

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Кваліфікаційна робота бакалавра

на тему: Дистрибутивний клієнт-серверний застосунок для
віддаленого проведення локальних голосувань та
референдумів

Виконав студент групи К19і
спеціальності 122 Комп'ютерні науки
Котобан Андрій Андрійович

Керівник Штефан Наталія Зінов'ївна

Консультант д. т. н., доцент
Великодний Станіслав Сергійович

Рецензент к. ф.-м. н., доцент
Ткач Тетяна Борисівна

Одеса 2021

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА ТЕРМІНІВ.....	5
ВСТУП.....	6
1 АНАЛІТИЧНА ЧАСТИНА.....	7
1.1 Опис предметної області.....	7
1.2 Характеристика об'єкту розробки	7
1.3 Огляд програмних аналогів	8
2 ВИБІР ПРОГРАМНИХ ЗАСОБІВ ДЛЯ РОЗРОБКИ ПРОГРАМНОГО ПРОДУКТУ.....	15
2.1 Мова програмування C#	15
2.2 Windows Presentation Foundation.....	16
2.3 Newtonsoft.Json	18
2.4 Xamarin і крос-платформна розробка	20
2.5 Розширювана мова розмітки XML	23
3 РЕАЛІЗАЦІЯ ПРОЕКТУ	26
3.1 Проектування інтерфейсу проекту	26
3.2 Діаграми Use-Case	28
3.3 UML-діаграма активності	31
3.4 Опис роботи з програмою-сервером.....	32
3.5 Опис роботи з програмою-клієнтом	39
3.6 TCP сервер та його імплементація в проекті	42
3.7 TCP клієнт та його імплементація в проекті.....	45
ВИСНОВКИ.....	48
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	49

ПЕРЕЛІК СКОРОЧЕНЬ ТА ТЕРМІНІВ

LINQ – Language Integrated Query

WPF – Windows Presentation Foundation

XML – Extensible Markup Language

Рендеринг – візуалізація у комп’ютерній графіці

C++ – мова програмування

C# – мова програмування

Xceed.WPF – бібліотека з елементів керування, компонентів та утиліт WPF для створення додатків Windows

Newtonsoft.Json – бібліотека

Android – операційна система

Xamarin – платформа з відкритим кодом для створення додатків для iOS, Android та Windows за допомогою .NET.

XAML – мова декларативною розмітки інтерфейсу

XSL – стильові таблиці

ВСТУП

Не завжди виправдані значні витрати на проведення досліджень за допомогою маркетингових агентств. Для проведення глибоко аналізу ринку або порівнянних за масштабом завдань пряма дорога до маркетологам, якщо, звичайно, дозволяє бюджет. Але в дослідженнях менш глобальних допоможуть онлайн опитування – існують сервіси, що дозволяють провести онлайн анкетування максимально просто: достатньо створити опитування, поширити його і отримати вже оброблені результати [1].

Сервіси для створення онлайн опитувань необхідні в різних ситуаціях, коли потрібно збір думок, статистика, перевірка навичок. Найчастіше такі сервіси доступні у соціальних мережах чи у вигляді окремого мобільного додатку, для використання якого користувачам необхідно його завантажити на платній основі.

Сервіси для опитувань широко використовуються маркетологами, вченими-дослідниками, студентами, психологами та соціологами: для вивчення ринку, реакції на бренд або продукт, перевірки знань, виявлення певних закономірностей, психодіагностики, а також в розважальних цілях, для жартівливих тестів і анкет.

Подібні сервіси дозволяють самостійно створити онлайн-голосування або провести анкетування, скориставшись програмними можливостями спеціального движка.

Мета дипломної роботи – розробка дистрибутивного клієнт-серверного застосунку для голосувань та референдумів.

Загальні характеристики кваліфікаційної роботи:

- повний обсяг сторінок пояснювальної записки – 48;
- кількість рисунків – 22;
- кількість посилань – 18.

1 АНАЛІТИЧНА ЧАСТИНА

1.1 Опис предметної області

Голосування та опитування – формат спілкування з аудиторією, в якому немає правильного чи неправильного відповіді. Вони ідеально підходять для ненав'язливого залучення аудиторії, збору думок, а також отримання цінних інсайтів.

Коли необхідно терміново провести невелике маркетингове дослідження аудиторії, вивчити її потреби або отримати актуальне для бізнесу думку клієнтів, на допомогу приходять сервіси онлайн-опитувань. Таких інструментів досить багато, вони відрізняються за своїм призначенням і функціоналу.

Онлайн-опитування – це хороша альтернатива маркетингових досліджень. Звичайно, для проведення глибокого аналізу ніші вам необхідна допомога маркетолога. А для менш глобальних досліджень можна використовувати різні програми для анкетування.

Такі сервіси застосовуються для невеликих опитувань або вікторин. Мета у кожного своя: вивчити нову аудиторію, залучити клієнтів, дізнатися думку про продукт.

1.2 Характеристика об'єкту розробки

Онлайн-голосування – це чудовий інструмент для оперативного отримання відгуків, який дозволяє зв'язатися з аудиторією або клієнтами в будь-який час. Такий інструмент буде в нагоді як на підприємствах, дитячих фестивалях та подіях, так і наприклад у навчальних закладах для організації студентського розвитку.

Метою дипломної роботи є розробка дистрибутивного клієнт-серверного застосунку для віддаленого проведення голосувань та референдумів.

Планується даний проект реалізувати у вигляді двох додатків для більш зручного використання:

1. Десктоп-додаток для адміністрування голосуванням, що дозволяє в режимі реального часу відкривати сесії голосувань і приймати результати, генерувати звіти і виносити вердикти рішень.

Всі голосування можуть проводитися публічно або відкрито – досить підключитися до будь-якої мережі, якщо локальна мережа, то голосувати зможуть тільки підключення користувачі, якщо глобальна мережа – то може підключитися будь-який знаючи IP-адреса і номер вільного порту.

Також планується функціональність номінацій для окремих голосувань і функція збереження шаблонів заходів для підготовки до них або для збереження в разі використання його заново.

2. Мобільний додаток, в який надходить повідомлення про голосування і його закінчення, в ньому можна вибрати бал, який налаштовується на сервері і бачити свої оцінки.

1.3 Огляд програмних аналогів

Щоб реалізувати проект на першому етапі розробки слід оглянути декілька аналогів для виявлення головних від'ємностей, які є суттєвими на даний час для користувачів.

«VoteMeApp» дозволяє організаціям, групам і співтовариствам приймати рішення в режимі онлайн за допомогою технології електронного голосування. Безліч видів виборів – один інструмент. Організатори виборів можуть вибрати тип виборів, найбільш підходящий для їх спільноти. У нас є три типи виборів: голосування з одним вибором, голосування з декількома варіантами відповіді і рейтингове голосування.

