Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук

Кафедра програмної інженерії

КУРСОВА РОБОТА  
пояснювальна записка

«Програмна система для організації черг у офісах»

Студент гр. ПІ-13-3 Задорожній С.А.

Керівник роботи: доц. каф. ПІ Лещинський В.О.

Роботу захищено «\_\_»\_\_\_\_\_\_2016 р.

з оцінкою \_\_\_\_\_\_\_\_\_\_\_\_\_

Комісія:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Харків

2016 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук

Кафедра програмної інженерії

Напрям підготовки Програмна інженерія

Курс 3 . Семестр 6 .

Навчальна дисципліна Аналіз та рефакторинг коду програмного забезпечення .

ЗАВДАННЯ  
НА КУРСОВУ РОБОТУ СТУДЕНТОВІ

*Задорожньому Станіславу Андріановичу .*

1. Тема роботи: *«Сервіс для організації черг у офісах»*

*.*

2. Термін узгодження завдання курсової роботи «*04*»\_\_*лютого*\_\_\_\_\_\_\_\_\_ 2016 р.

3. Термін здачі студентом закінченої роботи «*13*»\_\_\_*червня\_\_\_\_\_\_ \_\_* 2016р.

4. Вихідні дані до проекту (роботи): *Використовувати ОС Windows, середовище об'єктно-орієнтованого проектування Microsoft Visual Studio 2015 Community*

*.*

*.*

*.*

*.*

5. Зміст пояснювальної записки *(перелік питань, що належить розробити) вступ, аналіз предметної області, постановка задачі, моделювання програмного продукту, опис інтерфейсу та функціоналу, висновки, перелік посилань.*

*.*

*.*

6. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)   
*діаграма класів, діаграма прецедентів, діаграма послідовностей, діаграма станів, діаграма компонентів, діаграма кооперації.*

*.*

*.*

*.*

*.*

КАЛЕНДАРНИЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва етапів курсової роботи | Термін виконання етапів проекту (роботи) | Примітка |
| 1 | Функціональна специфікація програми | 19.03.2016 | виконано |
| 2 | Проектування програми | 16.04.2016 | виконано |
| 3 | Кодування програми | 27.04.2016 | виконано |
| 4 | Оформлення пояснювальної записки | 21.05.2016 | виконано |
| 5 | Захист курсової роботи |  | виконано |

Дата видачі завдання «»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2016 р.

Керівник роботи \_\_\_\_\_\_\_\_\_\_\_ к.т.н., доц. каф. ПІ Лещинський В.О.

Завдання прийняв до виконання

студент гр. ПІ-13-3 \_\_\_\_\_\_\_\_\_\_\_ Задорожній С. А.

РЕФЕРАТ

Пояснювальна записка до курсової̈ роботи: 26 с., 12 рис.,   
5 джер.

Метою роботи є проектування та розробка програмної системи для відстеження стану кавоварки і організації черги для приготування напою.

Методи розробки базуються на технології̈ C#, MS SQL базі даних, середовищі розробки Visual Studio 2015 та Android Studio.

Результатом роботи є програмна система, що складається з серверу, веб-клієнта, мобільного додатку та емулятору пристрою, що є спрощеною моделлю розумної кавоварки.

С#, .NET, WCF, IOT, МОБІЛЬНИЙ ДОДАТОК, ANDROID, ASP.NET MVC, ПРОГРАМНА СИСТЕМА.

ЗМІСТ

[ВСТУП 6](#_Toc452700079)

[1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 7](#_Toc452700080)

[2 ПОСТАНОВКА ЗАДАЧІ 9](#_Toc452700081)

[3 МОДЕЛЮВАННЯ ПРОГРАМНОГО ПРОДУКТУ 11](#_Toc452700082)

[4 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДУКТУ 16](#_Toc452700083)

[5 ІНТЕРФЕЙС ТА ФУНКЦІОНАЛ 19](#_Toc452700084)

[5.1 Сервер 19](#_Toc452700085)

[5.2 Веб-клієнт 20](#_Toc452700086)

[5.3 Додаток для Android 24](#_Toc452700087)

[ВИСНОВКИ 25](#_Toc452700088)

[ПЕРЕЛІК ПОСИЛАНЬ 26](#_Toc452700089)

# ВСТУП

Кожен співробітник офісу кілька разів за день п'є чай або каву. І якщо в офісі дуже багато співробітників, а автомат з кавою всього один на поверх, то виникають великі черги, і співробітники просто чекають на кухні і нічого не роблять.

В основі проекту лежить «розумний» пристрій, що дозволить відстежувати стан кавеварки. Він буде пов’язаний з сервером та відправлятиме повідомлення про зміни. Цей пристрій працює за концепцією Internet of Things.

Інтернет речей (Internet of Things) − єдина мережа, що об’єднує техніку, якою ми користуємося щодня, та віртуальний світ. Технологія не лише дозволяє віддалено керувати різними приладами, а й пов’язує їх між собою. Обмінюючись даними, речі починають «спілкуватися» один з одним.

Ця ситуація може бути втілена в життя в тому випадку, якщо побутові прилади та інші електронні пристрої будуть підключені до Інтернету і навчаться обмінюватися між собою інформацією, виконувати команди, відправлені з інших пристроїв – компонентів цієї глобальної мережі. Реалізація цієї концепції отримала назву Інтернету речей – «Internet of Things». Вперше термін був запропонований в 1999-му році. А кількісний перехід від «Інтернету людей» до «Інтернету речей» відбувся в 2008-2009 році: саме в той період кількість пристроїв, підключених до Інтернету, перевищила кількість інтернет-користувачів.

В наш час ця технологія стає дедалі популярнішою, наприклад такі гіганти, як Apple чи Google виробляють системи для керування «розумним будинком».

Футуристи і фахівці в технологіях називають Інтернет речей проміжним етапом на шляху до нового витка розвитку технологій – до мережі чергового покоління, названого «Інтернетом всього». У цій мережі просто не буде існувати пристроїв, що працюють в режимі «офлайн», а Інтернет буде те саме що електрика – без нього просто не зможе функціонувати жодний електронний пристрій. Сьогодні складно собі уявити, яким буде наше життя в «Інтернеті всього», проте вже зараз можна припустити, що сховатися від всевидющого ока такої воістину глобальної мережі буде практично неможливо.

Отже тема та засоби для виконання курсової роботи є актуальними. Метою курсової роботи є проектування та створення програмної системи, яка дозволить офісах організувати «розумні черги» для очікування приготування своїх напоїв, що істотно підвищить продуктивність в офісі. А у подальшому може вдосконалитись не тільки для приготовунная напоїв, а і для інших різноманітних засобів.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Як було зазначено вище обрана тема є актуальною на сьогодні і на близьке майбутнє. Щоб проаналізувати дану предметну область треба сказати про засоби, що все використовуються, їх плюси та мінуси.

Якщо казати про конкуренцію на ринку в даній сфері, то можна сміливо заявити, що наразі немає жодного аналогу, який міг би робити щось подібне.

Отже, ця програмна система має функціонал, якому немає аналогів. Якщо казати про те, кому саме може знадобитись додаток, то можна впевнено сказати – будь-якому більш-менш великому офісу.

