**Міністерство освіти і науки України**

**Національний технічний університет України**

**“Київський політехнічний інститут”**

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**КУРСОВА РОБОТА**

з дисципліни “ Архітектура та проектування програмного забезпечення ”

напряму підготовки 6.050103 – Програмна інженерія

на тему:

**«Розробка веб-сайту соціальної мережі для обміну фотографіями»**

**Студентка групи** КП-21 **Соколовська Анна Віталіївна**  \_\_\_\_\_\_\_\_\_\_

(підпис)

**Викладач: Мелащенко Андрій Олегович \_\_\_\_\_\_\_\_\_\_**

(підпис)

Захищено з оцінкою \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Київ – 2015

**Анотація**

В курсовому проекті основною метою було створення веб-сайту для обміну фото користувачами. В ході проведення роботи та аналізу предметної галузі, в якості СКБД була обрана графова база даних Neo4j, в якості технології створення сайту – ASP.NET MVC5 та мова С#. Цей потужний інструментарій дозволив створити інтуїтивно-зрозумілий та зручний у використанні продукт.

Під час створення проекту були набуті практичні навички створення сучасного програмного забезпечення, що взаємодіє з нереляційними базами даних, а саме графовими БД, веб-програмування, а також навички оформлення відповідного текстового, програмного та ілюстративного матеріалу у формі проектної документації.

У результаті було отримано надійну, просту у використанні систему, яка може впроваджуватись у використання.

**Зміст**

Вступ 4

1. Опис реалізованого програмного забезпечення 5

2. Опис структурно-логічної схеми ПЗ 6

2.1. Загальна структура програмного забезпечення 6

2.2. Опис модулів програмного забезпечення 6

2.3 Опис основних алгоритмів роботи 9

3. Тестування програмного забезпечення 13

Висновки 18

Список використаних джерел 19

Додатки 20

Додаток А. Приклад збереження даних в Neo4j 20

**Вступ**

Дана курсова робота присвячена розробці веб-сайту для обміну фото із соціальною складовою під назвою «Photos», використовуючи при цьому різноманітні можливості сучасних технологій веб програмування.

*Об’єктом* курсової роботи є сама веб-сайт.

*Метою* роботи є розроблення ПЗ для веб-сайту із використанням набутих під час вивчення предмету знань та навичок.

Для досягнення визначеної мети необхідно виконати такі *завдання*:

* абстрагувати об’єкти предметної галузі;
* розробити структурну організацію ПЗ;
* визначити та описати функціональні характеристики програми;
* виконати реалізацію ПЗ відповідно до технічного завдання (ТЗ);
* виконати тестування розробленої програми;
* оформити документацію з курсової роботи.

Розроблене ПЗ складається з 2-х структурних частин: графової бази даних користувачів та їх постів, веб-сайту.

Для функціонування веб-серверу потрібна наявність IIS 7.0 та забезпечення системних умов для його функціонування, а також наявність доступу до Інтернету.

Розроблене ПЗ може бути використане з розважальною метою.

Пояснювальна записка складається зі вступу, двох розділів, загальних висновків, списку використаних джерел (3 найменування) та додатків (1 додаток). Робота містить 12 рисунків. Загальний обсяг роботи – 20 друкованих сторінок, з них 15 сторінок основного тексту та 1 сторінка списку використаних джерел.

1. **Опис реалізованого програмного забезпечення**

В ході реалізації курсової роботи була створена багатокористувацький веб-сайт з соціальною складовою.

Для реєстрації на сайті користувачу достатньо бути зареєстрованим в соціальній мережі «Вконтакте». Подальша автентифікація користувача, при вході на сайт полягатиме у вході на сайт «Вконтакте».

Користувач може створювати пости, куди додаватиме фото та опис до нього. Крім того користувач може знаходити свої друзів та «слідувати» (англ. *follow*) ними. Активність друзів відображається у новинах користувача, за умови, якщо він слідує за ними.

Користувач може «вподобати» та прокоментувати пост друга. За необхідності користувач може видалити, написаний ним коментар, відмінити своє «вподобання» посту або перестати слідувати за другом.

Користувач може видаляти свої пости. В такому разі всі коментарі та вподобання даного посту також видаляються.

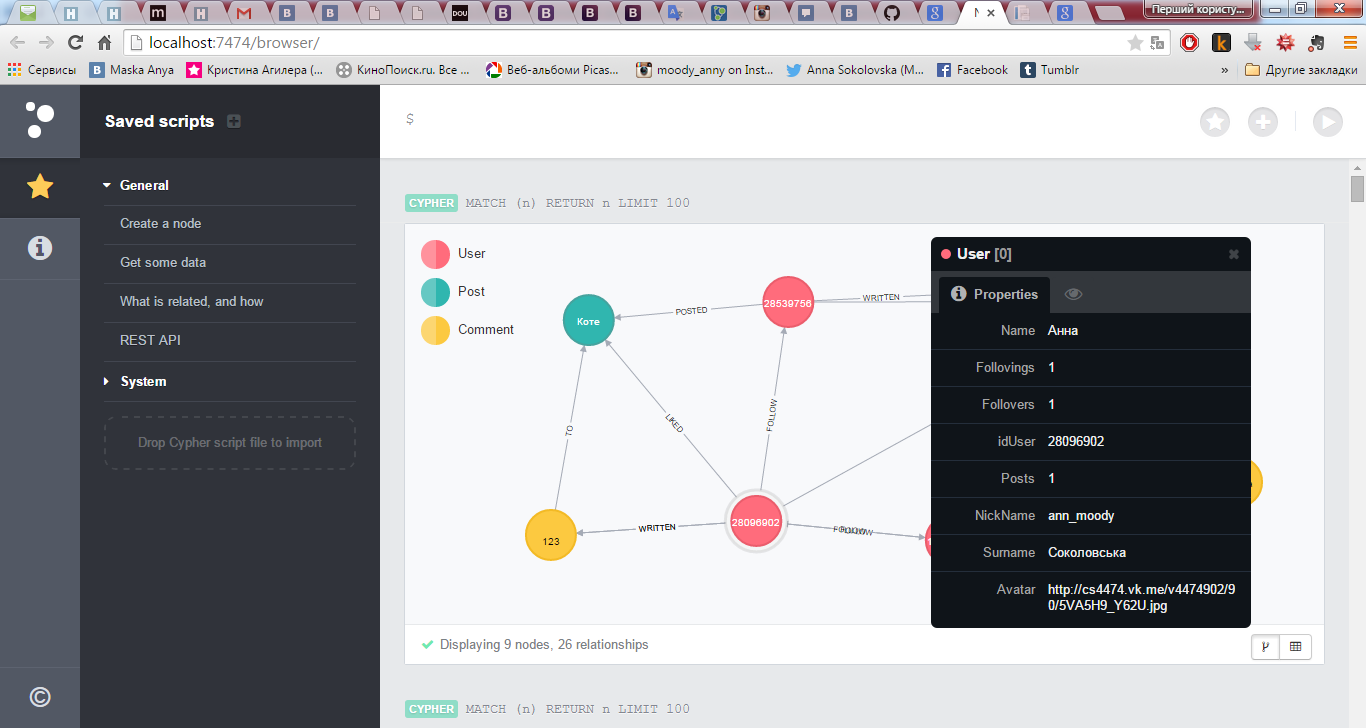
1. **Опис структурно-логічної схеми ПЗ**
   1. **Загальна структура програмного забезпечення**

У створеному та використовуваному програмному забезпечені можна виділити два структурні елементи, що взаємодіють між собою.