Голосування з одним вибором – цей тип виборів найкраще підходить для швидких рішень (Так/Ні/Утриматися) і коли ваші виборці однаково інформовані про рішення, які їм необхідно прийняти. Голосування з декількома

варіантами відповіді – багаторазове голосування краще підходить для складних питань, коли виборцям потрібно мати більше можливостей. Такий тип голосування рекомендовано, коли виборці можуть вибрати кілька варіантів і ці варіанти не виключають один одного.

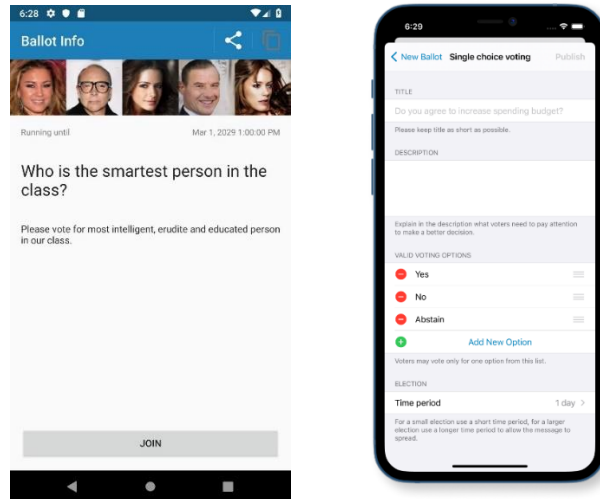


Рисунок 1 – Приклад роботи мобільного додатку «VoteMeApp»

Рейтингове голосування – цей тип виборів найкраще підходить для обрання лідерів, які користуються найбільшою довірою. Кандидатів немає як на інших виборах.

Усі виборці можуть голосувати один за одного. У підсумку ми обчислюємо граф голосування, і переможцем стає людина, якій довіряють найбільше.

VoteMeApp використовується:

- професійними асоціаціями;
- суспільно-політичними організаціями;
- спільнотами людей;
- організаторами заходів і конференцій;
- корпораціями agile;
- студентськими групами [2].

Платформа (додаток) «Event» (рис. 2) для конференції з відображенням інтерактивного контенту (голосування, "задати питання" та інше) в режимі online на екрані в залі. Без скачування і установки додаткових програм, для мобільних пристроїв і стаціонарних комп'ютерів.

Платформа являє собою додаток для заходів, доступ до якого здійснюється через веб-інтерфейс (браузер) з прямим доступом тільки до потрібного функціоналу. Наприклад, користувач може бути відразу направлений в інтерактивне опитування (онлайн голосування), без додаткових вкладок і меню. Або в повне меню конференції з доступом до всіх можливостей програми.

Призначено для мобільних і стаціонарних пристроїв (смартфона, планшета, ноутбука), з виведенням даних в режимі онлайн (голосування, питань для спікера) на екран в залі [3].

Переваги «Event»:

- є по QR-коду і через веб-браузер (по короткій посиланням без завантаження та встановлення програми);
- призначений для всіх смартфонів і планшетів (незалежно від операційної системи), а також є для комп'ютерів;
- зручний, інтуїтивно зрозумілий інтерфейс та широкі можливості по впровадженню необхідного функціоналу під конкретний захід.

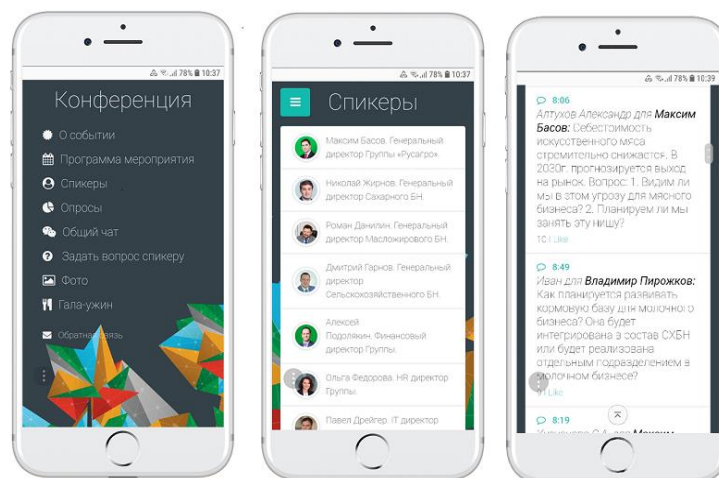


Рисунок 2 – Приклад роботи платформи «Event»

У меню програми доступні такі функції, як програма (розклад) заходи з таймингом, інформація про спікерів та доповідях, різні інформаційні схеми і карти заходи, фотографії по даній події.

Можливості програми:

- інформація про подію і програма заходу;
- інформація про спікерів (фото і необхідна додаткова інформація);
- голосування і інтерактивні опитування;
- задавати питання спікерам (з можливістю вибрати потрібного спікера);
- вибірка питань за певним спікеру;
- оцінювати поставлені запитання;
- загальний чат (залишати коментарі в процесі заходу);
- завантаження фото за поточним заходу;
- можливість для користувачів завантажити фотографії з заходу;
- online-трансляція;
- впровадження інших корисних функцій для вашого заходу [3].

«SurveyMonkey» (рис. 3) – один з найпопулярніших онлайн-сервісів для проведення опитувань. З його допомогою можна отримувати інформацію про цільову аудиторію, визначати ступінь задоволеності клієнтів товарами і послугами.

Основні функції:

- створення і розсилка опитувальних форм;
- підключення інструментів аналізу отриманих даних;
- можливість побудови тестів і контрольних робіт, налаштування автооцінювання за балами;
- наявність налаштовується брендингу компанії – колірне рішення, додавання логотипу і веб-адреси сайту;
- формування звітів [4].

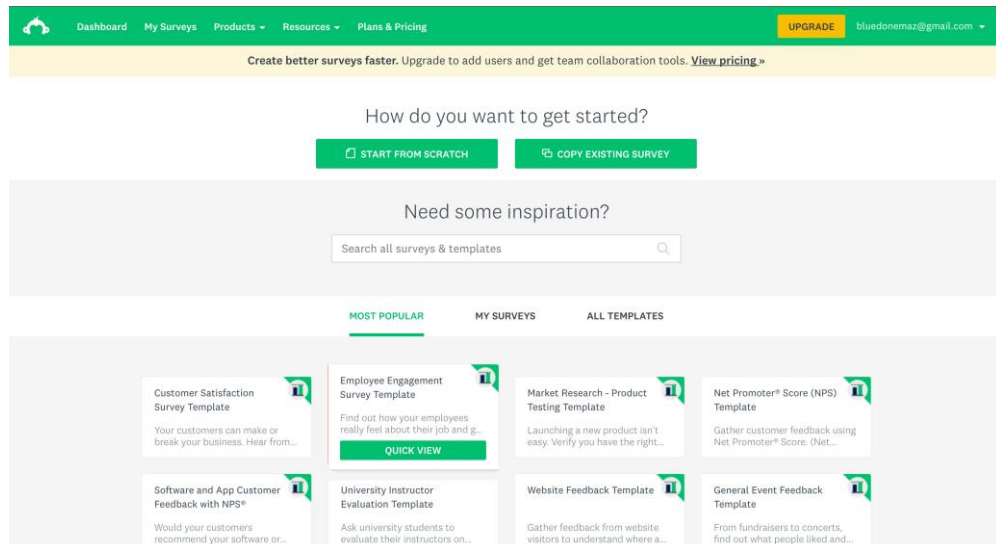


Рисунок 3 – Скріншот роботи «SurveyMonkey»