Розробляєма система дозволяє оптимізувати роботу в офісі: співробітникам не потрібно буде чекати поки приготується їх напій і чекати своєї черги. Завдяки даній системі співробітники зможуть віддалено зробити замовлення і отримати повідомлення, коли їх напій буде готовий. Це дозволить зменшити час, що витрачається на цю операцію.

Основною бізнес-метою є розробка гнучкого застосування для великих офісів з можливістю легкого підключення нових офісів, а так само з можливістю подальшого розвитку і підключення нових пристроїв. Критерії успіху базуються на якості розробленого сервісу, а саме на реалізації більшого числа необхідних функцій і зручного інтерфейсу для користувачів.

Дана система буде користуватися популярністю у великих офісах, де виникає проблема з очікуванням своєї черги. В першу чергу в цьому додатку будуть зацікавлені директори офісів. Ця програма допоможе їм оптимізувати роботу їх співробітників в офісі і прискорити виконання рутинних дій.

Основним бізнес-ризиком є – поширення програми. Так як установка даної системи вимагає додаткового обладнання, то цей крок може відлякати потенційних клієнтів. Тому необхідно вводити різні акції для швидкого поширення продукту. Слід додати, що проект не має фінансової підтримки і це може відбитися на швидкості його поширення і зміцнення на ринку освітніх проектів. Також бізнес-ризиком є –складність в реалізації подібного сервісу через нестачу знань, пов'язаних з реалізацією такого масштабного програмного продукту в стислі терміни. Брак досвіду в розробці подібних серйозних масштабованих додатків може також відбитися на швидкості і якості продукту.

Даний продукт може залучити користувачів своєю простотою і зручністю, а так само можливістю не витрачати свою час на пусте очікування.

Отже, в цьому розділі було розглянуто предметну область. Можна підсумувати, що була обрана проблемна предметна область, яка потребує саме цього продукту, адже наразі немає жодного аналогу. Розглянуті приклади підтвердили, що ринок потребує цей продукт і він буде корисним дуже багатьом людям і навіть може використовуватися у великих масштабах. Враховуючи, що продукт буде розширюваний, його можна змінювати у залежності від потреб ринку та галузі використання.

# 2 ПОСТАНОВКА ЗАДАЧІ

Метою поданої курсової роботи є створення сервісу для керування кавоваркою і організації онлайн черг для її використання.

Так як користувачі найчастіше починають користуватися своїми смартфонами, то необхідно розробити мобільний додаток для пристроїв з операційною системою Android 4.0.x Ice Cream Sandwich або вищою.

Дані користувачів передаються на сервер, який може бути розміщено де завгодно. У якості серверу повинен бути встановлений та налаштований IIS(Internet Information Services). Найкраще для цього підходить хмарне середовище Azure. Потужності хмарного середовища повинні витримувати одночасну обробку кількох тисяч запитів. На сервері баз даних зберігаються дані користувачів у базі даних SQL Server [2], що також розміщується у хмарному середовищі. Дані, оброблені на серверах, передаються власне клієнтам. Мобільний додаток реалізований для подальшого розміщення у Play Market. І так само цілодобово доступний веб-сайт, з якого можна керувати системою

Веб частина повинна мати наступний функціонал: будь-який користувач може зайти в сервіс після авторизації. Якщо він ще не був зареєстрований, йому буде запропонована форма для реєстрації. Але для подальшого входу йому необхідно буде дочекатися актування свого записи уповноваженою людиною. Авторизований користувач має змогу переглядати доступні кавоварки, налаштувати їх для себе та ставити свій запит на приготування у чергу. Також директор фірми має змогу дивитися усіх своїх співробітників та активувати нових. Існуючий адміністратор у системі має змогу додавати нові фірми до системи. Налаштувати для них доступні кавоварки та вразі необхідності блокувати ці фірми.

Мобільна частина повинна мати наступний функціонал: авторизований користувач може переглядати доступні кавоварки та робити замовлення і відстежувати його.

Серверна частина повинна зберігати усіх користувачів системи та налаштування. Вона має надавати REST сервіси для використання веб і мобільною частинами для реєстрації, логіну, отримання кавоварок з бази даних.

В ході планування проекту були обдумані різні варіанти реалізації призначеного для користувача інтерфейсу для підвищеного комфорту користувачів. Так само була продумана ієрархія структур даних так, щоб вони підтримували можливість додавання нових офісів в систему для залучення більшого числа користувачів.

У подальшому планується розширити соціальну та функціональну складову розробленої системи шляхом додавання наступних можливостей:

а) додати можливість спілкуватися у створеній черзі;

б) додати статистику використання кавоварок;

в) розширити сервіс іншими пристроями;

Можлива також розробка мобільного додатку для платформ iOS та Windows Phone.

# 3 МОДЕЛЮВАННЯ ПРОГРАМНОГО ПРОДУКТУ

Для проектування програмної системи було використано засоби проектування Microsoft Visual Studio 2013 та створено наступні діаграми UML:

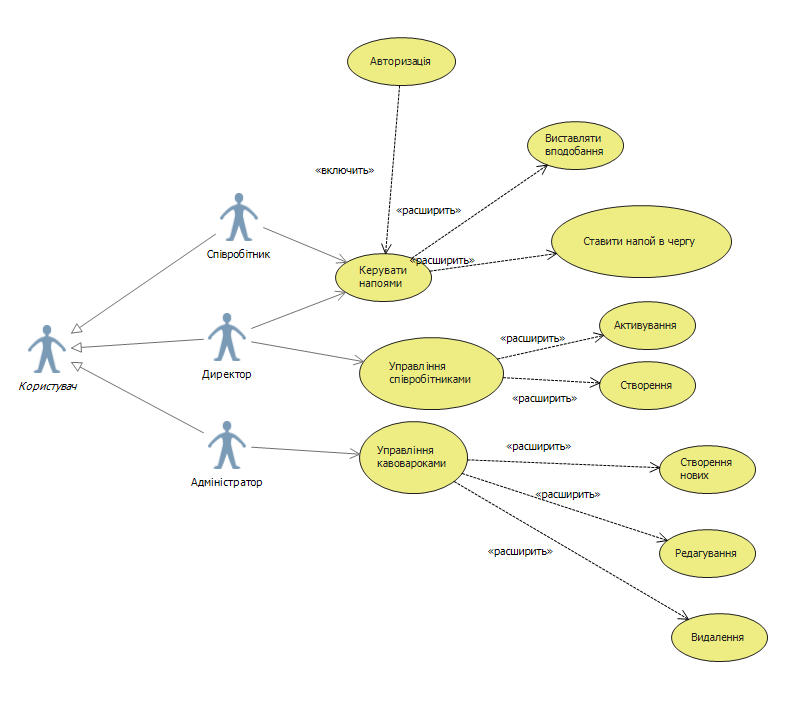
а) Діаграма прецедентів (див. рис. 3.1) має трьох акторів – Співробітника, Директора компанії, Адміністратора. Співробітник – це звичайна людина, яка увійшла у сервіс під своїм логіном. Вона має базовий набір доступних функцій:

* встановлювати свої вподобання для приготування напоїв;
* додавати до черги свій запит для приготування напоїв.

