**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 сервісу анонімного листування – “AirMail”»**

**Студент групи** КП-21 **Литвин Вадим Олександрович** \_\_\_\_\_\_\_\_\_\_

(підпис)

**Викладач: Мелащенко А. О. \_\_\_\_\_\_\_\_\_\_**

(підпис)

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

Київ – 2015

**Анотація**

В курсовій роботі основною метою було створення сервісу анонімного листування, який отримав назву "AirMail". Анонімність листування простежується лише на стороні клієнта. Було взято за мету реалізувати сайт та додаток для Android-девайсів, що дозволяв би користуватися даним сервісом. В ході проведення роботи та аналізу предметної галузі, в якості бази даних була обрана MySQL. Основа проекту Django (Джанго) — високорівневий відкритий Python-фреймворк для розробки веб-систем, з застосування Rest API фреймворку - Tastypie, який необхідний для забезпечення обміну інформацією між сервером та Android-девайсом.

Під час реалізації сервісу було закріплено та набуто нові знання в роботі з БД MySQL, написання сучасного програмного забезпечення для роботи з даною БД. Також було отримано та закріплено нові навички при розробці Android-додатків та розробки сайтів і їх дизайн за допомогою foundation.zurb. Для відображення певної статистики було обрано скористатися рішенням Flotr2.

В результаті було отримано сервіс, сайт та android-додаток, який можна впроваджувати в публічне використання.

Посилання на репозиторій GitHub:

**https://github.com/VadyaVL/Coursework\_AirMail\_Lytvyn**

**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 8

3. Тестування програмного забезпечення 11

Висновки 15

Список використаних джерел 16

**Вступ**

Дана курсова робота присв'ячена розробці сервісу анонімного листування, який включає в себе розробку мобільного додатку для Android та сайту, використовуючи фреймворк Django.

*Об’єктами* курсової роботи є мобільний додаток та сайт.

*Метою роботи* є розроблення ПЗ додатку 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 4-х структурних частин: бази даних користувачів, веб-серверу, мобільного клієнта для Android та web клієнта на Django.

Для функціонування сервісу потрібна наявність MySQL-Server'у, фреймворку django, Rest API tastypie та забезпечення системних умов для його функціонування.

Для функціонування мобільного клієнту потрібний мобільний пристрій на платформі Android, а також наявність доступу до мережі Інтернет (для відправлення, отрмання повідомлень).

Розроблене ПЗ може бути використане з розважальною метою для спілкування з невідомим людьми і з можливим подальшим знайомством.

Пояснювальна записка складається зі вступу, двох розділів, загальних висновків, списку використаних джерел (4 найменування). Загальний обсяг роботи – 16 друкованих сторінок.

1. **Опис реалізованого програмного забезпечення**

В ході реалізації курсової було розроблено сервіс анонімного листування “AirMail”, сайт та android-додаток.

Користувач має можливість реєстрації, після якої він має свій власний акаунт, за допомогою якого він може авторизуватися в сервісі через сайт чи Android-додаток. При авторизації, а також після неї в налаштунках, користувач має змогу змінити аватарку свого профілю, яка є необхідною.

Після авторизації в системі користувач може одразу відправляти анонімні листи, у відповідь він отримує анонімний лист від когось іншого, якщо такий є в базі даних. Відповівши на даний лист, він надсилається назад до того хто створив даний діалог, і переписку можна продовжувати до безкінечності.

На сайті відображається статистика користувача, яка містить інформацію про кількість надісланих повідомлень, і кількість створених ним діалогів, а також загальна статистика по всім користувачам.