* графова СКБД Neo4j, яка використовується для зберігання користувачів, постів та зв’язків між ними. Для її розміщення використовується Neo4j на локальному персональному комп’ютері;
* веб-сайт ASP.NET MVC5, який надає клієнту доступ до контенту сайту та обробляє його запити.
  1. **Опис модулів програмного забезпечення**

Розглянемо моделі даних, що зберігаються в БД. Приклад збереження даних в БД наведено в додатку 1.

В базі даних зберігаються наступні сутності:



*Рис. 1.* Структура документу Neo4j

Лейбл User представлений на рис.1 містить наступні сутності:

* *idUser* – зберігає унікальний ідентифікаційний код користувача, що розроблений сайт отримує від сайту «Вконтакте»;
* *Name* – зберігає ім’я користувача з сайту «Вконтакте»;
* *Surname* – зберігає прізвище користувача з сайту «Вконтакте»;
* *NickName* – зберігає псевдонім користувача з сайту «Вконтакте»;
* *Avatar*  – зберігає url-адресу «аватару» (головної сторінки профілю користувача) користувача з сайту «Вконтакте»;
* *Follovers* – зберігає кількість користувачів, що слідують за даним користувачем;
* *Follovings* – зберігає кількість користувачів, за яким слідує даний користувач;
* *Posts* – зберігає кількість постів користувача.

Лейбл Post представлений на рис.1 містить наступні сутності:

* *idUserPosted* – зберігає унікальний ідентифікаційний код користувача, який створив пост;
* *Date* – зберігає дату створення посту;
* *Text* – зберігає опис користувача до цього посту;
* *Photo*  – зберігає назву фотографії, яку користувач додав у пост;
* *Likes* – зберігає кількість користувачів, що вподобали пост;
* *Comments* – зберігає кількість коментарів до посту.

Лейбл Comment представлений на рис.1 містить наступні сутності:

* *idUserPosted* – зберігає унікальний ідентифікаційний код користувача, який додав коментар;
* *UserData* – зберігає інформацію про користувача, що додав коментар (прізвище, ім’я);
* *Text* – зберігає текст коментаря;
* *Photo*  – зберігає назву фотографії (посту, що містить дане фото), до якої користувач написав коментар;
* *Avatar* – зберігає url-адресу «аватару» (головної сторінки профілю користувача) користувача з сайту «Вконтакте»;
* *CreationTime* – зберігає унікальний ідентифікаційний коментаря.

Між лейблами (вершинами графа) існують такі зв’язки:

* *Follow* – користувач «слідує» за іншим користувачем (User– [Follow]–>User);
* *Posted*– користувач запостив пост (User– [Posted]–>Post);
* *To* – коментар написаний до посту (Comment– [To]–>Post);
* *Written* – коментар написаний користувачем (User–[Written]

– > Comment).

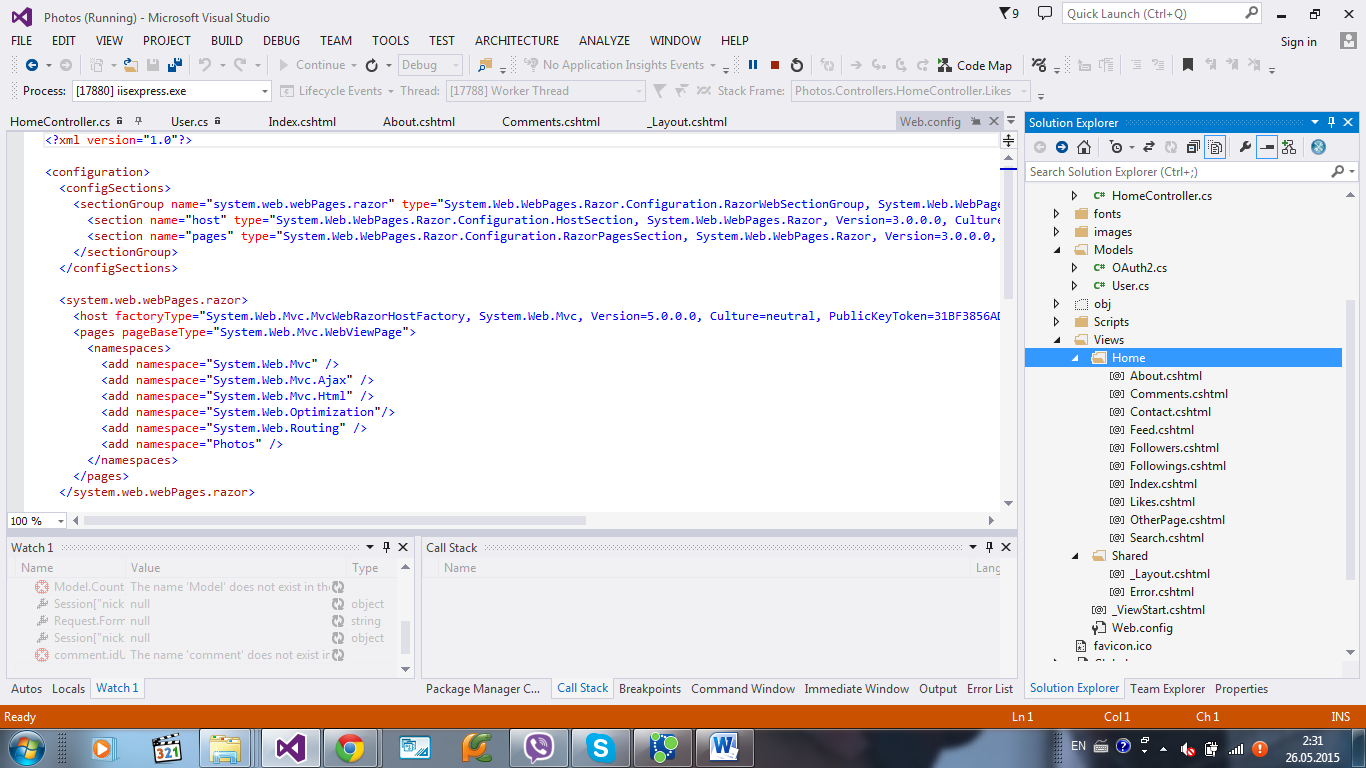
Веб-сайт побудований на основі концепції Model-view-controller з використанням технології ASP.NET MVC Framework використовує ідеї контролерів (Controllers), відображень (view) та даних з якими ми працюємо (Model).

* 1. **Опис основних алгоритмів роботи**

Робота сайту здійснюється за принципом Model-view-controller, тобто через view ми отримуємо запит користувача на виконання певної задачі, controller обробляє цей запит і здійснює за необхідності запит до model (в конкретному випадку до графової бази даних) на зміну, видалення, додавання або читання даних і надсилає на view відповідь користувачу на його запит.

Розглянемо декілька прикладів обробки даних у view. Всі .cshtml сторінки містять спільну частину – панель швидкого доступу до різних сторінок сайту, тому цей шматок коду винесений в спільну частину – \_Layout.cshtml.

