants.Text = participantsAmount + " particitants";

}

private void msgBox\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyData == (Keys.Control | Keys.Enter))

{

send\_Click(new object(), new EventArgs());

}

}

private void addChat\_Click(object sender, EventArgs e)

{

if((new AddChat(db)).ShowDialog() == DialogResult.OK)

ChatsInnit(true);

}

private void attach\_Click(object sender, EventArgs e)

{

}

private void logout\_Click(object sender, EventArgs e)

{

this.Close();

}

private void Client\_FormClosed(object sender, FormClosedEventArgs e)

{

notificationChecker.Abort();

auth.Show();

}

}

}

## Форми для введення

### GroupNameEditForm

Дана форма викликається, щоб надати користувачу змогу ввести нове ім’я чату, що далі буде перенаправлене до бази даних для зміни в мережі а також в форму Client для встановлення назви локально

**Код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace msg.InputForms

{

public partial class GropNameEditInput : Form

{

public GropNameEditInput(String currentName)

{

InitializeComponent();

input.Text = currentName;

}

private void ok\_Click(object sender, EventArgs e)

{

if(input.Text.Length >= 3)

{

DialogResult = DialogResult.OK;

this.Close();

}

else

{

MessageBox.Show("Назва групи повинна містити принаймні 3 символа", "You.NET messager", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

private void cancel\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

### InputForm

Ця форма є універсальною для запитання вводу будь-якої інформації. Її конструктор приймає заголовок, щоб пояснити користувачу, який саме ввід від нього очікує програма.

**Код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace msg.InputForms

{

public partial class GropNameEditInput : Form

{

public GropNameEditInput(String currentName)

{

InitializeComponent();

input.Text = currentName;

}

private void ok\_Click(object sender, EventArgs e)

{

if(input.Text.Length >= 3)

{

DialogResult = DialogResult.OK;

this.Close();

}

else

{

MessageBox.Show("Назва групи повинна містити принаймні 3 символа", "You.NET messager", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

private void cancel\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

# Classes

### DB

Цей клас реалізує всі зовнішні підключення програми. Забезпечує отримання, додавання, редагування та вилучення інформації з MySQL бази даних.

**Код:**

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

namespace msg

{

public class DB

{

private static String connStr = "Server=herbalife.in.ua;Database=dfbaddbt\_msg;port=3306;User Id=dfbaddbt\_admin;password=WUJRrtFb64qdaw7;Charset=utf8mb4";

public int UserId { get; private set; }

public int ChatId { get; private set; }

public String UserLogin { get; private set; }

// AUTH //

public bool CheckIfLoginExist(String login)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

if (Convert.ToInt32(new MySqlCommand($"SELECT COUNT(id) FROM `user` WHERE login = '{login}'", conn).ExecuteScalar()) == 0)

return false;

return true;

}

}

public bool Login(String login, String pass)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

int id = Convert.ToInt32(new MySqlCommand($"SELECT id FROM `user` WHERE (login = '{login}' AND pass = '{pass}')", conn).ExecuteScalar());

if (id != 0)

{

UserId = id;

UserLogin = login;

return true;

}

return false;

}

}

public bool Register(String login, String pass)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

if (new MySqlCommand($"INSERT INTO `user`(login, pass) VALUES ('{login}', '{pass}')", conn).ExecuteNonQuery() == 0)

return false;

if (!Login(login, pass))

return false;

return true;

}

}

// AUTH end //

// LOAD CHATS //

public Chat[] LoadChats()

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

var reader = new MySqlCommand($"SELECT max(msg.time) AS time, chat.chat\_id, chat.chat\_name, chat.new\_msgs FROM `chat` INNER JOIN `msg` ON chat.chat\_id = msg.chat\_id WHERE (user\_id = {UserId}) GROUP BY chat.chat\_id ORDER BY time DESC", conn).ExecuteReader();

List<Chat> chats = new List<Chat>();

while (reader.Read())

{

chats.Add(new Chat(reader.GetInt32("chat\_id"), reader.GetString("chat\_name"), reader.GetInt32("new\_msgs"), this));

}

reader.Close();

return chats.ToArray();

}

}

// LOAD MSGS //

public Messages[] LoadMsgs(int chatId)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

ChatId = chatId;

var reader = new MySqlCommand($"SELECT user.id, user.login, msg.id AS msg\_id, msg.text, msg.time FROM `msg` INNER JOIN `user` ON (user.id = sender\_id) WHERE (chat\_id = {chatId})", conn).ExecuteReader();