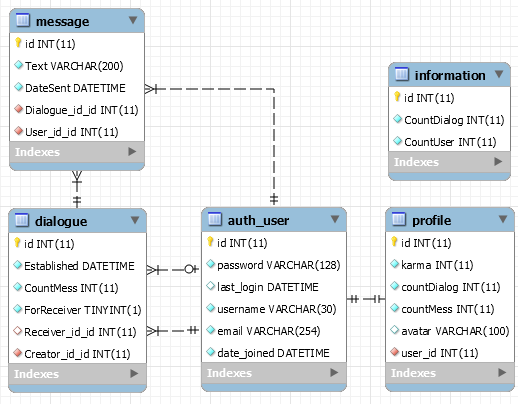
За допомогою Flotr2.js статистика конкретного користувача відображається у вигляді кругової діаграми. Android-додаток реалізує лише основний функціонал сервісу, тобто відправку повідомлень, їх отримання, авторизацію та реєстрацію в системі..

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

У створеному та використовуваному програмному забезпечені можна виділити п’ять структурних елементів, що взаємодіють між собою.

база даних сервісу, сервер, мобільний клієнта та web клієнт.

* Реляційна БД MySQL, де зберігається вся необхідна інформація сервісу, акаунти користувачів, їх профайли, повідомлення, діалоги.
* web клієнт – сайт з дизайном foundation - front-end framework.
* веб-сервер на Django, який працює як Rest сервер для спілкування з мобільними пристроями за допомогою tastypie та сервер для web додатку
* клієнт на Android-девайси.
  1. **Опис модулів програмного забезпечення**

Розглянемо моделі даних, що зберігаються в БД. На рисунку відображена загальна схема бази даних.

Веб-сервер реалізований за допомогою фреймворку Django, що виявився досить зручним для використання його в поставленій задачі. Для надання можливості мобільному додатку спілкуватися з сервером, було вирішено використовувати tastypie, який в парі з Django був легким інструментом для реалізації Rest API – обміну мобільного клієнта та сервера інформацією за допомогою json.

Всі моделі предметної області описані на сервері як Django-моделі(models.Model), на сонові яких було описано Rest-ресурси, що мають дозвіл на всі операції – читання, видалення, редагування, додавання, проте мають обмеження, яке базується на тому, що дані операції можуть виконувати лише авторизовані користувачі. Користувачі при запиті певного Rest-ресурсу отримують лише власні Rest-дані.

Для сутності користувач було вирішено додати модель – Profile, замість того щоб перевизначати стандартну Django-User модель. В ній містить інформація про аватар користувача, кількість надісланих листів, кількість створених діалогів, а також показник “Карма”, який збільшується при відправці кожного 5-го повідомлення в одному діалозі конкретним користувачем. Використання даного показника, не придумано – планується використання при впровадженні сервісу в публічний доступ.

Мобільний додаток реалізовано на мові програмування java з залученням Android SDK, розробка хорошого дизайну передбачається перед впровадженням в публічний доступ.

* 1. **Опис основних алгоритмів роботи**

Для того, щоб запобігти накопичення великої кількості неактивних діалогів, було вирішено реалізувати на MySQL-Server подію. Подія виконується кожні два дні о третій годині ночі. Суть полягає в тому, що з системи видаляються ті діалоги в яких присутні менше шести повідомлень, і дата створення останнього більше ніж два дні. При впровадженні до публічного доступу подія потребує вдосконалення, наприклад, видалення діалогів, в яких присутні більше п’яти повідомлень, і останнє було створено більше місяця тому.

|  |
| --- |
| delimiter |  CREATE EVENT IF NOT EXISTS `DeleteInactive`  ON SCHEDULE  EVERY 2 DAY STARTS DATE\_FORMAT(NOW(), '%Y-%m-%d 03:00:00')  DO  BEGIN  SET SQL\_SAFE\_UPDATES = 0;  DELETE FROM airmaildb.message WHERE Dialogue\_id\_id IN (  SELECT id FROM airmaildb.dialogue WHERE  TIMESTAMPDIFF(DAY, dialogue.Established,NOW())>2 and dialogue.CountMess<6);  DELETE FROM airmaildb.dialogue WHERE  TIMESTAMPDIFF(DAY, dialogue.Established,NOW())>2 and dialogue.CountMess<6;  END |  delimiter ; |