Користувач має доступ до 10 різних сторінок .cshtml, тобто view.



*Рис. 2.* Структура сайту (сторінок .cshtml, view)

* about.cshtml – відображує сторінку профілю користувача та всі його пости;
* comments.cshtml – відображує всі коментарі до конкретного посту, та надає можливість написати новий;
* contact.cshtml – відображує сторінку створення нового посту і надає можливість його створити;
* feed.cshtml – відображує новини про активність (нові пости) користувачів, за якими слідує даний користувач;
* followers.cshtml – відображує всфх користувачів, що слідують за даним користувачем
* followings.cshtml – відображує всіх користувачів, за який слідує даний користувач;
* index.cshtml – відображує основну сторінку сайту, яку бачать зареєстровані і незареєстровані користувачі.
* likes.cshtml – відображує всіх користувачів, що вподобали даний пост, а також надає можливість вподобати (якщо ще не вподобаний, або видалити з вподобань, якщо вже був вподобаний раніше);
* otherpage.cshtml – відображує сторінку іншого користувача;
* search.cshtml – відображує сторінку пошуку та дає змогу знайти бажаного користувача.

Приклад HTML-коду, який дозволяє зберегти зображення в потрібну папку з унікальним ім’ям:

<fieldset>

<legend> Upload Image </legend>

<label for="Image">Image</label>

<input class="btn btn-primary" type="file" name="Image" />

<br />

<input class="btn btn-primary" type="submit" value="Upload" />

<br />

</fieldset>

@{

WebImage photo = null;

var newFileName = "";

var imagePath = "";

if (IsPost)

{

photo = WebImage.GetImageFromRequest();

if (photo != null)

{

newFileName = Guid.NewGuid().ToString() + "\_" +

Path.GetFileName(photo.FileName);

imagePath = newFileName;

photo.Save(@"~/images/" + imagePath);

}

}

}

Приклад методу, що перевіряє авторизацію через соціальну мережу «Вконтакте»:

public static bool IsAuth(string data)

{

string token = data;

return token != null && token != String.Empty;

}

Приклад контролеру, що здійснює автентифікацію або реєстрацію користувача в системі через соціальну мережу «Вконтакте» через OAuth2 (використовуючи відповідний клас OAuth2.cs):

public ActionResult vk(string code)

{

if (!string.IsNullOrEmpty(code))

{

OAuth2 vk = new OAuth2("4912533", "RMuwuJroByXRlK6DbvGa", "http://api.vkontakte.ru/oauth/authorize", "https://api.vkontakte.ru/oauth/access\_token", "http://localhost:51173/Home/vk");

vk.Code = code;

OAuth2Token token = vk.GetAccessToken(new Dictionary<string, string> { { "client\_secret", "RMuwuJroByXRlK6DbvGa" } }, OAuth2.AccessTokenType.JsonDictionary);

if (token != null)

{

if (token.dictionary\_token != null)

{

Session.Add("access\_token", token.dictionary\_token["access\_token"]);

Session.Add("idUser", token.dictionary\_token["user\_id"]);

string res = OAuth2UserData.GetVKUserData(token.dictionary\_token["access\_token"], new Dictionary<string, string>() {

{"user\_ids", token.dictionary\_token["user\_id"]},

{"fields", "photo\_200, screen\_name"}});

res = res.Replace("\"", "'");

res = res.Replace("{'response':[", "");

res = res.Replace("}]}", "}");

dynamic stuff = JObject.Parse(res);

string first\_name = stuff.first\_name;

Session.Add("Name", first\_name);

string last\_name = stuff.last\_name;

Session.Add("Surname", last\_name);

string screen\_name = stuff.screen\_name;

Session.Add("NickName", screen\_name);

string photo\_id = stuff.photo\_200;

Session.Add("Avatar", photo\_id);

Session.Add("Follovers", 0);

Session.Add("Follovings", 0);

Session.Add("Posts", 0);

}

}

}

else

{

OAuth2 vk = new OAuth2("4912533", "RMuwuJroByXRlK6DbvGa", "http://api.vkontakte.ru/oauth/authorize", "https://api.vkontakte.ru/oauth/access\_token", "http://localhost:51173/Home/vk");

vk.GetAuthCode(new Dictionary<string, string>() { { "display", "popup" } });

}

return RedirectToAction("About");

}

Приклад запиту до бази:

Створюємо підключення за допомогою Neo4jClient:

var client = new GraphClient(new Uri("http://localhost:7474/db/data"));

client.Connect();

Знайдемо всі пости, що створив конкретний користувач:

var p = client.Cypher.OptionalMatch("(user:User)-[POSTED]->(post:Post)")

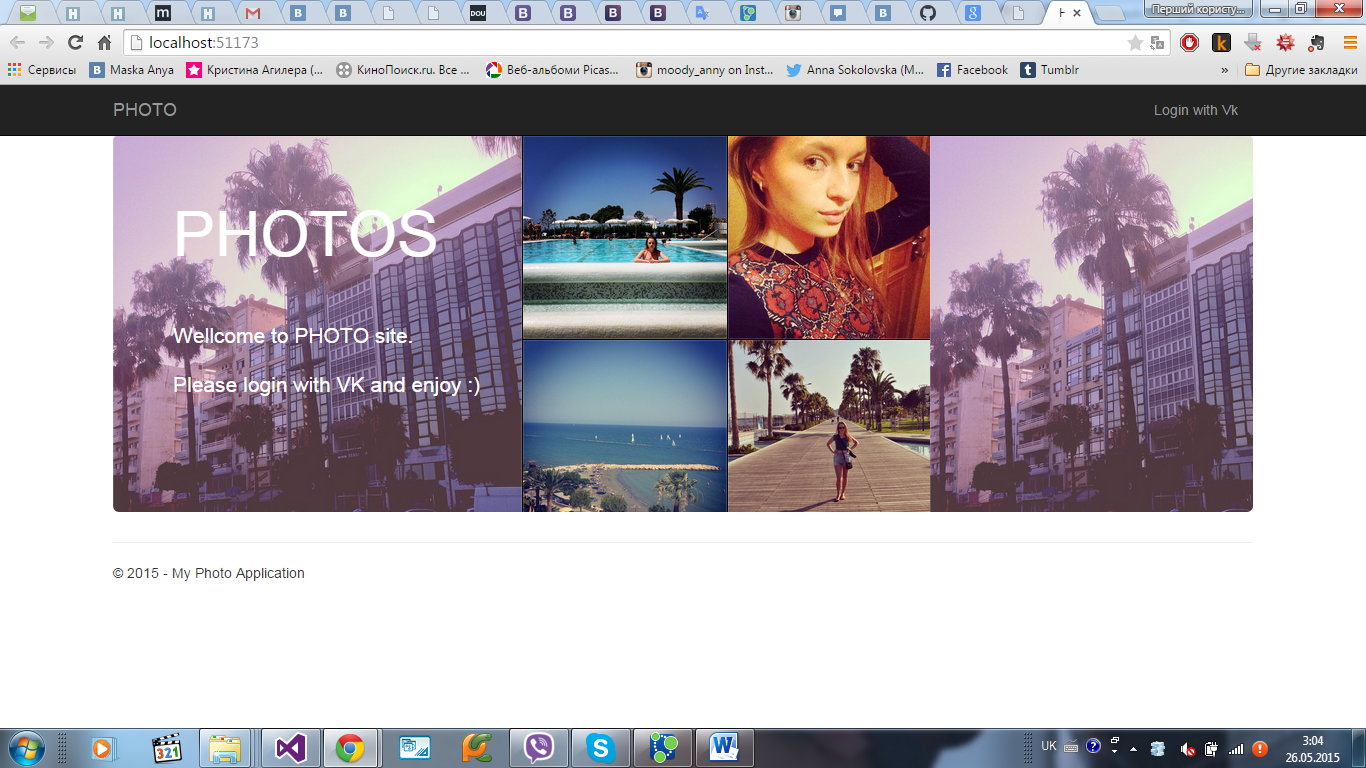
.Where((User user) => user.idUser == id)