Також у системі присутній користувач з роллю «Директор». Це роль для управлення системою у рамках свого офісу. Директор має можливість створювати облікові записи для своїх співробітників. Або може активувати тільки-но зареєстрованих самостійно користувачів.

Користувач з роллю «Адмін» може робити практично усе для керування сайтом. Він керує компаніями, які доступні на сервері. Для цього він може:

* створювати нові компанії;
* видавати роль директора для певної компанії;
* створювати нові кавоварки та налаштовувати їх для роботи с певним офісом;
* редагувати налаштування для існуючих кавоварок;
* видаляти непотрібні кавоварки із системі.

Рисунок 3.1 – Діаграма прецедентів

б) Діаграма класів (див. рис. 3.2) включає в себе серверні класи CoffeeMachine, як модель кавоварки; Company, як модель компанії, до якої входить список співробітників та список доступних кавоварок; User, як модель користувача, до якого входить номер, ім’я та фамілія користувача; Order, як модель для зберігання замовлень для створення напоїв; CoffeePreferenes, як модель для зберігання налаштувань за замовчуванням для користувача. Для усіх цих моделей розроблено спільну абстрактну модель, яку всі успадковують – BaseEntity, яка зберігає Id для кожного запису.

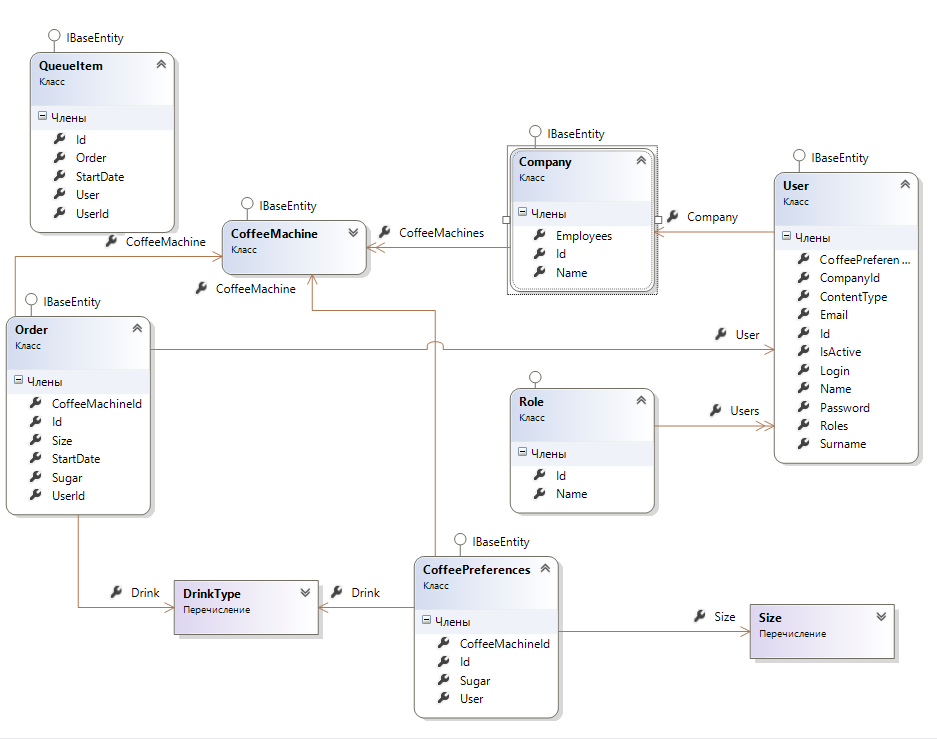


Рисунок 3.2 – Діаграма класів

в) На діаграмі послідовностей (див. рис. 3.3) зображена система в цілому на вищому рівні абстракції та взаємодія основних компонентів – клієнту, сервісу та менеджера бази даних;

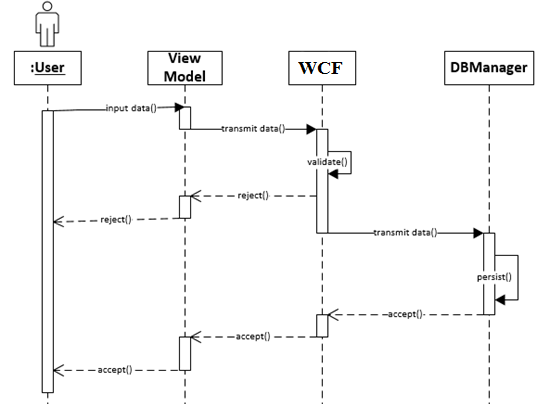


Рисунок 3.3 – Діаграма послідовностей

г) На діаграмі станів (див. рис. 3.4) зображений процес створення, редагування та просмотру компанії адміністратором. Для того, щоб користувачу було доступне створення компанії, він повинен авторизуватися у системі під роллю адміністратора;

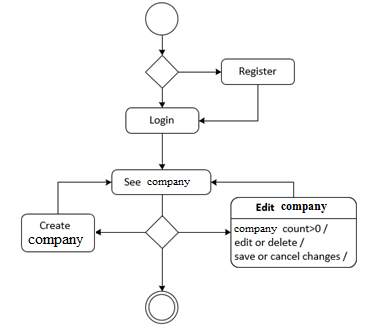


Рисунок 3.4 – Діаграма станів

д) На діаграмі компонентів (див. рис. 3.5) зображені основні частини системи веб-клієнт, мобільний клієнт, основний сервер системи, на якому розмішений сервіс та окремо класи даних, та сервер баз даних;

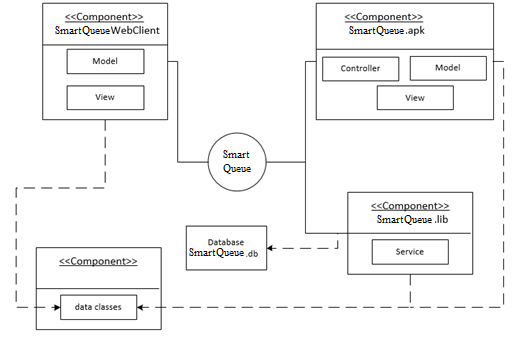


Рисунок 3.5 – Діаграма компонентів

е) На діаграмі кооперації (див. рис. 3.6) зображені стосунки між директором, який створює та надає кавоварку, та користувачем, що використовує її для приготування напоїв.

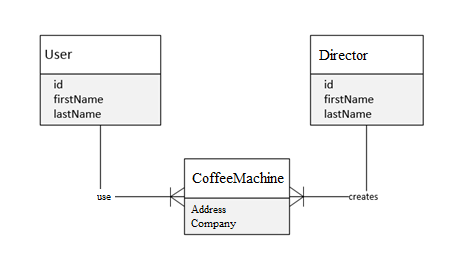


Рисунок 3.6 – Діаграма кооперації

# 4 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДУКТУ

Під час планування та моделювання програмного продукту було визначено, що проект має складатися з трьох частин [4]: сервер, на якому будуть зберігатися дані, до серверу підключаються клієнти за допомогою REST API. На першому етапі реалізується веб-клієнт та мобільний додаток на платформі Android.