Переваги «SurveyMonkey»:

- існує безкоштовний тариф, що дозволяє опитати до 100 відвідувачів сайту;
- платні версії мають розширений функціонал для професійного використання;
- доступні різні види питань і сценарії проведення анкетування;
- форму опитування можна вбудувати на сайт, опублікувати на сторінці в соцмережі, відправити електронною поштою;
- анкету можна створити самостійно з нуля або з використанням шаблону;
- підсумкові дані конвертуються в різні формати для подальшої обробки;
- результати можна фільтрувати і групувати. На їх основі будуються графіки, і створюються звіти;
- існує можливість роботи в розрахованому на багато користувачів режимі;
- платформа доступна в будь-якому браузері;

- існують додатки для смартфонів;
- доступний обмін даними зі сторонніми сервісами [4].

«Simpoll» (рис. 4) – платформа для створення опитувань будь-якої складності. За допомогою конструктора можна формувати анкети, тестування, голосування та інше.

Основні функції:

- створення опитувальників будь-якої складності і тематичного спрямування;
- можливість поширення форм опитувань – створену анкету можна вбудувати на сайт, відправити e-mail, через соціальні мережі Twitter і «Facebook»;
- проведення маркетингових досліджень, отримання інформації про клієнтів [5].

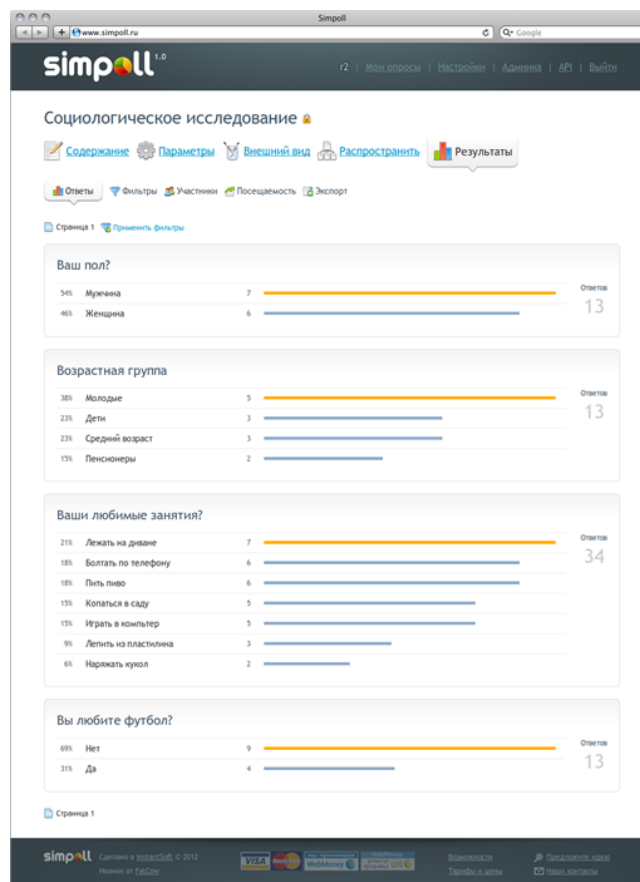


Рисунок 4 – Скріншот роботи «Simpoll»

Переваги:

- зрозумілий інтерфейс. Об'єкти створюються з візуальних блоків. Для кожного елемента можна вибрати колір і стиль оформлення;
- варіативність способів поширення анкет;
- експорт результатів. Підсумки опитування можна вивантажити в таблицю Excel для подальшої обробки;
- додавання питань в форму за допомогою технології Drag&Drop (просто перетягування);
- різноманітні фільтри для відображення результатів. Існує можливість подивитися відповіді респондентів різних категорій, наприклад тільки чоловіків або жінок, осіб певної вікової категорії, жителів того чи іншого регіону.

Недоліки:

- завищені тарифи (в контексті доступного функціоналу);
- слабка технічна підтримка.

Сервіс «Simpoll» підходить для особистого і корпоративного використання. Існують різні варіанти використання сервісу. Безкоштовний тариф має обмеження за кількістю створених анкет, питань в них і отриманих відповідей.

Також є кілька платних тарифів. Їх ціна залежить від доступних можливостей. Коробкова версія програми призначена для установки на власний сервіс. У ній відсутні будь-які обмеження [5].

2 ВИБІР ПРОГРАМНИХ ЗАСОБІВ РОЗРОБКИ ПРОГРАМНОГО ПРОДУКТУ

Для десктопного додатку будуть використані такі засоби, як C#, WPF, Xceed.WPF, Newtonsoft.Json, Extended.Wpf.Toolkit; для мобільного додатку: Android, C#, Xamarin, Xamarin.Form, XML, Mono.Android, Newtonsoft.Json.

2.1 Мова програмування C#

C# програмісти стають більш затребуваними завдяки розвивається екосистемі мови. На ньому пишуть бекенд, програми, ігри.

Так як C# – це проект Microsoft, то у нього все в порядку з підтримкою і з бібліотеками. У світі C# бібліотеки є практично для всього, в тому числі і для роботи з нейросетями і машинним навчанням – ML.NET.

Синтаксис мови усуває складності C++ і надає такі потужні можливості, як обнуляти значення типів, перерахування, делегати, лямбда-вирази і прямий доступ до пам'яті. C# підтримує універсальні методи і типи, які підвищують безпеку типів і продуктивність. Ітератори дозволяють класів творців колекцій визначати кастомні поведінки ітерацій, які легко використовувати в клієнтському коді. Вирази інтегрованої мови запитів (LINQ – Language Integrated Query) роблять строго типізований запит першокласної конструкцією мови.

C# – об'єктно-орієнтована мова, а значить, підтримує інкапсуляцію, успадкування і поліморфізм. Клас може успадковуватися безпосередньо від одного батьківського класу, але може реалізовувати будь-яке число інтерфейсів.

Методи, які скасовують віртуальні методи в батьківському класі, вимагають вказівки ключового слова `override` для запобігання випадкового перевизначення.

На додаток до всіх плюшками ООП мова C# полегшує розробку програмних компонентів за допомогою декількох інноваційних конструкцій мови:

- інкапсульовані сигнатури методів, названі делегатами, включають оповіщення про безпеку типів;
- властивості служать акцесорами до змінних закритих частин;
- атрибути надають декларативні метадані про типах під час виконання;
- рядкові документаційні коментарі xml;
- інтегрований мову запитів (linq) надає вбудовані можливості запитів між різними джерелами даних [6].