.ReturnDistinct(post => post.As<Post>())

.Results;

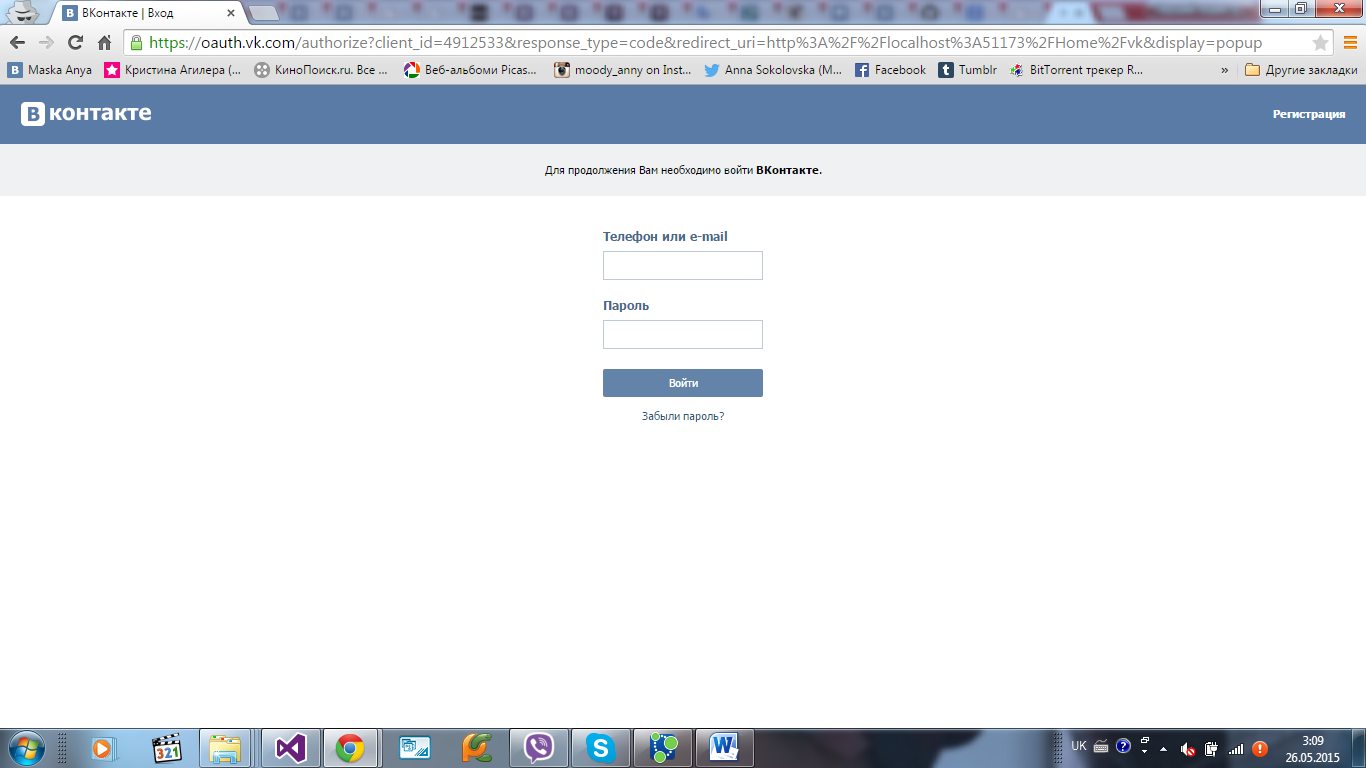
1. **Тестування програмного забезпечення**

При запуску веб-сайту гравця зустрічає index, де він може виконати вхід до свого аккаунту або зареєструватись за допомогою соціальної мережі «Вконтакті», натиснувши відповідну кнопку в правому верхньому кутку сторінки.



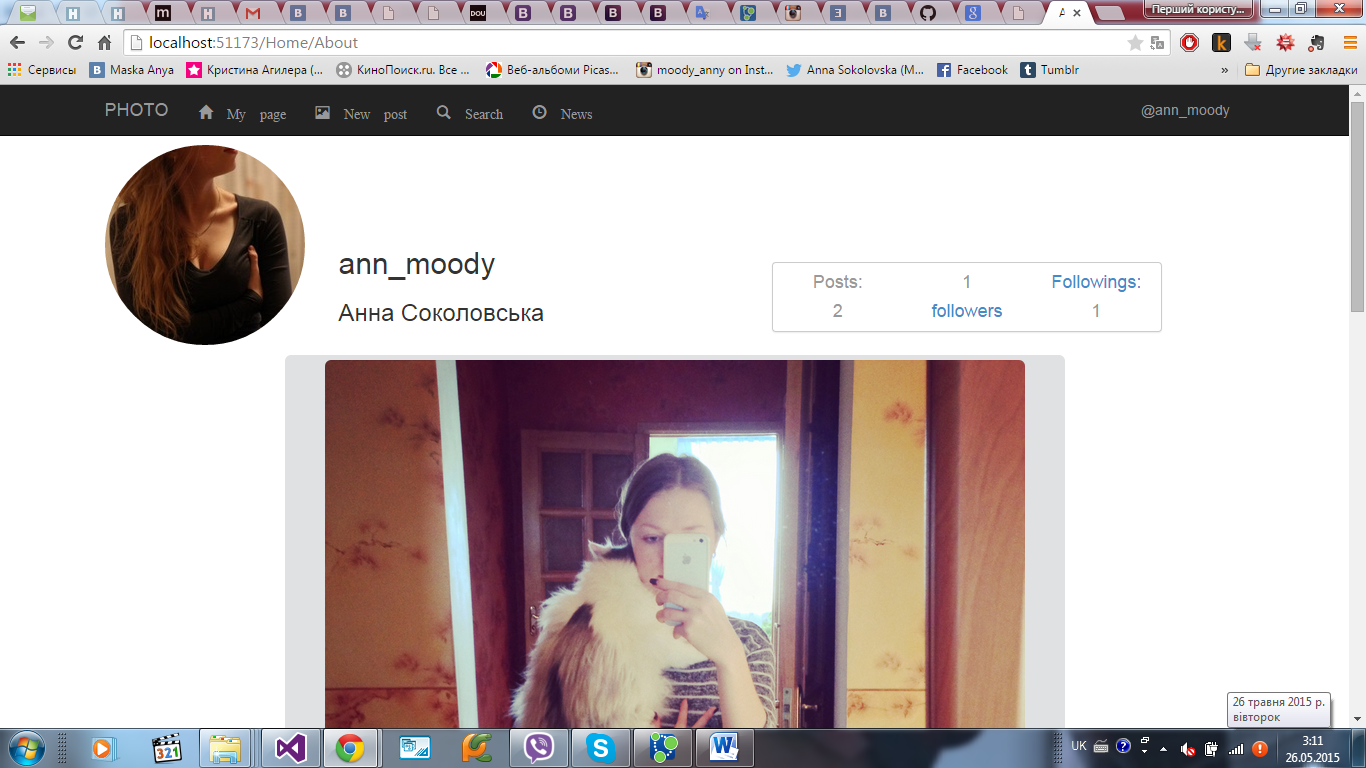
*Рис. 3.* Головна сторінка сайту

В випадку, коли людина не автентифікована в соціальній мережі «Вконтакте» з’являється сторінка входу на сайт «Вконтакте»:



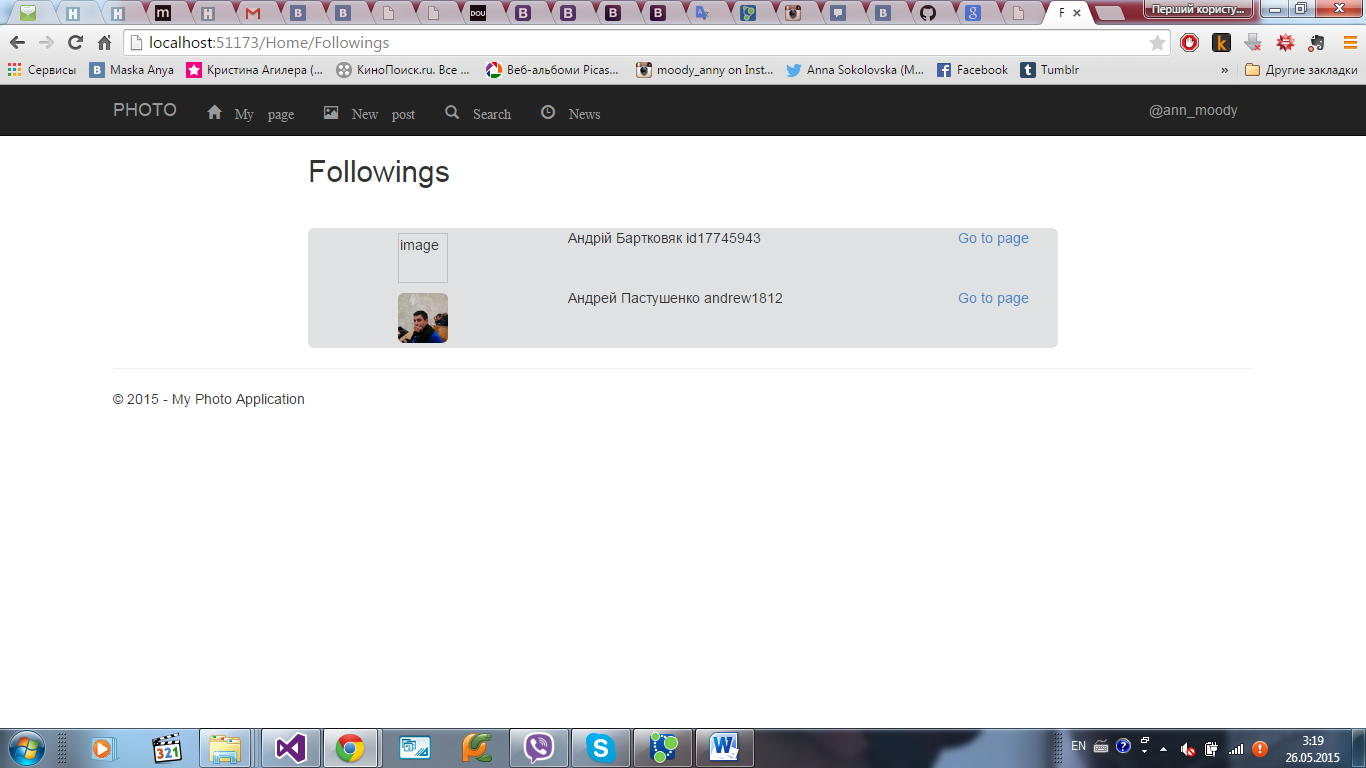
*Рис. 4.* Сторінка входу на сайт «Вконтакте»

Після вдалої автентифікації (або виконання реєстрації), користувач потрапляє на головну сторінку свого профілю, де відображеня його особисті дані та пости.



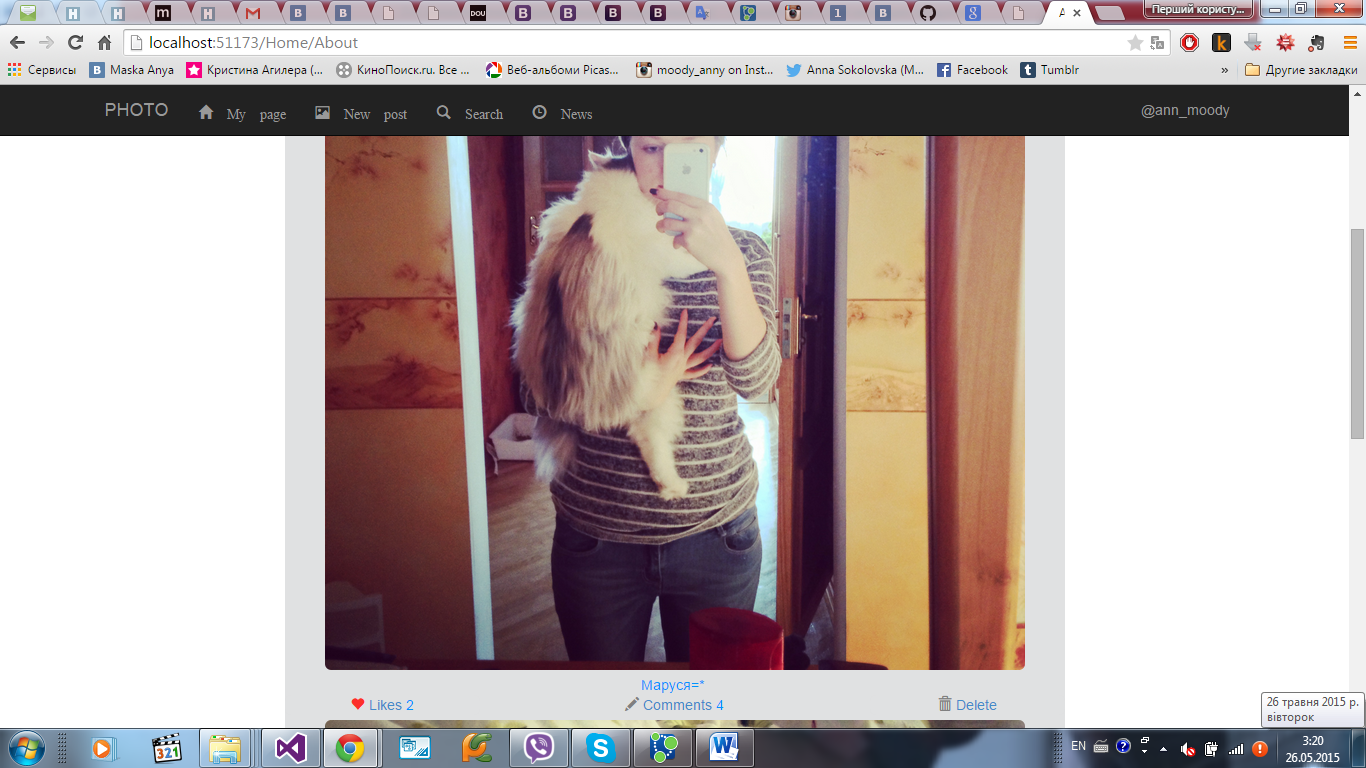
*Рис. 5.* Головна сторінка профілю користувача