Для розробки серверу використовується технологія створення Windows Communication Foundation (WCF) [5]. WCF — набір клієнтських бібліотек, що дозволяють застосункам на базі відкритої платформи .NET Core взаємодіяти з сервісами WCF, відправляючи повідомлення між сервісами в асинхронному режимі. WCF робить можливою побудову безпечних, надійних і транзакційних систем через спрощену уніфіковану програмну модель міжплатформової взаємодії.

Для розробки використовується середовище Visual Studio 2015 та платформа .NET. Для виклику методів сервера використовуються REST[6] контролери з фреймворку ASP.NET MVC, що дозволяє отримувати та відправляти різноформатні дані по протоколу http.

На сьогоднішній день прийнято використовувати REST - (скор. від англ. Representational State Transfer - «передача репрезентативного стану») - метод взаємодії компонентів розподіленого додатка в мережі Інтернет, при якому виклик віддаленої процедури являє собою звичайний HTTP-запит, а необхідні дані передаються як параметри запиту.

У свою чергу HTTP — протокол передачі даних, що використовується в комп'ютерних мережах. Назва скорочена від Hyper Text Transfer Protocol, протокол передачі гіпер-текстових документів.

HTTP — протокол прикладного рівня, схожими на нього є FTP і SMTP. Обмін повідомленнями йде за звичайною схемою «запит-відповідь». Для ідентифікації ресурсів HTTP використовує глобальні URI. На відміну від багатьох інших протоколів, HTTP не зберігає свого стану. Це означає відсутність збереження проміжного стану між парами «запит-відповідь». Компоненти, що використовують HTTP, можуть самостійно здійснювати збереження інформації про стан, пов'язаний з останніми запитами та відповідями. Браузер, котрий посилає запити, може відстежувати затримки відповідей. Сервер може зберігати IP-адреси та заголовки запитів останніх клієнтів. Проте, згідно з протоколом, клієнт та сервер не мають бути обізнаними з попередніми запитами та відповідями, у протоколі не передбачена внутрішня підтримка стану й він не ставить таких вимог до клієнта та сервера

Дані зберігаються у базі даних MS SQL Server, доступ до якої здійснюється за допомогою фреймворку EntityFramework. Microsoft SQL Server — комерційна система керування базами даних, що розповсюджується корпорацією Microsoft. Microsoft SQL Server як мову запитів використовує версію SQL, що отримала назву TRANSACT-SQL (скорочено T-SQL), яка є реалізацією SQL-92 (стандарт ISO для SQL) з багатьма розширеннями. T-SQL дозволяє використовувати додатковий синтаксис процедур, що зберігаються і забезпечує підтримку транзакцій (взаємодія бази даних з керуючим застосунком). Microsoft SQL Server та Sybase ASE для взаємодії з мережею використовують протокол рівня застосунка під назвою Tabular Data Stream (TDS, протокол передачі табличних даних).

Microsoft SQL Server також підтримує Open Database Connectivity (ODBC) — інтерфейс взаємодії застосунків з СУБД. Версія SQL Server 2005 надає можливість підключення користувачів через веб-сервер-сервіси, що використовують протокол SOAP. Це дозволяє клієнтським програмам, не призначеним для Windows, кроссплатформенно з'єднуватися з SQL Server. Microsoft також випустила сертифікований драйвер JDBC, що дозволяє застосункам під керування Java (таким як BEA і IBM Websphere) з'єднуватися з Microsoft SQL Server 2000 і 2005.

SQL Server підтримує дзеркалювання та кластеризацію баз даних. Кластер серверу SQL — це сукупність однаково конфігурованих серверів; така схема допомагає розподілити робоче навантаження між декількома серверами. Усі сервери мають одне віртуальне ім'я, а дані розподіляються за IP-адресами машин кластеру протягом робочого циклу. Також у разі відмови або збою на одному з серверів кластеру доступне автоматичне перенесення навантаження на інший сервер.

SQL Server 2005 має вбудовану підтримку .NET Framework. Завдяки цьому, процедури бази даних, що зберігаються, можуть бути написані на будь-якій мові платформи .NET з використанням повного набору бібліотек, доступних для .NET Framework. На відміну від інших процесів, .NET Framework виділяє додаткову пам'ять і будує засоби керування SQL Server, не використовуючи вбудовані засоби Windows. Це підвищує продуктивність порівняно із загальними алгоритмами Windows, оскільки алгоритми розподілу ресурсів спеціально налагоджені для використання у структурах SQL Server.

В якості системи контролю версій використовується Git. Git — розподілена система керування версіями файлів та спільної роботи. Проект створив Лінус Торвальдс для управління розробкою ядра Linux, а сьогодні підтримується Джуніо Хамано (англ. Junio C. Hamano). Git є однією з найефективніших, надійних і високопродуктивних систем керування версіями, що надає гнучкі засоби нелінійної розробки, що базуються на відгалуженні і злитті гілок. Для забезпечення цілісності історії та стійкості до змін заднім числом використовуються криптографічні методи, також можлива прив'язка цифрових підписів розробників до тегів і комітів.

Для розробки Front end веб-клієнту використовується Java Script, CSS, HTML та ASP.NET MVC, який використовує розроблений спільний API для роботи с данними.

Для розробки мобільного додатку на платформі Android використовується Android Studio та мова Java. Додаток не має бази даних, усі дані, необхідні додатку, отримуються з серверу

Якщо описувати модель взаємодії усіх компонентів системи, то можна зрозуміти, що головною ланкою між усіма клієнтськими додатками є сервер з REST-сервісом. В усіх трьох компонентах системи присутні одні й ті ж класи-моделі для зручності передачі даних.

# 5 ІНТЕРФЕЙС ТА ФУНКЦІОНАЛ

Як вже було неодноразово сказано, програмна система складається з трьох компонентів: серверу, веб-клієнту та мобільного додатку. Під час розробки програмної системи однією з цілей було створити найбільш простий інтерфейс у мобільному додатку, шляхом перенесення більшості функціоналу у веб-клієнт. Тому було вирішено залишити можливість переглядати список своїх маршрутів і визначати своє положення під час поїздки у мобільному додатку, а створення маршрутів, пошук по публічних маршрутах реалізувати у веб-клієнті.

## 5.1 Сервер

На сервері реалізований REST-сервіс, за допомогою якого можна скористатися функціями системи. Методи, що вертають дані, вертають їх у форматі JSON. Наведемо список доступних методів для API:

* GET /coffeeMachines/ – вертає всі кавеварки, зареєстровані в системі;
* GET /coffeeMachines/{id} – вертає кавеварку з ідентифікатором id;
* PUT /coffeeMachines – створює нову кавоварку;
* POST /coffeeMachines – змінює існуючу кавоварку;
* POST /companies/{id}/activate – активує існуючу компанію;
* POST /companies/{id}/ban – блокує існуючу компанію;
* GET /companies/ – отримує усі компанії, підтримую фільтрацію по статусу компанії;
* GET /companies/{id}/employees – отримує усі підприемців із компанії, підтримую фільтрацію по статусу підприємця;
* GET /companies/{id}/coffeeMachines – вертає усі кавеварки, зареєстровані у системі;
* PUT /queue – створює новий запит до черг на приготування напою. Цей метод потім використовує IoT пристрій для створення самого напою;
* DELETE /queue видаляє доданий до черги запис;
* GET /order/{userId} – отримання поточних замовлень користувача;
* GET /order/{userId}/timeLeft – отримує час, який залишився до приготування напою;
* GET /users/{userId}/preference – отримує для заданого користувача його вподобання;
* PUT /users/{userId}/preference – встановлює для заданого користувача його вподобання;
* POST /users/{userId}/preference – змінює для заданого користувача його вподобання;
* GET /users – повертає всіх користувачів, зареєстрованих у системі;
* GET /users/{userId} – отримує заданого користувача;
* PUT /users – реєструю нового користувача;
* POST /users/{userId} – змінює заданого користувача;
* POST /users/{userId}/activate – активує акаунт заданого користувача
* POST /users/{userId}/ban – блокує акаунт заданого користувача