Наступним чином реалізоване додавання нового діалогу(надсилання нового повідомлення). Після відправлення повідомлення воно фіксується в базі як “вільне” – може бути комусь в подальшому надіслане. Натомість в базі відбувається пошук вільного повідомлення і присвоєння його поточному користувачеві.

|  |
| --- |
| **def** home(request):  args = {}  args[**'user'**] = auth.get\_user(request)  args[**'info'**] = getInformation()  args[**'profile'**] = getProfile(args[**'user'**])  **if** type(args[**'user'**]) != AnonymousUser:  args[**'MessForm'**] = forms.MessageForms()   **if** request.method == **'POST' and** args[**'user'**] **is not** None:  mess\_form = forms.MessageForms(request.POST)  **if** mess\_form.is\_valid():  dialogue = Dialogue(CountMess=1, ForReceiver=True)  dialogue.Established = datetime.now()  dialogue.Creator\_id = args[**'user'**]  dialogue.Receiver\_id = args[**'user'**]  dialogue.save()  updInformation(**'cMess'**)  mess = mess\_form.save(commit=False)  mess.DateSent = datetime.now()  mess.Dialogue\_id = dialogue  mess.User\_id = args[**'user'**]  mess\_form.save()   pr = getProfile(request.user)  pr.countMess += 1  pr.countDialog += 1  pr.save()   *#отримуємо рандомне. тут і умову отримання пропишемо* NewMessage = Dialogue.objects.filter(Q(CountMess=1) & ~Q(Creator\_id=args[**'user'**]) & Q(Receiver\_id=F(**'Creator\_id'**)))   **if** len(NewMessage)!=0:  i = random.randint(0, len(NewMessage)-1)  NewMessage = NewMessage[i]  NewMessage.Receiver\_id = args[**'user'**]  NewMessage.save()  *##################################* args = {}  args[**'info'**] = getInformation()  args[**'dialog'**] = NewMessage  args[**'MessForm'**] = forms.MessageForms()  args[**'user'**] = auth.get\_user(request)  args[**'profile'**] = getProfile(args[**'user'**])   messages = Message.objects.filter(Q(Dialogue\_id=args[**'dialog'**]))  current\_page = Paginator(messages, 10)  args[**'messages'**] = current\_page.page(current\_page.num\_pages)   **return** render(request, **'viewDialog.html'**, args)  **else**:  **return** redirect(**'/'**)  **else**:  args[**'MessForm'**] = mess\_form  **else**:  args[**'mess'**] = notLoginAtHomePage  **return** render(request, **'home.html'**, args) |

Ще одним із основних моментів розробки сервісу була реалізація мобільного додатку, який має змогу спілкуватися з сервером за допомогою json – Rest API. Наведемо код авторизації в системі через мобільний додаток.

|  |
| --- |
| **public** **static** **void** login() {  **boolean** ok = **false**;  String loginURL; // Одночасно залогінюємося і беремо профайл юзера  // Тут перевірка залогінення  loginURL = "http://" + AirMailService.*ip* + ":" + AirMailService.*port*  + "/" + AirMailService.*urlKey*  .get(AirMailService.AirMailObjects.***PROFILE***);    JSONParser jsp = **new** JSONParser();  Log.*e*("JSON", loginURL);  JSONObject jsO = jsp.getJsonProfile(loginURL);  JSONArray arr;  **try** {  arr = jsO.getJSONArray("objects");  *profile* = **new** ProfileProvider(arr.getJSONObject(0));  ok = **true**;  } **catch** (JSONException e) {  e.printStackTrace();  }  // /////////////////////////  **if** (ok) {  *logged* = **true**;  Log.*d*("Логін", "Залогінено");  } **else** {  *logged* = **false**;  Log.*d*("Логін", "Не залогінено");  }  } |

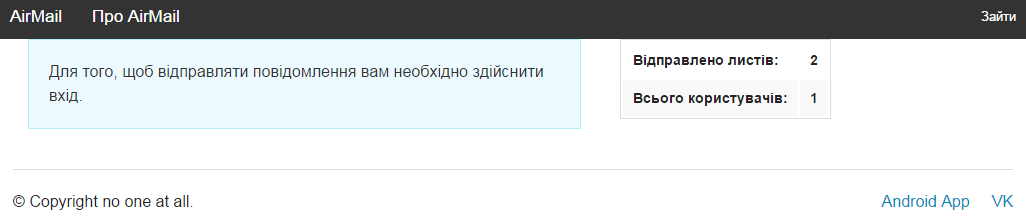
У випадку успішної авторизації, ми отримуємо профайл даного користувача. Поки що лише він слугує “прапорцем” успішної авторизації в системі.