Це означає, що ви можете використовувати всі можливості нейронів в додатках і об'єднувати їх за допомогою одного і того ж мови програмування. А так як C# – мультиплатформенний мову, то машинне навчання можна вмонтувати практично будь-що завгодно, від мобільних додатків до переносної електроніки.

Щоб один і той же C# – код працював на різних платформах, використовують Mono і Xamarin – продукти для крос-платформної розробки під різні операційні платформи [7].

2.2 Windows Presentation Foundation

WPF (Windows Presentation Foundation) – аналог WinForms, система для побудови клієнтських додатків Windows з візуально привабливими можливостями взаємодії з користувачем, графічна (презентаційна) підсистема в складі .NET Framework (починаючи з версії 3.0), яка використовує мову XAML.

Якщо при створенні традиційних додатків на основі WinForms за отрисовку елементів управління і графіки відповідали такі частини ОС Windows, як User32 і GDI +, то додатки WPF засновані на DirectX. У цьому полягає ключова особливість рендеринга графіки в WPF.

Використовуючи WPF, значна частина роботи по відображенні графіки, як найпростіших кнопочок, так і складних 3D-моделей, лягати на графічний

процесор на відеокарті, що також дозволяє скористатися апаратним прискоренням графіки (рис. 5).

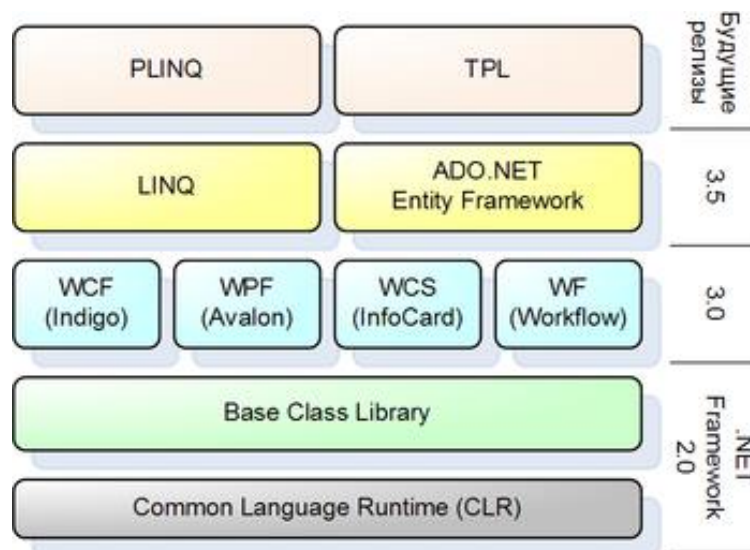


Рисунок 5 – WPF у складі .NET Framework

Однією з важливих особливостей є використання мови декларативної розмітки інтерфейсу XAML, заснованого на XML: можна створювати насичений графічний інтерфейс, використовуючи або декларативне оголошення інтерфейсу, або код на керованих мовах C# і VB.NET, або поєднувати і те, і інше.

Переваги WPF:

1. Використання традиційних мов .NET-платформи – C# і VB.NET для створення логіки додатка.
2. Можливість декларативного визначення графічного інтерфейсу за допомогою спеціальної мови розмітки XAML, заснованому на XML і представляє альтернативу програмному створенню графіки та елементів управління, а також можливість комбінувати XAML і C#/VB.NET.
3. Незалежність від дозволу екрану: оскільки в WPF всі елементи вимірюються в незалежних від пристрою одиницях, додатки на WPF легко масштабуються під різні екрани з різним дозволом.

4. Нові можливості, яких складно було досягти в WinForms, наприклад, створення тривимірних моделей, прив'язка даних, використання таких елементів, як стилі, шаблони, теми.
5. Хорошу взаємодію з WinForms, завдяки чому, наприклад, в додатках WPF можна використовувати традиційні елементи управління з WinForms.

Багаті можливості по створенню різних додатків: це і мультимедіа, і двовимірна і тривимірна графіка, і багатий набір вбудованих елементів управління, а також можливість самим створювати нові елементи, створення анімацій, прив'язка даних, стилі, шаблони, теми і багато іншого.

Апаратне прискорення графіки – незалежно від того, чи працюєте ви з 2D або 3D, графікою або текстом, все компоненти програми транслюються в об'єкти, зрозумілі Direct3D, і потім візуалізуються за допомогою процесора на відеокарті, що підвищує продуктивність, робить графіком більш плавною.

Створення додатків під безліч ОС сімейства Windows – від Windows XP до Windows 10 [8].

2.3 Newtonsoft.Json

JavaScript Object Notation (JSON) – текстовий формат відображення даних, заснований на упорядкованому списку і парах "ключ-значення". Ви можете використовувати їх для зберігання даних в логічному і впорядкованому вигляді, щоб отримати до них безперешкодний доступ. Також його використовують для обміну даними між веб-серверами і клієнтами.

Крім обміну даними, його можна використовувати для міграції баз даних, наприклад, з JSON на SQL. Ви навіть можете експортувати дані з допомогою JSON з уже зареєстрованих веб-додатків.

Незважаючи на те, що JSON був створений на основі JavaScript, він є незалежним від мови, що означає, що різні мови програмування можуть виконувати розбір даних.

Вперше Json.NET був випущений у червні 2006 р. З тих пір Json.NET завантажували сотні тисяч разів розробниками з усього світу. Він використовується в багатьох великих проектах з відкритим кодом, зокрема:

- Mono, реалізація платформи .NET із відкритим кодом;
- RavenDB, база даних документів на основі JSON;
- ASP.NET SignalR, асинхронна бібліотека для побудови багатокористувацьких інтерактивних веб-додатків у режимі реального часу;
- ASP.NET Core, веб-програма та сервісна база Microsoft. Серіалізація – це процес перекладу структури даних в послідовність бітів, або ж в іншу структуру даних, яку зручно зберігати, передавати (цей процес в статті докладно ми не будемо розглядати).

Десеріалізацію – зворотний процес. Процес перетворення серіалізованих даних в структуру даних. Серіалізація JSON – важливий фактор в веб-додатках. На неї покладається велика частина сьогоденного REST API.

Основна функціональність по роботі з JSON зосереджена в просторі імен System.Text.Json. Ключовим типом є клас JsonSerializer, який і дозволяє серіалізувати об'єкт в json і, навпаки, десеріалізувати код json в об'єкт C#.

Коли javascript-клієнт відправляє JSON в тілі POST запиту, сервер використовує десеріалізацію JSON для перетворення його в C# об'єкт. І коли сервер повертає у відповідь об'єкт, він серіалізує цей об'єкт в JSON, щоб javascript клієнт міг його зрозуміти. Це великі операції, які виконуються для кожного запиту з об'єктами.