Натиснувши на Followers або Followings перед користувачем з’явиться сторінка з користувачами, що слідуть за даним користувачем, або за якими слідує користувач відповідно. Наприклад:



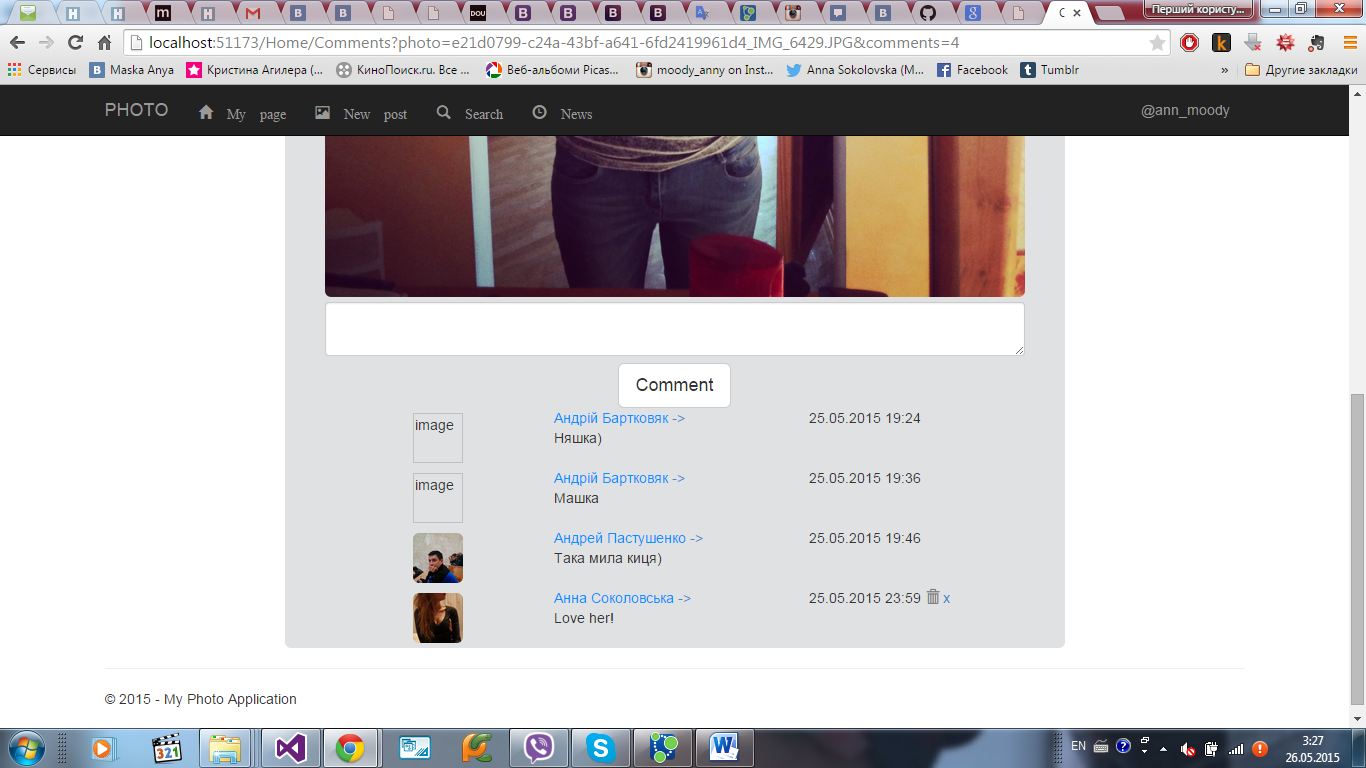
*Рис. 6.* Сторінка Followings

Під постом користувач може бачити опис, кількість вподобань, коментарів та має можливість видалити пост:



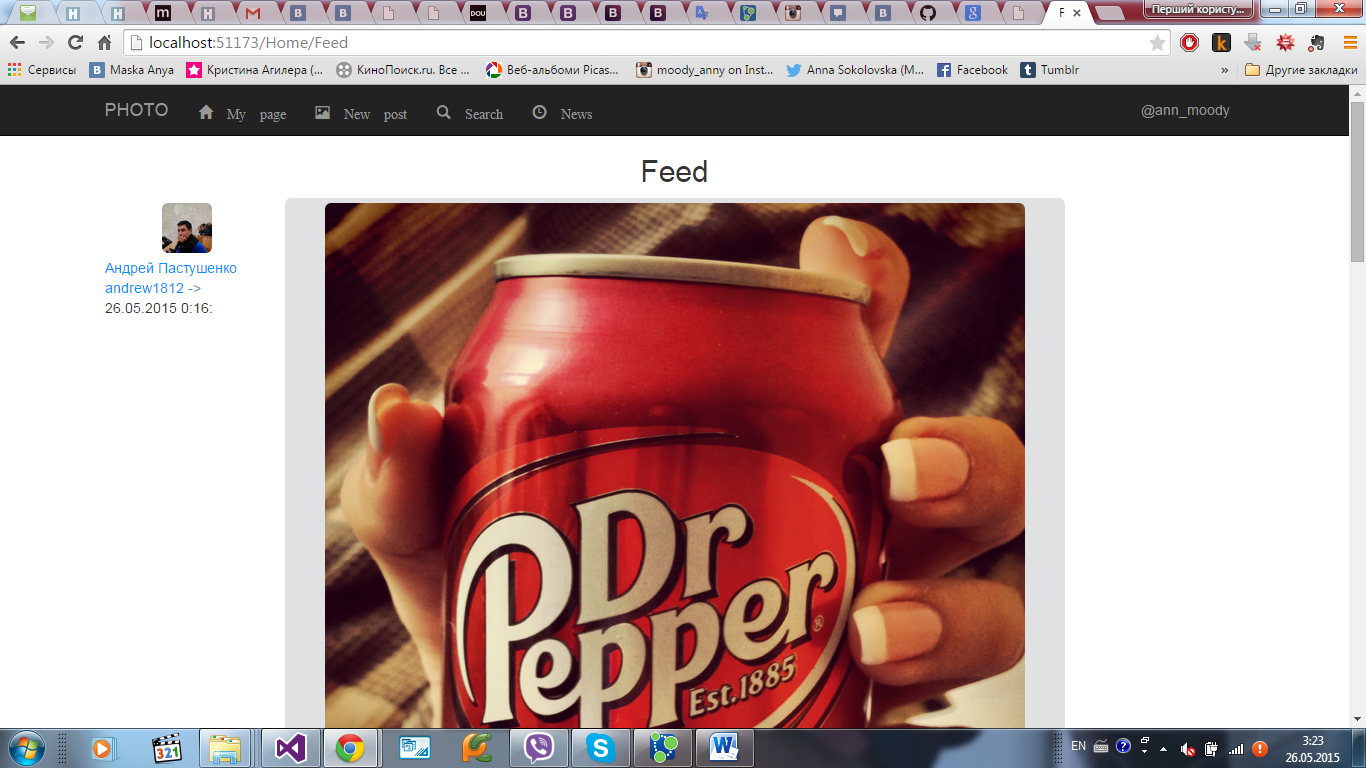
*Рис. 7.* Дані про пост

Переглянемо коментарі:



*Рис. 8.* Сторінка коментарів до посту з можливістю додати новий коментар

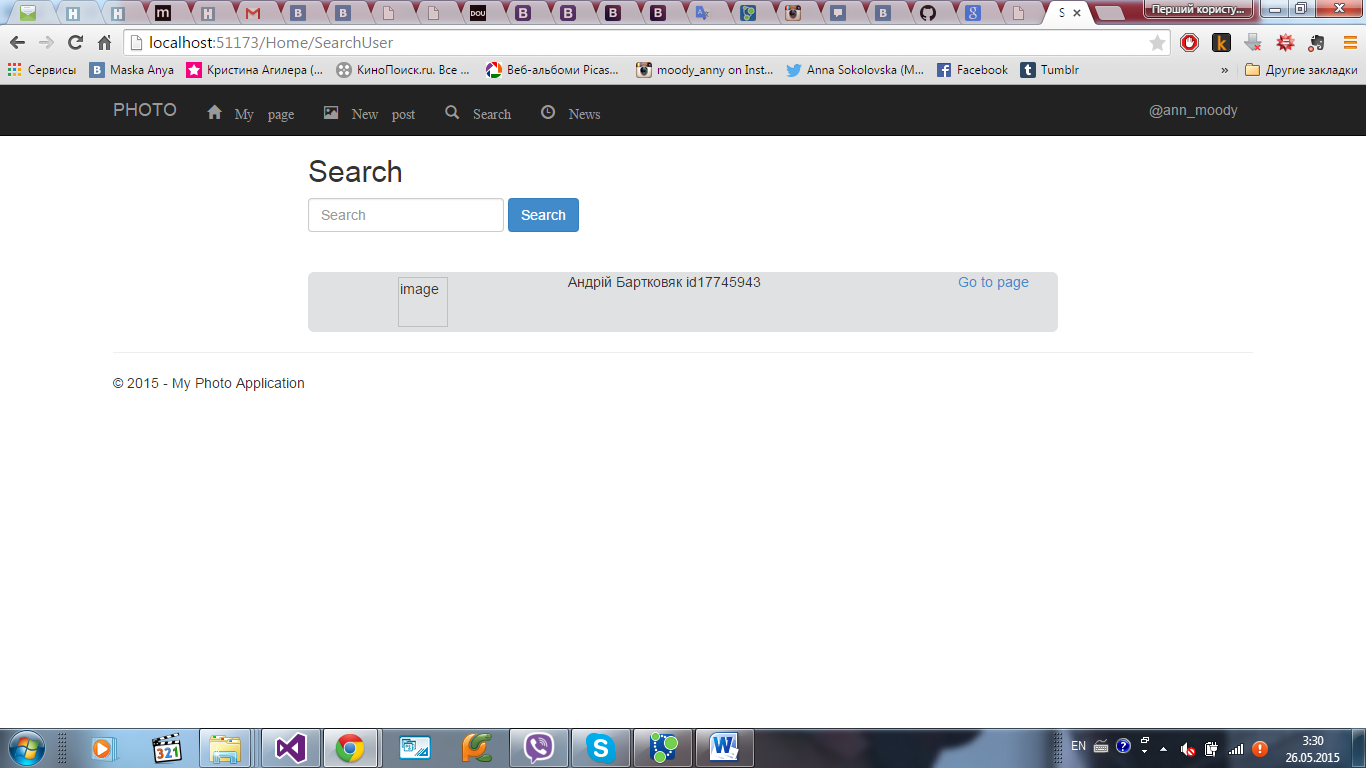
Натиснувши на «Новини» у навігаційному полі вгорі сторінки ми отримаємо дані про активність користувачів, за якими слідуємо:



*Рис. 9.* Сторінка новин

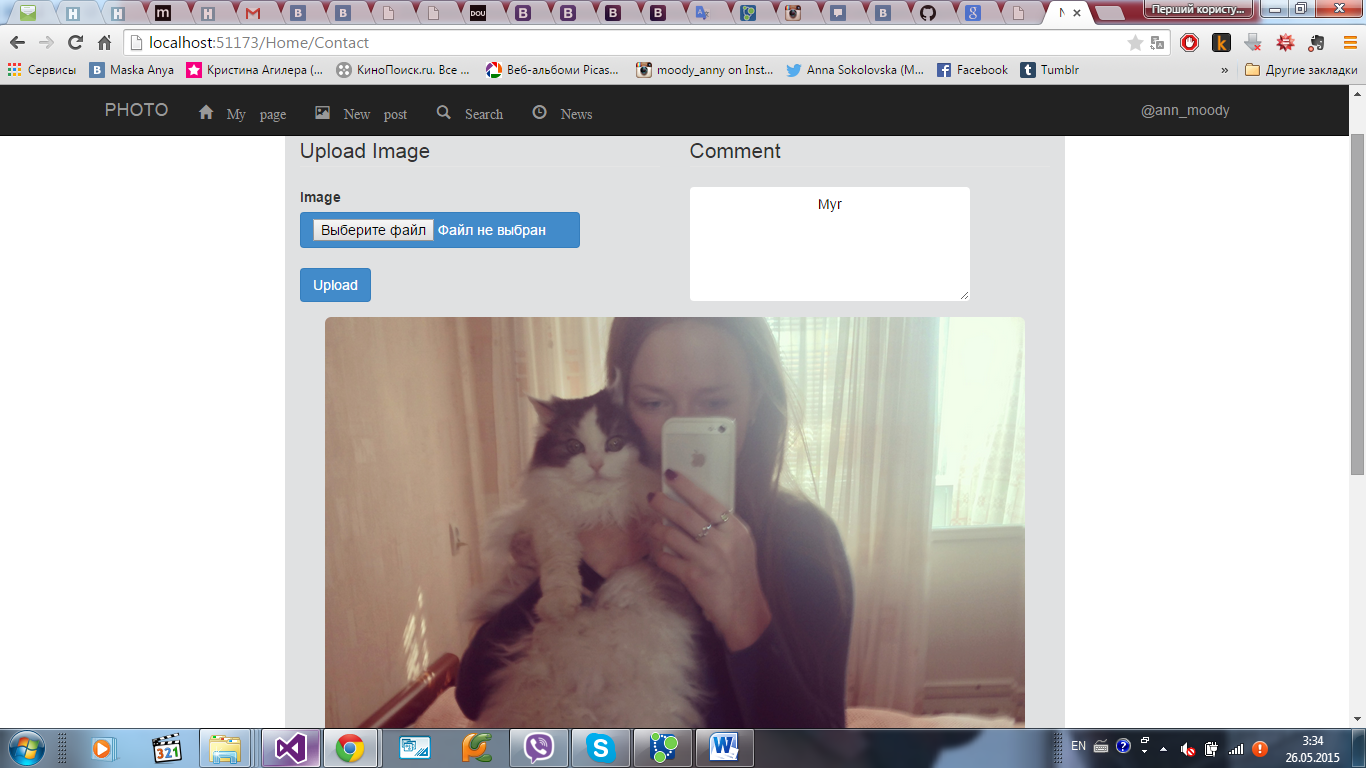
Можемо знайти користувача:





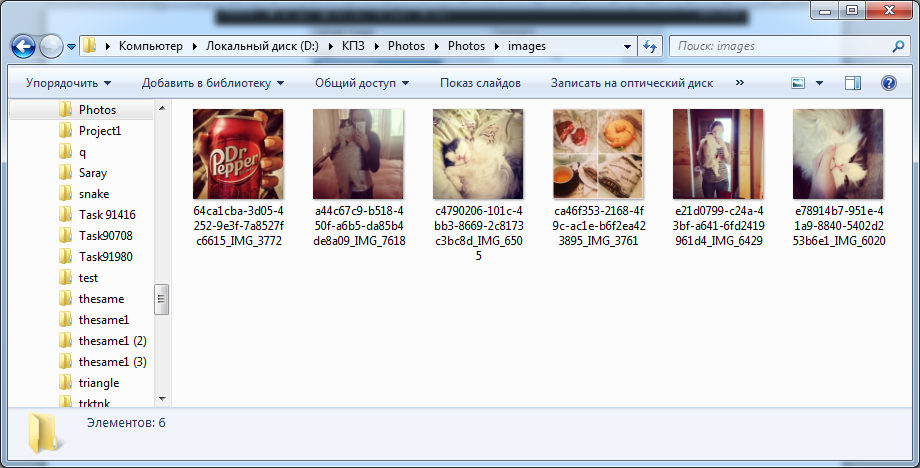
*Рис. 10.* Пошук користувача

Створення нового посту виглядає так:



*Рис. 11.* Додавання посту

Фотографії копіюються з комп’ютера у директорію проекту сайту з унікальним ім’ям. При видалення посту фото теж видаляється з даної директорії:



*Рис. 12.* Збереження фото

**Висновки**

Метою даної курсової роботи було розроблення сайту для обміну фото з соціальною складовою. Підставою для розроблення стало завдання на виконання курсової роботи з дисципліни «Архітектура та проектування програмного забезпечення» студентами ІІІ курсу кафедри програмного забезпечення комп’ютерних систем Національного технічного університету України «КПІ».

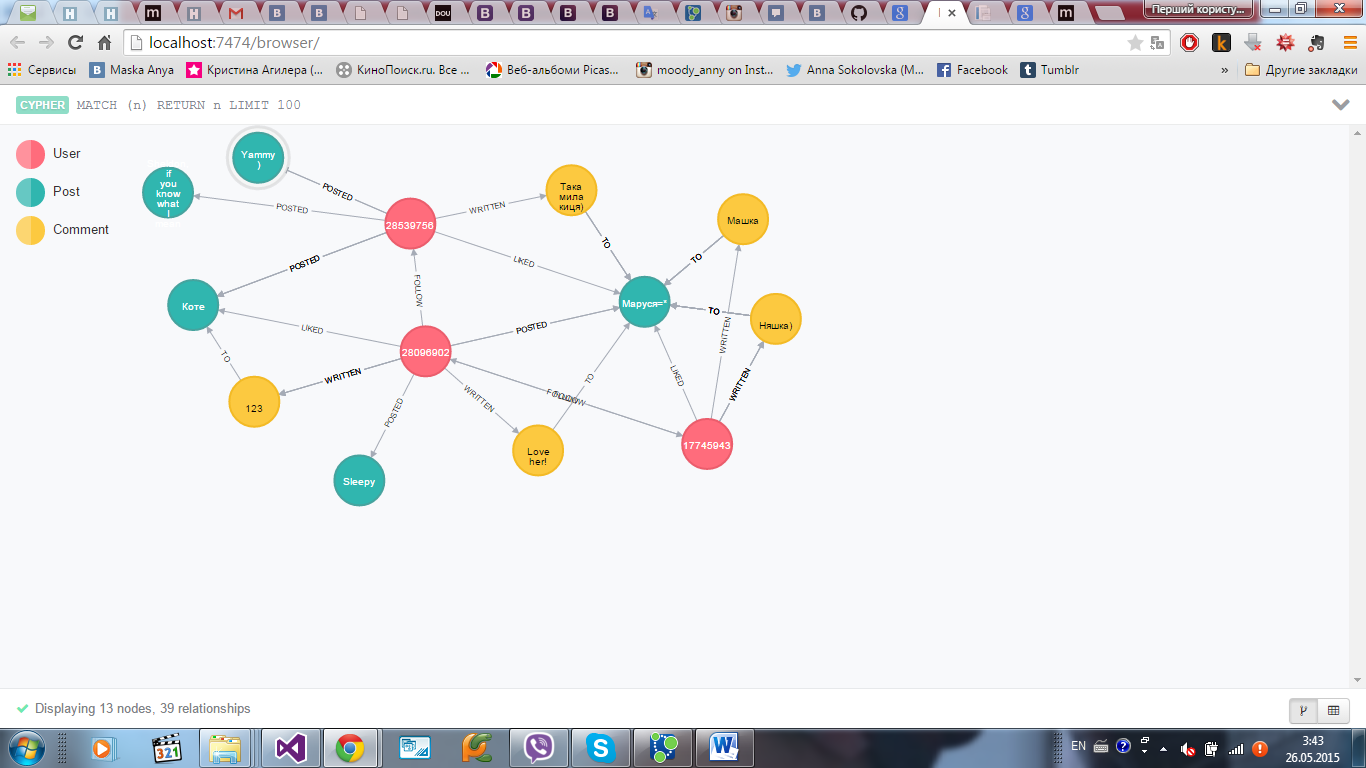
Для досягнення поставленої мети у повному обсязі виконано завдання, визначені у аркуші завдання на курсову роботу; розроблено графічні матеріали; реалізовано всі вимоги до програмного продукту, наведені у технічному завданні; створено відповідну документацію.

Перспективним напрямком подальшого дослідження даної тематики є доопрацювання сайту і додавання до нього нового функціоналу (надсилання повідомлень, автентифікація через інші соціальні мережі).

**Список використаних джерел**

1. ASP.NET MVC on MSDN [Електронний ресурс] – Режим доступу: https://msdn.microsoft.com/ru-ru/library/dd381412%28v=vs.108%29.aspx
2. Сypher examples for Neo4j Client / Tatham Oddie [Електронний ресурс] – Режим доступу: https://github.com/Readify/Neo4jClient/wiki/cypher-examples
3. Manual foe Neo4j [Електронний ресурс] – Режим доступу: http://neo4j.com/

**Додатки**



**Додаток А. Приклад збереження даних в Neo4j**