Профайл використовується для оновлення статистики користувача через мобільний додаток. Наведемо код реалізації getJsonProfile.

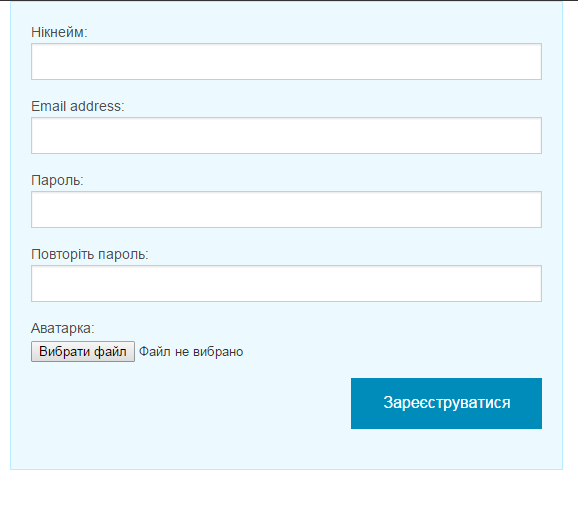
|  |
| --- |
| **public** JSONObject getJsonProfile(String url) {  JSONObject jsO = **null**;  HttpClient httpClient = **new** DefaultHttpClient();  HttpContext localContext = **new** BasicHttpContext();  HttpGet httpGet = **new** HttpGet(url);  String text = **null**;  **try** {  String credentials = AirMailService.*name* + ":" + AirMailService.*password*;  String base64EncodedCredentials = Base64.*encodeToString*(  credentials.getBytes(), Base64.***NO\_WRAP***);  httpGet.addHeader("Authorization", "Basic "  + base64EncodedCredentials);  HttpResponse response = httpClient.execute(httpGet, localContext);  HttpEntity entity = response.getEntity();  text = getASCIIContentFromEntity(entity);  jsO = **new** JSONObject(text);    } **catch** (Exception e) {  Log.*e*("va0", e.toString());  }    **return** jsO;  } |

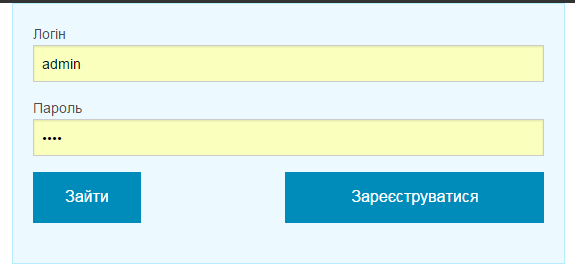
1. **Тестування програмного забезпечення**