Newtonsoft.Json (також відомий як Json.NET) – серіалізатор, в даний час є стандартом в галузі. Був інтегрований в ASP.NET, хоча і був стороннім [9].

Переваги використовувати бібліотеку JSON.NET:

- Json.NET робить просте легким, а складне можливим;
- наявність об'єкти Json.NET JObject, JArray і JValue;
- Json.NET підтримує Windows, Windows Store, Windows Phone, Mono і Xamarin;

- Json.NET – це програмне забезпечення з відкритим вихідним кодом і повністю безкоштовне для комерційного використання [10].

2.4 Xamarin і крос-платформна розробка

Xamarin – це фреймворк для кроссплатформенної розробки мобільних додатків (iOS, Android, Windows Phone) з використанням мови C# (рис. 6).



Рисунок 6 – Можливості Xamarin

Ідея дуже проста. Код пишеться на улюбленому мові розробника, із застосуванням всіх звичних для нього мовних фіч нібито LINQ, лямбда-виразів, Generic`ов і async`ов: при цьому отримується повний доступ до всіх можливостей SDK платформи і рідного механізму створення UI, отримуючи на виході додаток, яке, строго кажучи, нічим не відрізняється від нативних і (принаймні по завіреннях) не поступається їм у продуктивності [11].

Xamarin.Forms представляє платформу, яка націлена на створення кроссплатформенних додатків під Android, iOS і Windows 10. Є певні статистичні дані, що значна частина мобільних додатків створюється більш ніж для однієї платформи, наприклад, для Android і iOS. Однак неминуче розробники стикаються з наступними труднощами:

1. Відмінність в підходах побудова графічного інтерфейсу так чи інакше впливає на розробку. Розробники змушені підлаштовувати додаток під вимоги до інтерфейсу на конкретній платформі.
2. Різні API – розбіжність у програмних інтерфейсів і реалізації тих чи інших функціональностей також вимагає від програміста облік цих специфічних особливостей.
3. Різні платформи для розробки. Наприклад, щоб створювати додатки для IOS нам необхідна відповідне середовище – MAC OS X і ряд спеціальних інструментів, типу XCODE.

А в якості мови програмування вибирається Objective-C або Swift. Для Android можна використовувати найрізноманітніший набір середовищ. Але тут для переважної більшості додатків застосовується Java або Kotlin.

А для створення додатків під Windows використовується Visual Studio, а в якості мов – C#, F#, VB.NET, C++. Такий діапазон платформ, засобів розробки і мов програмування не може позитивно позначатися на терміни створення додатків, і, в кінцевому рахунку, на кошти, що виділяються на розробку.

Було б дуже ефективно мати один інструмент, який дозволяв легко і просто створювати додатки відразу для всіх платформ. Xamarin дозволяє створювати одну єдинственну логіку додатка із застосуванням C# і .NET відразу для всіх трьох платформ – Android, iOS, UWP [12]. Роботу Xamarin графічно можна представити таким чином (рис. 7).

Xamarin працює поверх фреймворка Mono, який надає opensource-реалізацію .NET Framework. Mono може працювати поверх різних платформ – Linux, MacOS.

На рівні кожної окремої платформи Xamarin покладається на ряд субплатформ. Зокрема:

- Xamarin.Android – бібліотеки для створення додатків на ОС Android;
- Xamarin.iOS – бібліотеки для створення додатків для iOS [9].

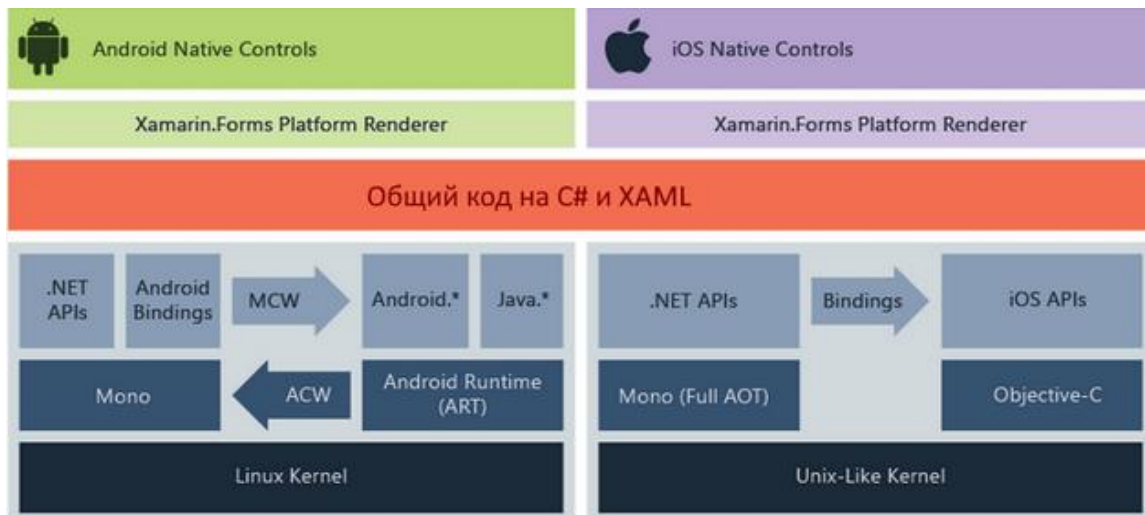


Рисунок 7 – Графічне представлення Xamarin

Ці субплатформи відіграють велику роль – через них додатки можуть направляються запити до прикладних інтерфейсів на пристроях під управлінням ОС Android або iOS.

За допомогою Xamarin.Android код C# з використанням Xamarin компілюється в Intermediate Language (IL), який потім при запуску програми компілюється в нативну збірку. Xamarin-додатки запускаються в середовищі виконання Mono. Безпосередньо код не може звертатися до API Android.

Спеціальна прошарок Managed Callable Wrappers (MCW) дозволяє транслювати виклику managed-коду в нативні виклики і звертатися до функціональності просторів імен Android. * I Java. *.

І навпаки, коли Android Runtime (ART) звертається до додатка з кодом Xamarin, то всі виклики проходять через обгортку Android Callable Wrappers (ACW) [12].

Додатки Xamarin.iOS на відміну від Xamarin.Android, який використовує JIT-компіляцію, застосовують AOT-компіляцію (Ahead-of-Time) коду C# в нативний ARM-код. Xamarin використовує проміжний шар Selectors (селектори) для трансляції викликів коду Objective-C в код на C# і шар Registrars (реєстратори) для трансляції коду C# в Objective-C.

В результаті шари Selectors і Registrars в цілому представляють премежуточний шар, який на ілюстрації вище позначений як "bindings" і який власне дозволяє взаємодіяти коду Objective-C з кодом C#.