List<Messages> m = new List<Messages>();

while (reader.Read())

{

int senderId = reader.GetInt32("id");

String senderLogin = reader.GetString("login");

String msgText = reader.GetString("text");

DateTime time = reader.GetDateTime("time");

// setup data

m.Add(displayNewDateIfRequired(time));

// msgs

if (senderId == UserId)

{

m.Add(new Outbox(msgText, time.ToString("HH:mm:ss"), reader.GetInt32("msg\_id"), this));

}

else if (senderId == 0)

{

m.Add(new ChatInfo(msgText));

}

else

{

m.Add(new Inbox(senderLogin, msgText, time.ToString("HH:mm:ss")));

}

}

reader.Close();

return m.ToArray();

}

}

// SEND MSG //

public bool SendMsg(String text)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

if (new MySqlCommand($"INSERT INTO `msg`(chat\_id, sender\_id, text) VALUES ({ChatId}, {UserId}, '{text}')", conn).ExecuteNonQuery() == 0

|| new MySqlCommand($"UPDATE `chat` SET new\_msgs = new\_msgs + 1 WHERE(chat\_id = {ChatId} AND user\_id != {UserId})", conn).ExecuteNonQuery() == 0

|| !TriggerChattersUpdate(ChatId, conn))

return false;

return true;

}

}

// ADD CHAT //

public Users[] LoadUsers(String search, int[] selectedUsers)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

var reader = new MySqlCommand($"SELECT id, login FROM `user` WHERE (login LIKE '%{search}%' AND id != 0 AND id != {UserId})", conn).ExecuteReader();

List<Users> users = new List<Users>();

while (reader.Read())

{

int id = reader.GetInt32("id");

var c = new Users(id, reader.GetString("login"), selectedUsers.Contains(id));

users.Add(c);

}

reader.Close();

return users.ToArray();

}

}

// CREATE CHAT //

public bool CreatePrivateChat(int participantId)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

int chatId = GetFollowingChatId(conn);

if (!CreateChat(participantId, chatId, UserLogin) || !CreateChat(UserId, chatId, GetUsernameById(participantId, conn)))

// створення автоназваних екземплярів чатів для обох учасників

return false;

ChatId = chatId;

InsertInfoMsg(ChatId, $"Чат розпочато", conn);

return true;

}

}

public bool CreatePublicChat(int[] participantIds, String chatname)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

int chatId = GetFollowingChatId(conn);

foreach (var id in participantIds)

{

if (!CreateChat(id, chatId, chatname))

return false;

}

ChatId = chatId;

InsertInfoMsg(ChatId, $"{UserLogin} створив чат!", conn);

return true;

}

}

public bool CreateChat(int participantId, int chatId, String chatname)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

if (new MySqlCommand($"INSERT INTO `chat`(chat\_id, user\_id, chat\_name, new\_msgs) VALUES ({chatId}, {participantId}, '{chatname}', 1)", conn).ExecuteNonQuery() == 0)

return false;

return true;

}

}

// extra client actions //

public int GetParticipantsAmountById(int id)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

return Convert.ToInt32(new MySqlCommand($"SELECT COUNT(user\_id) FROM `chat` WHERE chat\_id = {id}", conn).ExecuteScalar());

}

}

public bool CheckForNotifications()

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

if ((bool)new MySqlCommand($"SELECT hasNewMsg FROM `user` WHERE id = {UserId}", conn).ExecuteScalar())

{

new MySqlCommand($"UPDATE `user` SET hasNewMsg = FALSE WHERE id = {UserId}", conn).ExecuteNonQuery();

return true;

}

return false;

}

}

public bool SetMessagesRead()

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

if (new MySqlCommand($"UPDATE `chat` SET new\_msgs = 0 WHERE(chat\_id = {ChatId} AND user\_id = {UserId})", conn).ExecuteNonQuery() == 0)

return false;

return true;

}

}

// DELETE MSG //

public bool DeleteMsg(int id)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

if (new MySqlCommand($"DELETE FROM `msg` WHERE `msg`.`id` = {id}", conn).ExecuteNonQuery() == 0)

return false;

return true;

}

}

// EDIT CHATNAME //

public bool EditChatNameForYourself(int id, String newChatName)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

if (new MySqlCommand($"UPDATE `chat` SET chat\_name = '{newChatName}' WHERE (chat\_id = {id} AND user\_id = {UserId})", conn).ExecuteNonQuery() == 0)

return false;

return true;