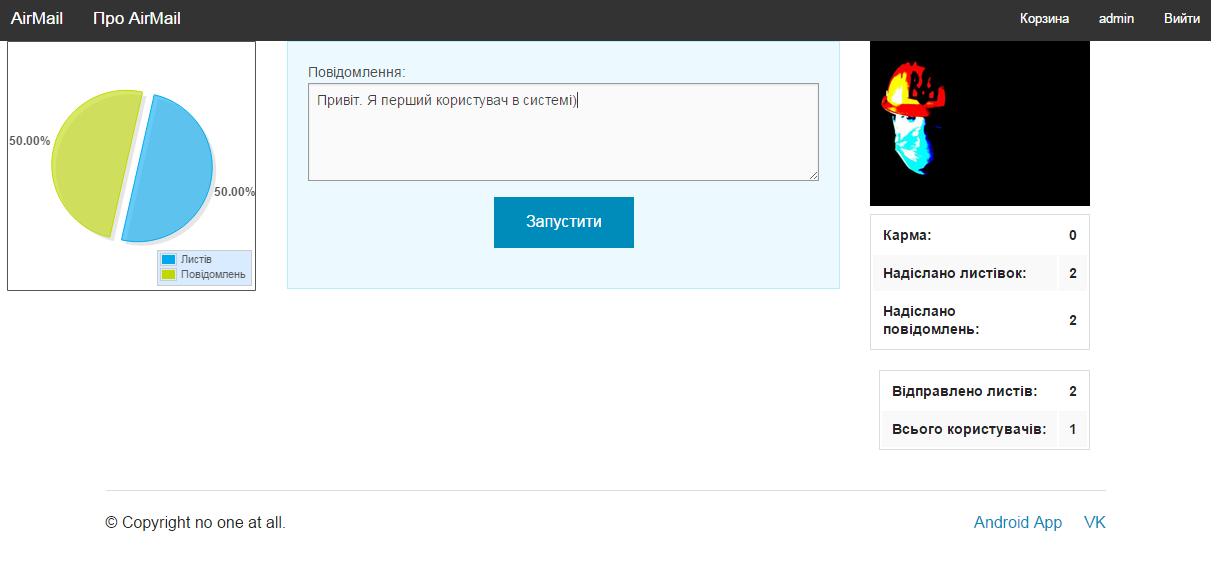
При запуску web додатку користувач потрапляє на головну сторінку, де він може перейти до входу чи реєстрації в системі. Користувачеві відображається повідомлення, що необхідно здійснити вхід.



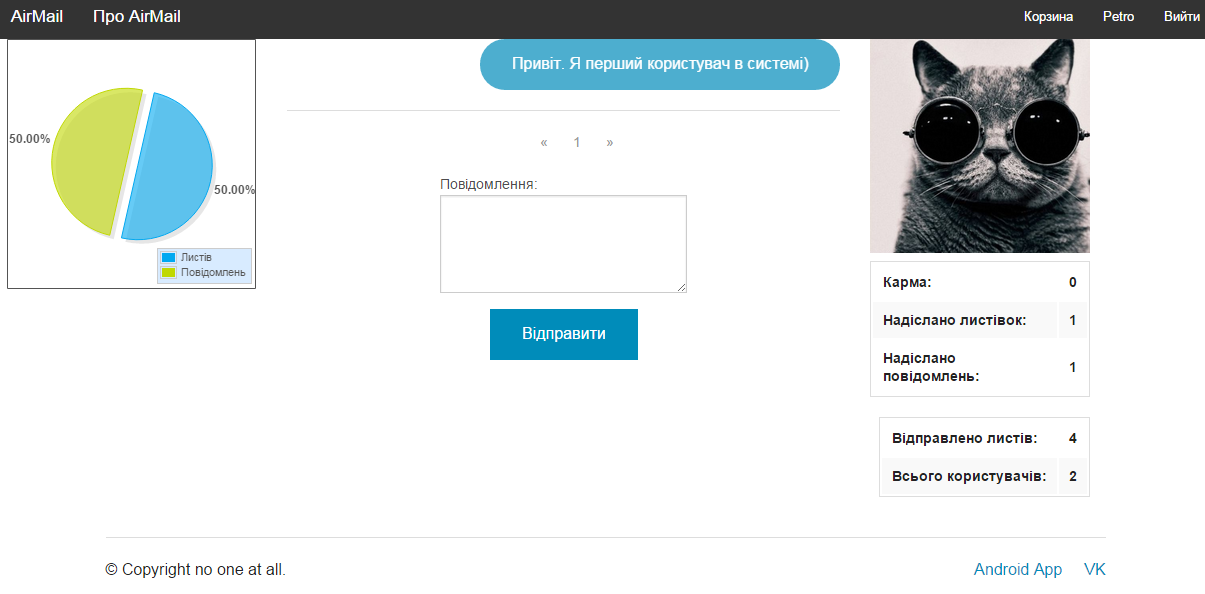
Покажемо скріншот форми реєстрації, та здійснимо вхід в систему, а також відправимо повідомлення.