Завдяки цим платформам можна створювати окремо додатки для Android, окремо для iOS, але найбільш важливою особливістю Xamarin є можливість створювати кросплатформені додатки – тобто одна логіка для всіх платформ. Дана можливість представлена технологією Xamarin.Forms і яка працює як би рівнем вище Xamarin.Android і Xamarin.iOS.

Тобто за допомогою Xamarin.Forms один раз можна визначити візуальний інтерфейс, один раз до нього прив'язати якусь логіку на C#, і все це буде працювати на Android, iOS і Windows. Потім Xamarin.Forms за допомогою рендерер (renderer) – спеціальних об'єктів для зв'язку контролери на XAML/C# з нативними контролери траншують візуальні компоненти Xamarin.Forms в графічний інтерфейс, специфічний для кожної платформи [13].

2.5 Розширювана мова розмітки XML

Сьогодні XML (англ. EXtensible Markup Language) може використовуватися в будь-яких додатках, яким потрібна структурована інформація – від складних геоінформаційних систем, з гігантськими обсягами інформації, що передається до звичайних "однокомп'ютерних" програм, що використовують цю мову для опису службової інформації.

При уважному погляді на навколишній нас інформаційний світ можна виділити безліч завдань, пов'язаних зі створенням і обробкою структурованої інформації, для вирішення яких може використовуватися XML.

В першу чергу, ця технологія може виявитися корисною для розробників складних інформаційних систем, з великою кількістю додатків, пов'язаних потоками інформації самої різної структурою [14].

В цьому випадку XML – документи виконують роль універсального формату для обміну інформацією між окремими компонентами великої програми.

XML є базовим стандартом для нової мови опису ресурсів, RDF, що дозволяє спростити багато проблем в Web, пов'язані з пошуком потрібної інформації, забезпеченням контролю за вмістом мережеских ресурсів, створення електронних бібліотек .

Мова XML дозволяє описувати дані довільного типу і використовується для подання спеціалізованої інформації, наприклад хімічних, математичних, фізичних формул, медичних рецептів, нотних записів.

Це означає, що XML може служити потужним доповненням до HTML для поширення в Web "нестандартної" інформації. Можливо, в найближчому майбутньому XML повністю замінить собою HTML, по крайній мере, перші спроби інтеграції цих двох мов уже робляться (специфікація XHTML) [14].

XML – документи можуть використовуватися в якості проміжного формату даних в триланкових системах. Зазвичай схема взаємодії між серверами додатків і баз даних залежить від конкретної СКБД і діалекту SQL, що використовується для доступу до даних.

Якщо ж результати запиту будуть представлені в деякому універсальному текстовому форматі, то ланка СКБД, як таке, стане "прозорим" для додатка. Крім того, сьогодні на розгляд W3C запропонована специфікація нової мови запитів до баз даних XQL, який в майбутньому може стати альтернативою SQL.

Інформація, що міститься в XML-документах, може змінюватися, передаватися на машину клієнта і оновлюватися по частинах. Розробляються специфікації XLink і Xpointer посилатися на окремі елементи документа, з урахуванням їх вкладеності і значень атрибутів.

Використання стильових таблиць (XSL) дозволяє забезпечити незалежне від конкретного пристрою виведення відображення XML-документів.

XML може використовуватися в звичайних додатках для зберігання і обробки структурованих даних в єдиному форматі [15].

XML-документ являє собою звичайний текстовий файл, в якому за допомогою спеціальних маркерів створюються елементи даних, послідовність і вкладеність яких визначає структуру документа і його зміст.

Основною перевагою XML документів є те, що при відносно простому способі створення та обробки (звичайний текст може редагуватися будь-яким тестовим процесором і оброблятися стандартними XML аналізаторами), вони дозволяють створювати структуровану інформацію, яку добре "розуміють" комп'ютери [15].

3 РЕАЛІЗАЦІЯ ПРОЕКТУ

Проект розроблено у вигляді двох додатків. Десктоп-додаток для адміністрування голосуванням, дозволяє в режимі реального часу відкривати сесії голосувань і приймати результати, генерувати звіти і виносити вердикти рішень.

Всі голосування можуть проводитися публічно або відкрито – досить підключитися до будь-якої мережі, якщо локальна мережа, то голосувати зможуть тільки підключення користувачі, якщо глобальна мережа – то може підключитися будь-який – знаючи IP-адреса і номер вільного порту.

Є функціональність номінацій для окремих голосувань і функція збереження шаблонів заходів для підготовки до них або для збереження в разі використання його заново. І мобільний додаток, в який надходить повідомлення про голосування і його закінчення, в ньому можна вибрати бал, який налаштовується на сервері і бачити свої оцінки.

3.1 Проектування інтерфейсу проекту

Для проектування інтерфейсу для програмного додатку «Телефонний довідник» було обрано «WireframeSketcher Studio» – у ньому бібліотека шаблонів навіть більше, ніж в Balsamiq, є макети під всі популярні мобільні ОС (iOS, Windows Phone і Android) і навіть деякі веб-фреймворки, наприклад «Bootstrap».

Інтерфейс цієї програми, в порівнянні з тим же «Balsamiq Mockups», більш зручний: найбільш часто використовувані функції, такі як угруповання елементів і винос їх на передній або задній план, виконується кнопками на головній панелі, а не з меню.

Проект може включати в себе кілька «екранів», а всередині кожного екрану елементи інтерфейсу можна об'єднувати в групи. При редагуванні наві-

гація по екранах і групам елементів здійснюється за допомогою панелі, постійно розташованої в лівій частині вікна програми, що вельми зручно при роботі над складними макетами. Праву сторону вікна займає браузер бібліотеки шаблонів. Макет для інтерфейсу десктопного додатку представлено на рисунках 8-9.

The wireframe shows a window with two tabs at the top: 'Создать' (Create) and 'Запустить' (Run). The 'Создать' tab is active. Below the tabs is a text input field labeled 'Название мероприятия' (Event Name) with a placeholder 'Text field'. Underneath is a large rectangular area labeled 'Команды' (Teams). To the right of this area are two buttons: 'Удалить' (Delete) and 'Добавить' (Add). Below the 'Команды' area are two side-by-side rectangular areas. The left one is labeled 'Судьи' (Judges) and the right one is labeled 'Номинации' (Nominations). Each of these areas has its own 'Удалить' and 'Добавить' buttons at the bottom.

Рисунок 8 – Макет закладки «Создать»

The wireframe shows a window with two tabs at the top: 'Запустить' (Run) and 'Создать' (Create). The 'Запустить' tab is active. The main area contains a text input field labeled 'IP-адреса' (IP address) with an 'OK' button to its right. Below this is another text input field labeled 'New Event Name'. Underneath are two buttons: 'Текущий' (Current) and 'Открыть' (Open). At the bottom is a large button labeled 'ЗАПУСТИТЬ' (RUN).