## 5.2 Веб-клієнт

Веб-клієнт є найбільш насиченим з точки зору інтерфейсу. Головна сторінка клієнту являє собою форму, за допомогою якої можна авторизуватися використовуючи свій власний акаунт. Після авторизації користувач потрапляє на головну сторінку (див. рис. 5.1), де він може побачити усі доступні йому операції. Зліва знаходиться навігаційне меню, з якого можна потрапити на сторінку профілю, натиснувши на своє зображення, на сторінку для перегляду усіх черг та на сторінку для реєстрації до нової черги, або на сторінку редагування вподобань користувача.

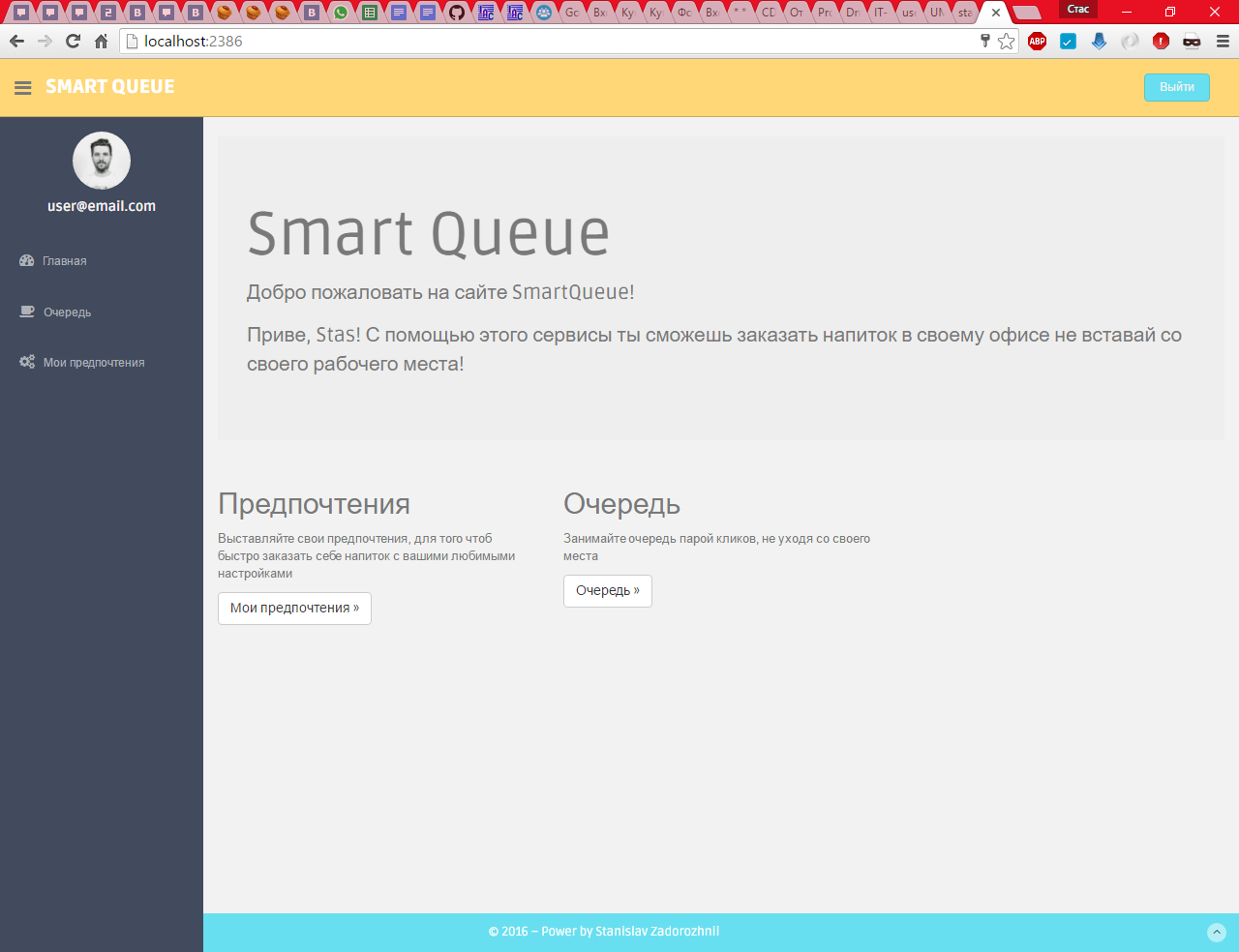


Рисунок 5.1 – Сторінка профілю

На сторінці з моїми вподобанням (див. рис. 5.2) можна настроїти значення за замовченням для майбутнього створення замовлень. Також можна обрати кавомашину, яка буде за замовчення використовуватися для приготування замовлень. Ці данні можна змінити у будь-який момент часу.

Усі поля для даної форми є обов’язковими. Для зберігання вподобань треба натиснути на кнопку «Сохранить».