}

}

public bool EditChatNameForGroup(int id, String newChatName)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

if (new MySqlCommand($"UPDATE `chat` SET chat\_name = '{newChatName}' WHERE chat\_id = {id}", conn).ExecuteNonQuery() == 0)

return false;

return true;

}

}

// LEAVE CHAT //

public bool LeaveChat(int id)

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

if (new MySqlCommand($"DELETE FROM `chat` WHERE (chat\_id = {id} AND user\_id = {UserId})", conn).ExecuteNonQuery() == 0)

return false;

InsertInfoMsg(id, $"{UserLogin} покинув чат", conn);

return true;

}

}

// COMMON //

public String GetUsernameById(int id, MySqlConnection conn)

{

return (String)new MySqlCommand($"SELECT login FROM `user` WHERE id = {id}", conn).ExecuteScalar();

}

public int GetFollowingChatId(MySqlConnection conn)

{

return (int)new MySqlCommand($"SELECT chat\_id FROM `chat` ORDER BY chat\_id DESC LIMIT 1", conn).ExecuteScalar() + 1;

}

public int GetSentMsgId()

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

return (int)new MySqlCommand($"SELECT id FROM `msg` ORDER BY id DESC LIMIT 1", conn).ExecuteScalar();

}

}

public int GetNewMessagesAmount()

{

using(MySqlConnection conn = new MySqlConnection(connStr))

{

conn.Open();

return (int)new MySqlCommand($"SELECT new\_msgs FROM `chat` WHERE (chat\_id = {ChatId} AND user\_id = {UserId})", conn).ExecuteScalar();

}

}

private bool InsertInfoMsg(int chatId, String text, MySqlConnection conn)

{

if (new MySqlCommand($"INSERT INTO `msg`(chat\_id, sender\_id, text) VALUES ({chatId}, 0, '{text}')", conn).ExecuteNonQuery() == 0

|| TriggerChattersUpdate(chatId, conn))

return false;

return true;

}

private bool TriggerChattersUpdate(int chatId, MySqlConnection conn)

{

if (new MySqlCommand($"UPDATE `user`, `chat` SET hasNewMsg = TRUE WHERE(user.id = chat.user\_id AND chat.chat\_id = {chatId} AND user.id != {UserId})", conn).ExecuteNonQuery() == 0)

return false;

return true;

}

public int lastMsgDate = -1;

public ChatInfo displayNewDateIfRequired(DateTime date)

{

ChatInfo ch = null;

if (lastMsgDate != date.DayOfYear)

{

ch = new ChatInfo(date.ToString("dddd, dd MMMM"));

}

lastMsgDate = date.DayOfYear;

return ch;

}

}

}

### Program

Стандартний автогенерований клас, із якого розпочинається виконання програми.

**Код:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace msg

{

static class Program

{

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Auth());

}

}

}

# User controls

## Для повідомлень

### Messages

User control, що є по своїй суті просто батьківським елементом для всіх інших повідомлень і застосовується для узагальнення.

**Код:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace msg

{

public class Messages : UserControl

{

}

}

### ChatInfo

User control, призначений для виведення загальної інформації в чаті, без конкретно визначеногно автора. Використовується для відображення дат та подій(створення чату, зміна назви, вихід учасника).

**Код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Drawing;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace msg

{

public partial class ChatInfo : Messages

{

public ChatInfo(String text)

{

InitializeComponent();

this.text.Text = text;

}

}

}

### Inbox

User control, призначений відбраження вхідних повідомлень від різних учаників чату.

**Код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Drawing;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace msg

{

public partial class Inbox : Messages

{

public Inbox(String sender, String text, String time)

{

InitializeComponent();

this.text.Text = text;

this.time.Text = time;

container.Text = sender;

}

}

}

### Outbox

User control, призначений відбраження вихідних повідомлень користувача.

**Код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Drawing;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace msg

{

public partial class Outbox : Messages

{

public int Id { get; private set; }

DB db;

public Outbox(String text, String time, int id, DB db)

{

InitializeComponent();

this.text.Text = text;

this.time.Text = time;

Id = id;

this.db = db;

}

private void delete\_Click(object sender, EventArgs e)

{

if(MessageBox.Show("Видалити повідомлення?", "You.NET", MessageBoxButtons.YesNo) == DialogResult.Yes)

{

db.DeleteMsg(Id);

this.Dispose();

}

}

private void edit\_Click(object sender, EventArgs e)

{

// not today

}

}

}

## Інші

### Chat

User control, відображає окремовзятий чат із його назвою та кількістю нових повідомлень, кнопками редагування та видалення.