Рисунок 9 – Макет закладки «Запустить»

Всі елементи піддаються тонкій настройці. Проілюструвати механізм роботи інтерфейсу можна за допомогою функції клонування початкового екрана. Після цього кожен крок можна проілюструвати окремо, змінивши відповідні елементи.

Ще одна корисна функція – можливість імпортувати графічні файли. Завдяки цьому можна зробити скріншот реального інтерфейсу, а потім домалювати зміни. «WireframeSketcher Studio» доступна для Windows, Linux і OS X, а також у вигляді плагіна для Eclipse.

3.2 Діаграми Use-Case

Мета діаграми використання в UML – продемонструвати різні способи взаємодії користувача із системою. В Уніфікованій мові моделювання (UML) діаграма Use-Case може узагальнювати деталі користувачів системи (також відомих як актори) та їх взаємодії з системою. Основні елементи діаграм – учасник (актор) та прецедент (варіант).

Актор – це безліч логічно пов'язаних ролей, що виконуються при взаємодії з прецедентами або спільностями (система, підсистема або клас). Учасником може бути людина або інша система, підсистема або клас, які представляють неточність сутності.

Прецедент (випадок використання) – опис безлічі послідовних подій (включаючи варіанти), що виконуються системою, яка призводить до спостереження за результатами учасників. Прецедент представляє введення сутності, описової взаємодії між учасниками та системою.

Прецедент не показує, "як" досягається певний результат, а лише "що" саме виконується. Прецеденти обозначаються дуже простим чином – у вигляді еліпси, всередині якого вказано його назва [16].

Для його створення слід використовувати набір спеціалізованих символів та сполучників.

Дієва схема ефективного використання може допомогти при проектуванні обговорити та представити:

- сценарії, в яких ваша система або програма взаємодіє з людьми, організаціями або зовнішніми системами;
- цілі, яких ваша система або додаток допомагає досягти цим суб'єктам (відомим як актори).

Діаграма випадків використання не демонструє багато деталей – наприклад, вона не змодельює порядок виконання кроків. Натомість правильна діаграма відображає огляд високого рівня взаємозв'язку між випадками використання, акторами та системами.

Експерти рекомендують використовувати схеми випадків використання, щоб доповнити більш описовий варіант використання тексту [17].

Для дипломного проекту змодельюємо дві діаграми Use-Case: для адмін-додатку (рис. 10) та для клієнтського мобільного додатку (рис.11).

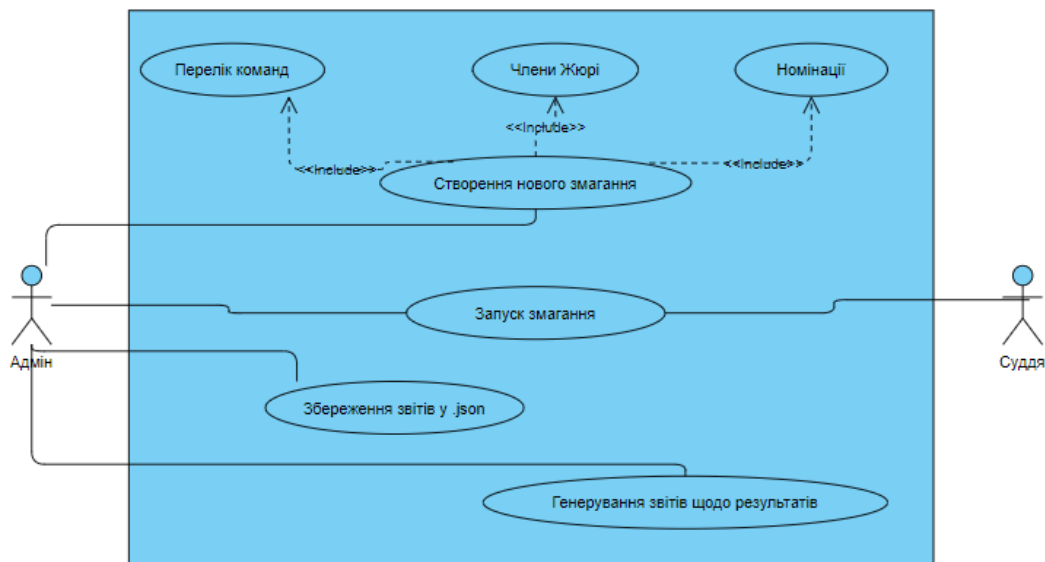


Рисунок 10 – Діаграма Use-Case для адмін-додатку

UML – це інструментарій для моделювання, який можна використовувати для побудови своїх діаграм.

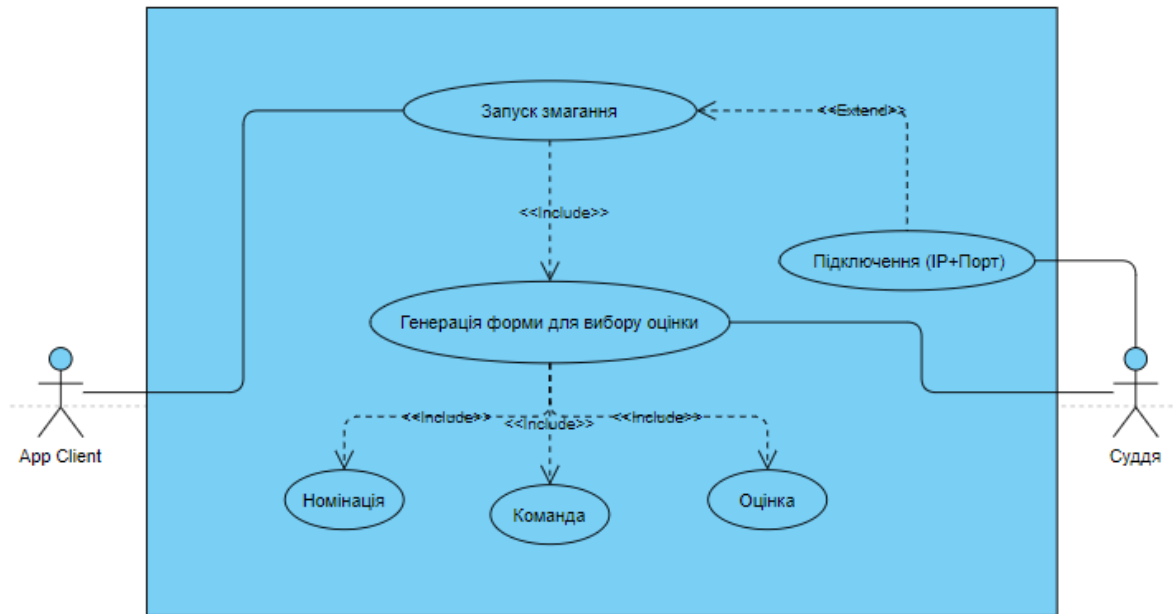


Рисунок 11 – Діаграма Use-Case для клієнтського додатку

Варіанти використання представлені позначеною овальною формою. Фігури представляють акторів у процесі, а участь актора в системі моделюється з допомогою межі між актором та випадком використання.