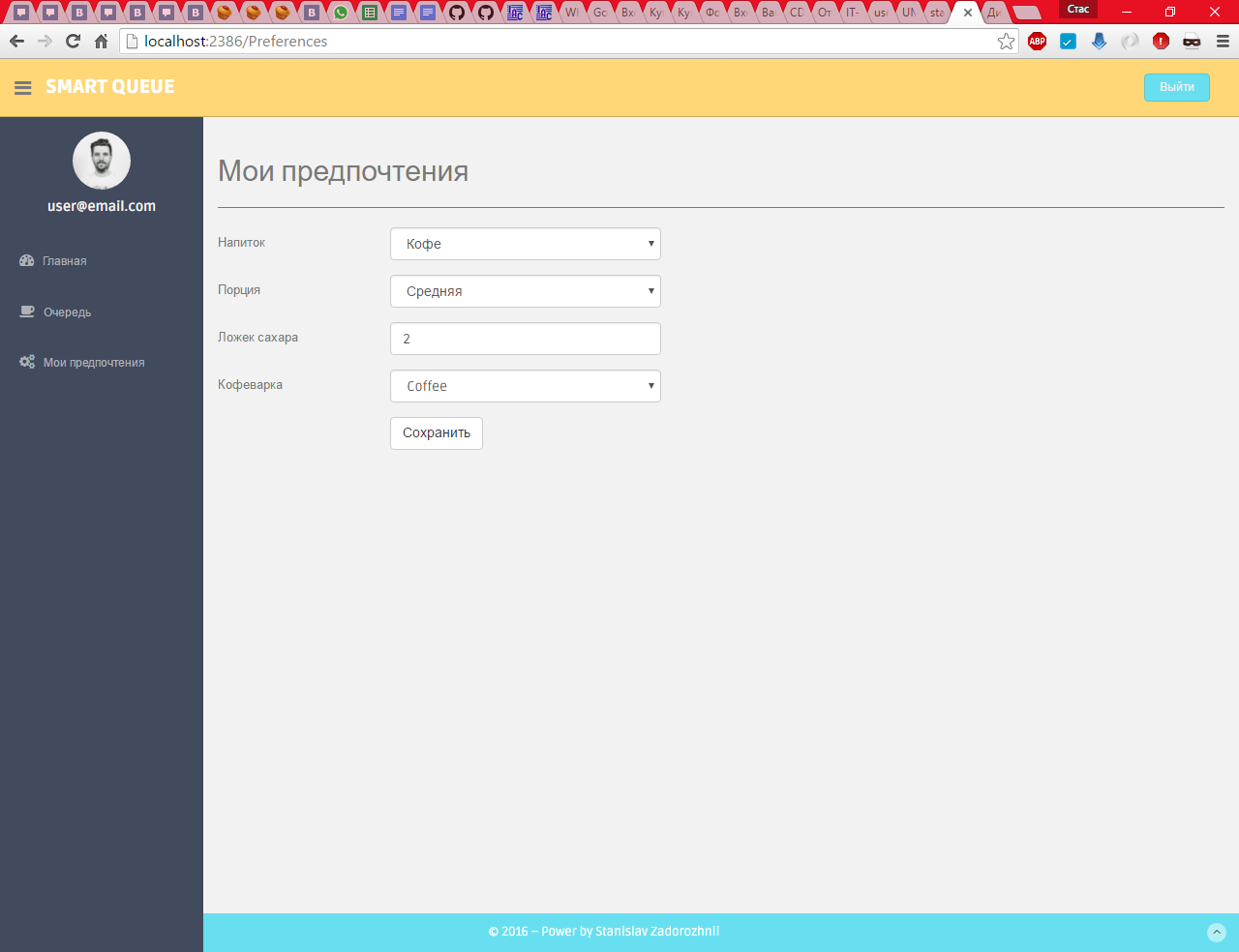


Рисунок 5.2 – Додавання вподобань користувача

На сторінці з редагуванням власного профіля (див. рис. 5.3), користувач може змінити данні свого профіля та завантажити свою фотографію. Також за допомогою цієї сторінки можна змінити пароль від своєї сторінки

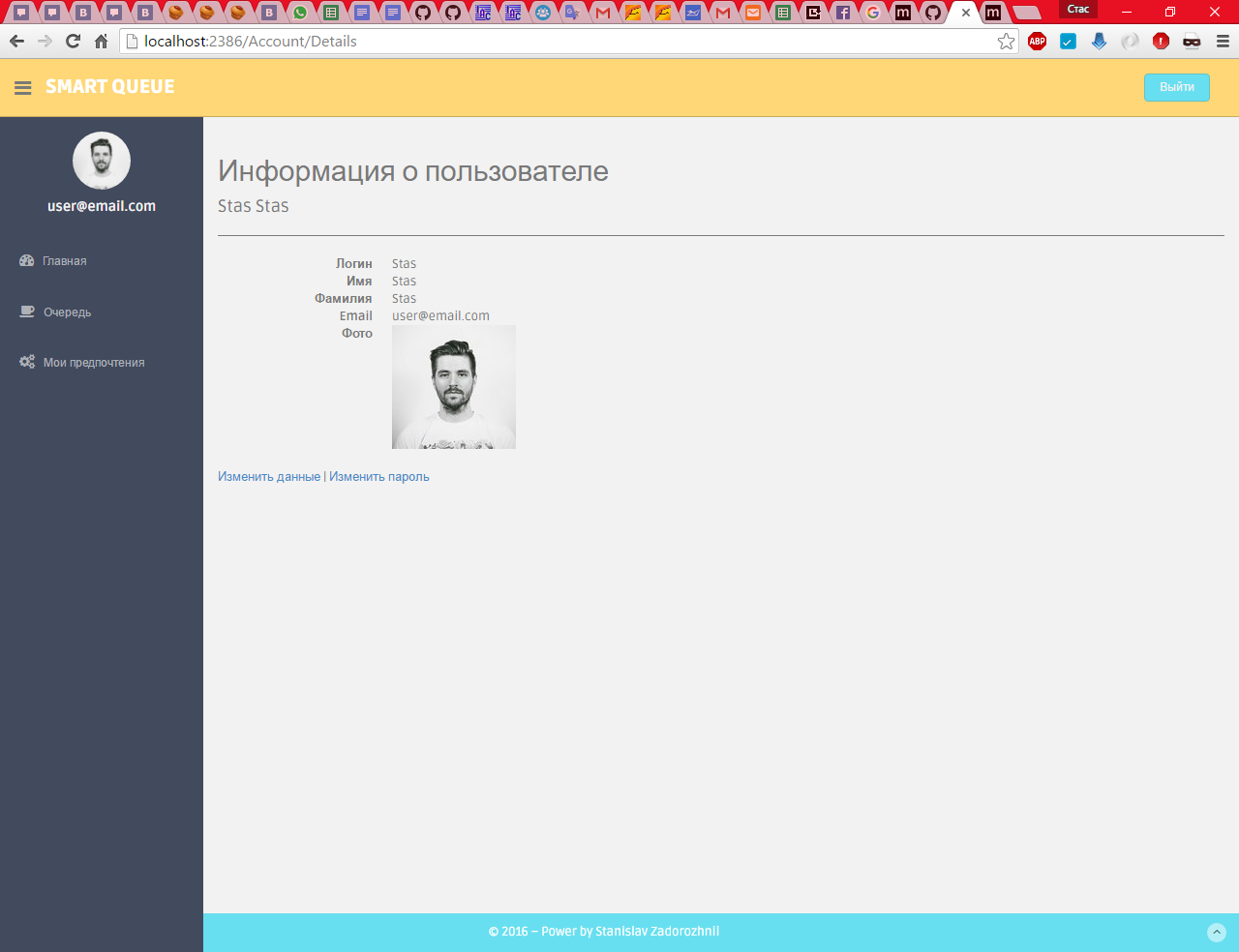


Рисунок 5.3 – Сторінка з редагуванням користувача

На сторінці для запису у чергу (див. рис. 5.4) співробітник може створити замовлення для приготування кави. Для цього необхідно заповнити необхідні поля, але завдяки тому, що є сторінка з вподобаннями, то ці поля заповнюються автоматично. Але при необхідності їх можна змінити. Після створення замовлення користувач потрапляю на сторінку з очікуванням, де йому відображається час, до завершення приготування (див. рис. 5.5).