**Код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Drawing;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using msg.InputForms;

namespace msg

{

public partial class Chat : UserControl

{

DB db;

public int Id { get; private set; }

public bool isUnreaded { get; private set; }

public Chat(int id, String name, int unreaded, DB db)

{

InitializeComponent();

Id = id;

user.Text = name;

this.db = db;

isUnreaded = unreaded > 0;

this.unreaded.Text = isUnreaded ? Convert.ToString(unreaded) : "";

}

private void edit\_Click(object sender, EventArgs e)

{

var inp = new GropNameEditInput(user.Text);

if (inp.ShowDialog() == DialogResult.OK)

{

if(inp.forEveryoneCheck.Checked)

{

db.EditChatNameForGroup(Id, inp.input.Text);

}

else

{

db.EditChatNameForYourself(Id, inp.input.Text);

}

user.Text = inp.input.Text;

}

}

private void leave\_Click(object sender, EventArgs e)

{

if(MessageBox.Show("Are u sure u want to leave this chat?", "You.NET", MessageBoxButtons.YesNo) == DialogResult.Yes)

{

if(db.LeaveChat(Id))

this.Dispose();

}

}

}

}

### Users

User control, відображає користувачів, яких пропонує долучсити до чату форма AddChat.

**Код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Drawing;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace msg

{

public partial class Users : UserControl

{

public Users(int id, String name, bool isSelected)

{

InitializeComponent();

this.name.Text = name;

add.Tag = id;

add.Click += AddUserEffects;

if (isSelected)

{

AddUserEffects(new object(), new EventArgs());

add.Text = "-";

}

}

private void AddUserEffects(object sender, EventArgs e)

{

if(add.Text == "+")

{

this.BackColor = Color.Pink;

add.BackColor = Color.Pink;

add.ForeColor = Color.DeepPink;

}

else

{

this.BackColor = Color.PaleTurquoise;

add.BackColor = Color.PaleTurquoise;

add.ForeColor = Color.Turquoise;

}

}

}

}

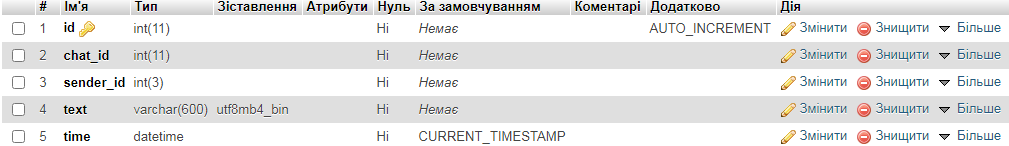
# MySQL Server

Для зберігання даних та обміну ними в реальному часі програма використовує MySQL Server. Далі я наведу скріншоти структури відповідних SQL таблиць.

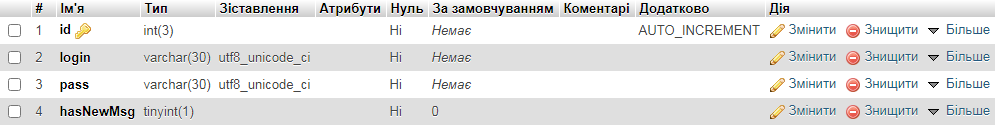
### Таблиця chat



### Таблиця msg



### Таблиця user



**Зміст**

[Опис задачі 2](#_Toc43231605)

[Файловий вміст проєкта 2](#_Toc43231606)

[Forms 3](#_Toc43231607)

[Користувацькі форми 3](#_Toc43231608)

[1. AddChat 3](#_Toc43231609)

[2. Auth 4](#_Toc43231610)

[3. Client 6](#_Toc43231611)

[Форми для введення 8](#_Toc43231612)

[4. GroupNameEditForm 8](#_Toc43231613)

[5. InputForm 9](#_Toc43231614)

[Classes 10](#_Toc43231615)

[1. DB 10](#_Toc43231616)

[2. Program 15](#_Toc43231617)

[User controls 16](#_Toc43231618)

[Для повідомлень 16](#_Toc43231619)

[1. Messages 16](#_Toc43231620)

[2. ChatInfo 16](#_Toc43231621)

[3. Inbox 16](#_Toc43231622)

[4. Outbox 17](#_Toc43231623)

[Інші 18](#_Toc43231624)

[5. Chat 18](#_Toc43231625)

[6. Users 19](#_Toc43231626)

[MySQL Server 20](#_Toc43231627)

[1. Таблиця chat 20](#_Toc43231628)

[2. Таблиця msg 20](#_Toc43231629)

[3. Таблиця user 20](#_Toc43231630)