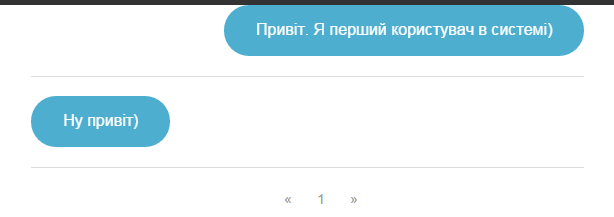




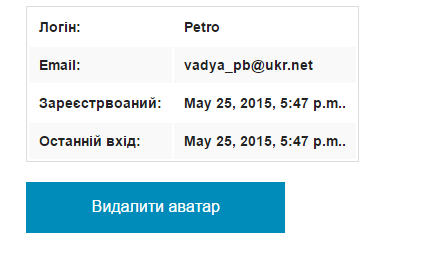
Після надсилання повідомлення у відповідь нічого не отримано. Здійснимо реєстрацію та авторизацію нового користувача, і відправимо від нього нове повідомлення.



Як бачимо ми отримали відповідь від першого користувача. Після надання йому відповіді ми більше нічого не можемо робити. Залишається лише очікувати відповідь від першого користувача.



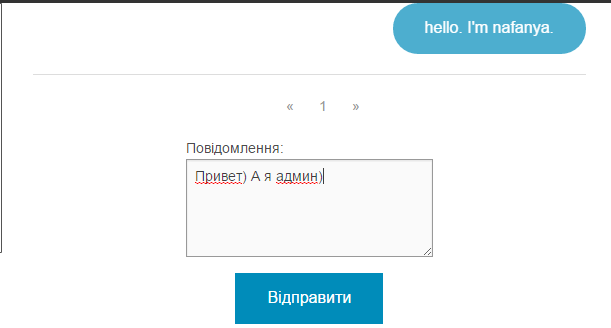
Для кінця відобразимо інформацію про поточного користувача.



Перейдемо до тестування андроїд додатку.

|  |  |  |
| --- | --- | --- |
| C:\Users\Vadym\Desktop\Курсова\pic8.PNG | C:\Users\Vadym\Desktop\Курсова\pic9.PNG | C:\Users\Vadym\Desktop\Курсова\pic10.PNG |
| Спроба зареєструвати нового користувача виявилася успішною. | Залогінившись можна переглянути листи на які можна відповісти, написати новий лист, або вийти з системи. | Ось так виглядає надсилання листа. |

Як видно з малюнку нижче, повідомлення успішно дійшло до сервера, і ми його проглядаємо з іншого акаунту через web-додаток.



|  |  |
| --- | --- |
| C:\Users\Vadym\Desktop\Курсова\pic12.PNG | C:\Users\Vadym\Desktop\Курсова\pic13.PNG |
| Ось так проглядаються доступні для відповіді діалоги на мобільному додатку. | Ось так власне відображається сам діалог. |

**Висновки**

Метою даної курсової роботи було розроблення сервісу анонімного листування “AirMail”. 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».

Було реалізовано повністю робочий сервер з можливістю Rest API обміну, web-клієнт, та android-клієнт на java. При тестуванні обох клієнтів недоліків помічено не було. Все працює коректно. Інтерфейс web-клієнту зручний, приємний на око, має зручну пагінацію.

Перспективним напрямком подальшого дослідження даної тематики є доопрацювання додатку та її випуск на Google Play та на іншх ресурсах, тобто впровадження сервісу в публічний доступ.

**Список використаних джерел**

1. https://django-tastypie.readthedocs.org
2. https://developer.android.com
3. http://www.humblesoftware.com/flotr2
4. Google Android. Создание приложений для смартфонов и планшетных ПК. Алексей Голощапов. Санкт-Петербург 2013.