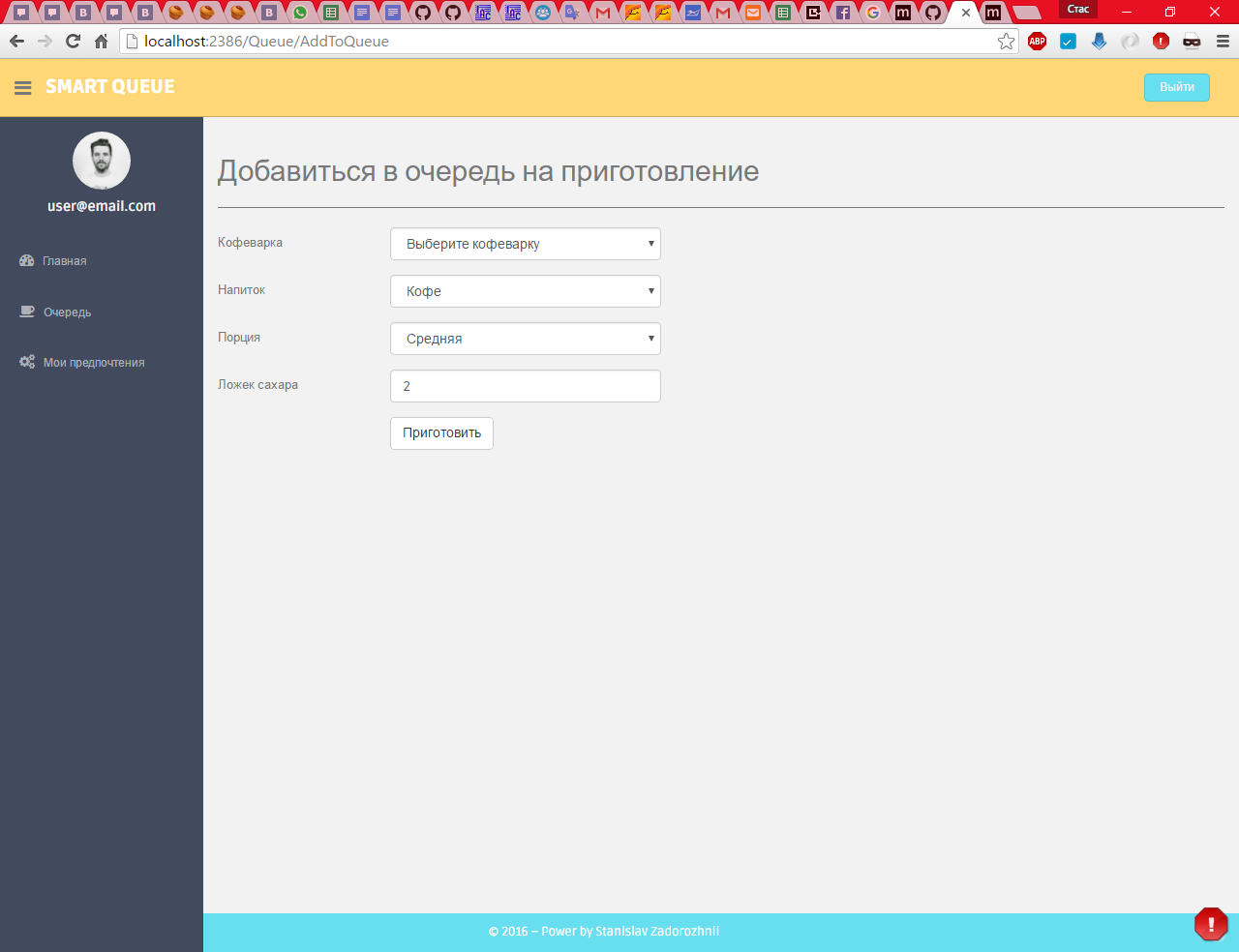


Рисунок 5.4 – Створення нового замовлення

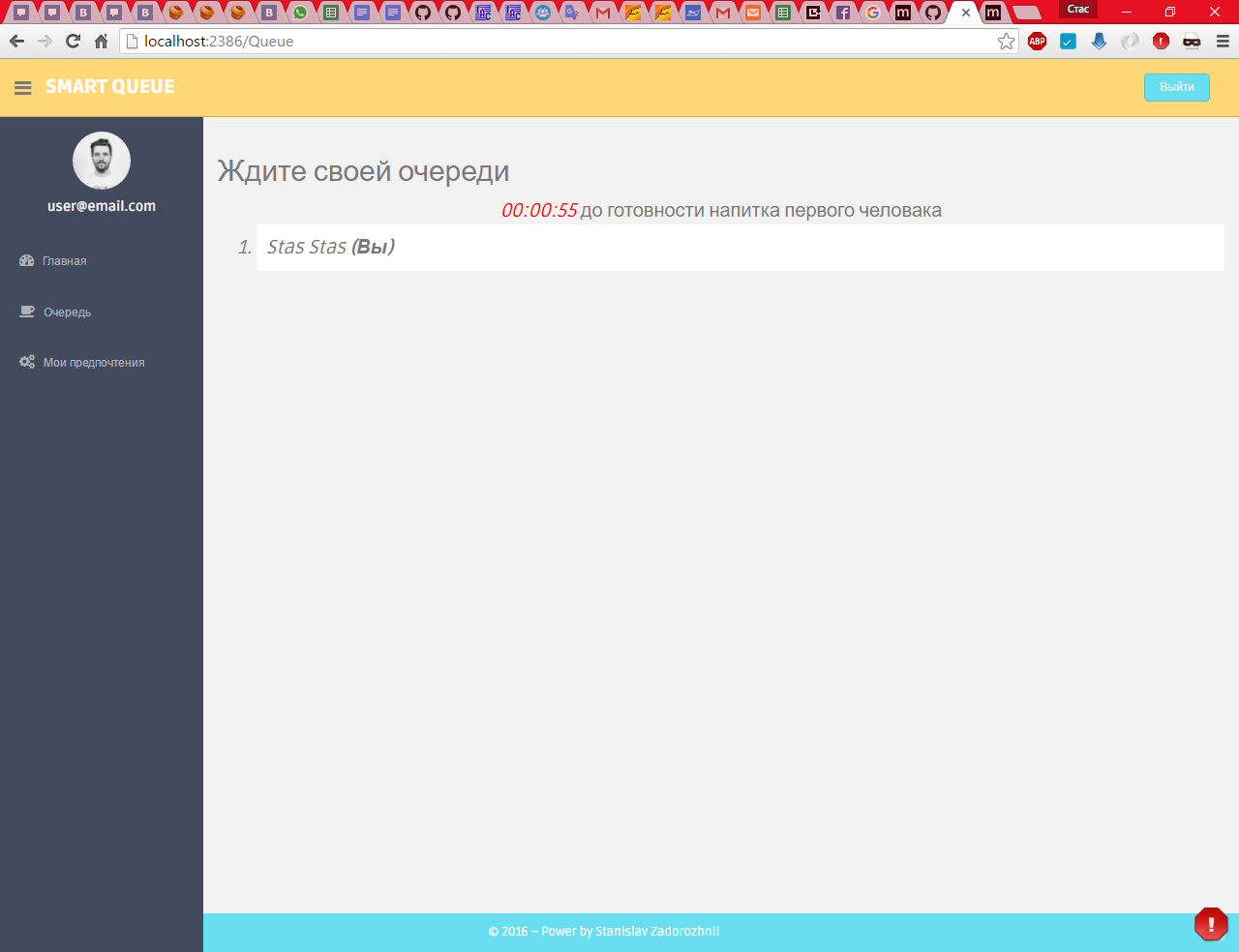


Рисунок 5.5 – Очікування замовлення

Після закінчення часу для приготування напою користувач отримує про це відповідне повідомлення (див. рис. 5.6).

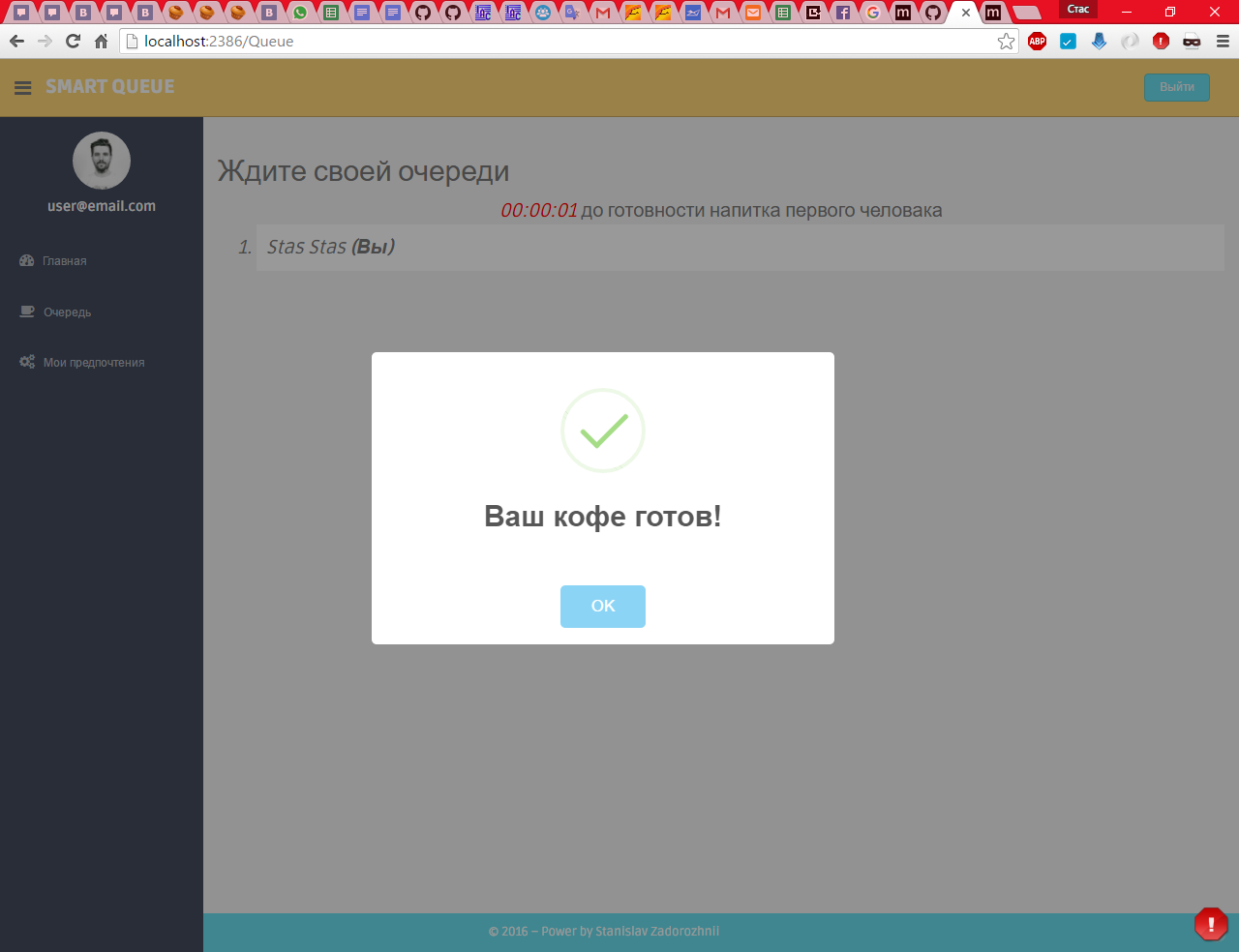


Рисунок 5.6 – Повідомлення про закінчення черги

## 5.3 Додаток для Android

Мобільний клієнт має досить простий інтерфейс, що повністю відповідає усім функціям, що він повинен виконувати. Після відкриття додатку користувач має авторизуватися через свій обліковий запис як на основному сайті. Після авторизації він потрапляє на вікно, у якому є список всіх доступних кавоварок. На цьому вікні він має можливість додати свій запит до черг і перейти потім на сторінку, де він зможе переглянути поточний статус приготування його напою.

Також у меню доступний пункт за допомогою якого користувач може налаштовувати свої вподобання для подальшої швидкої роботи з сайтом.

# ВИСНОВКИ

У результаті виконання курсової роботи був розроблений програмний продукт, що дозволяє керувати приготуванням напоїв. Система базується на сервісно-орієнтованій архітектурі, тому складається з окремих частин: сервера, веб-клієнта та мобільного додатку.

Клієнт для платформи Android було розроблено з використанням мови програмування Java. Сервер було розроблено у середовищі Visual Studio за допомогою мови програмування C#, мобільний додаток у Android Studio. Сервер розроблявся з використанням фреймворка ASP.NET MVC та був розгорнутий у хмарному середовищі Azure. База даних розміщується так само у цьому хмарному середовищі. Для зручного оновлення сайту було налаштовану автоматичну розгортку сайту після зміни даних у головному репозеторії (Continuous delivery).

В результаті розробки поставлену задачу було цілком виконано. Програмний продукт має зрозумілий інтерфейс для користувачів, не викликає труднощів з відправкою даних на сервер.

Як вже зауважувалося вище, програмний продукт навіть після реалізації усіх поставлених на курсову роботу задач, для комерційного запуску має дороблюватися. Перш за все, необхідно доробити додатковий функціонал та створити комерційну складову проекту. Другорядним завданням стоїть реалізація клієнтів для різних мобільних платформ (iOS, OS Android, Windows Phone), це дозволить покрити більшість сучасних смартфонів та планшетів. Також, для зручного користування у різних країнах необхідно зробити локалізацію додатку.

# ПЕРЕЛІК ПОСИЛАНЬ

1. Библиотека MSDN. Разработка на .NET / MSDN: сеть разработчиков Microsoft. [Электронный ресурс]: – Режим доступа: http://msdn.microsoft.com/ru-ru/library/aa139615.aspx
2. Бондарев, В.М. Объектно-ориентированное программирование на С# [Текст] : учеб. пособ. /В.М. Бондарев. – Х. : Компания СМИТ, 2009. – 224 с.
3. Троелсен, Э. Язык программирования C# 5.0 и платформа .NET 4.5, 6-е изд. [Текст]: пер. с англ.- М.: Вильямс, 2013 – 1312 с.
4. Уотсон, К., Нейгел, К., Педерсен, Я. Visual C# 2010: полный курс. [Текст] : Пер. с англ. – М.:ООО «И.Д. ильямс», 2011 – 960 с.
5. Шилдт Г. C# 4.0.: Полное руководство [Текст]: пер. с англ.- М.: Вильямс, 2011 – 1056 с.