Діаграми випадків використання UML ідеально підходять для:

- представлення цілей взаємодії системи та користувача;
- визначення та організація функціональних вимог у системі;
- визначення контексту та вимог системи;
- моделювання основного потоку подій у випадку використання.

У разі підключення журі до додатку по локальній мережі їм необхідно заповнити форму для оцінок щодо виступу команд-учасників.

Якщо при голосуванні буде використане мобільний додаток, то для з'єднання з сервером необхідно буде ввести IP-адресу та номер вільного порту. Система перевіряє кількість з'єднаних членів журі відповідно до кількості введених даних у шаблоні події.

3.3 UML-діаграма активності

Уніфікована мова моделювання UML включає кілька підмножин діаграм:

- структурні діаграми;
- діаграми взаємодії;
- діаграми поведінки.

Діаграми діяльності, поряд із діаграмами випадків використання та стану машини, вважаються діаграмами поведінки, оскільки вони описують, що має відбутися в системі, що моделюється.

Зацікавлені сторони мають багато питань для управління, тому важливо спілкуватися з ясністю та стислістю. Діаграми діяльності допомагають людям у бізнесі та на сторонах розвитку організації об'єднатися, щоб зрозуміти той самий процес та поведінку.

Основне використання діаграм активності аналогічно іншим чотирьом діаграмам UML. Конкретне використання – моделювання потоку управління від одного дії до іншого. Цей потік управління не містить повідомлень.

Приклад діаграми представлено на рисунку 12 для серверної частини програмного додатку.

Діаграма діяльності підходить для моделювання потоків активності систем. Пропозиція може мати кілька систем. Діаграма діяльності також фіксує ці системи та описує перехід від однієї системи до іншої. Це конкретне використання недоступно для інших діаграм. Ці системи можуть бути базовими даними, зовнішніми черговими або будь-якою іншою системою.

У багатьох випадках діаграми активності нагадують блок-схеми, а принципова розробка між діаграмами діяльності та нотацією блок-схем складається в тому, що перші підтримують паралельні процеси.

Можна скористатися перевагами, наданими розгалуженням і об'єднанням, для опису паралельних алгоритмів одночасно виконуються програм [18].

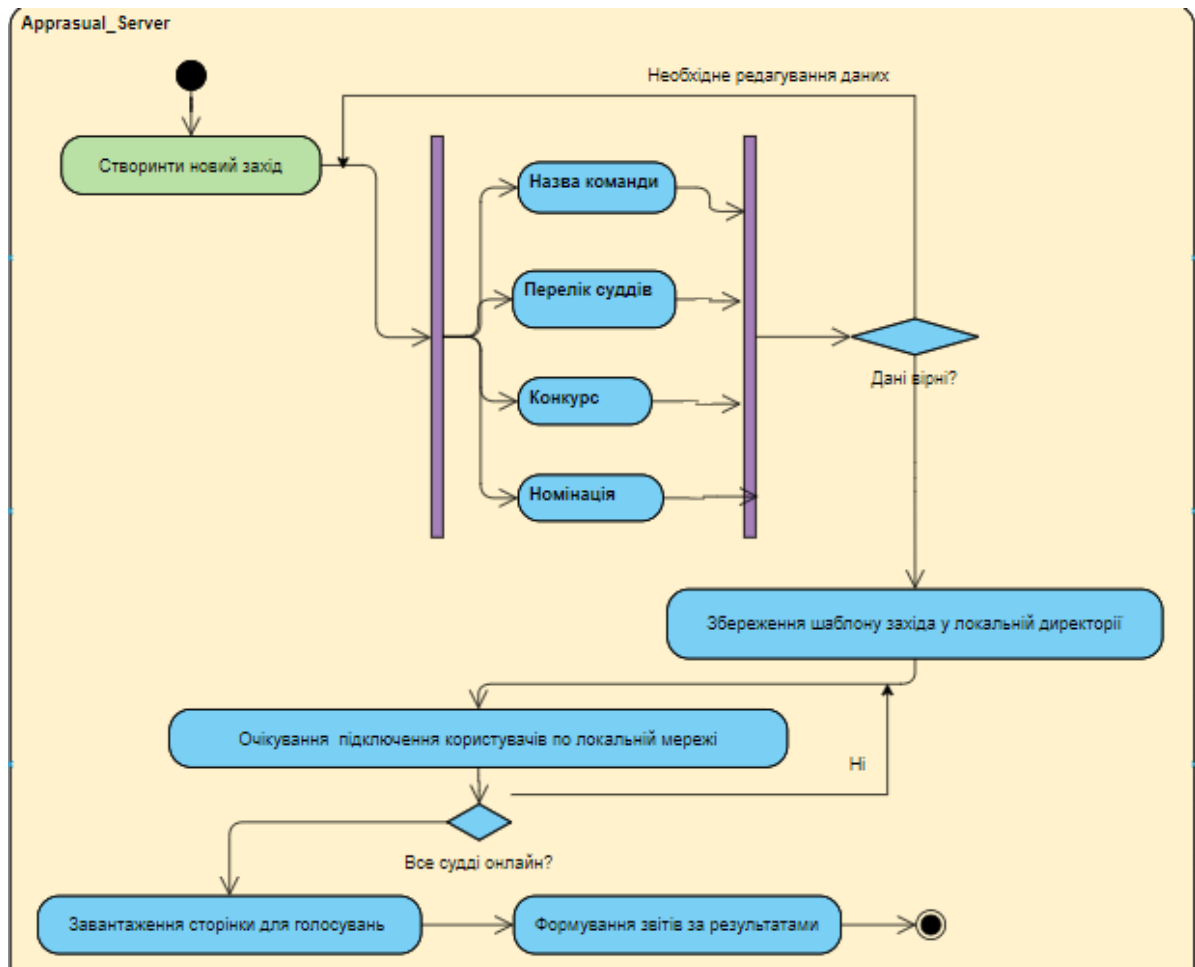


Рисунок 12 – Діаграма активності для серверного додатку

3.4 Опис роботи з програмою-сервером

На початку роботи з програмним додатком «AppraisalServer» відкривається вікно, яке забезпечує оператора сервера усіма необхідними функціями. До складу головного вікна входить дві вкладки: «Створити» та «Запустити».

На першій вкладці знаходиться:

- поле для назви змагання;
- перелік команд;
- перелік членів журі;
- номінації.

Кожен із цих компонентів дозволяє додавати та видаляти інформацію з них завдяки кнопкам «Додати» та «Видалити», відповідно (рис. 13). Розглянемо з яких компонентів складається вікно «Додати команду», як показано на рисунку 14.

Перше поле введення необхідно для запису групи, яка буде виступати на конкурсі, а друге – це назва команди (псевдонім), що дозволяє легше ідентифікувати команду серед інших.

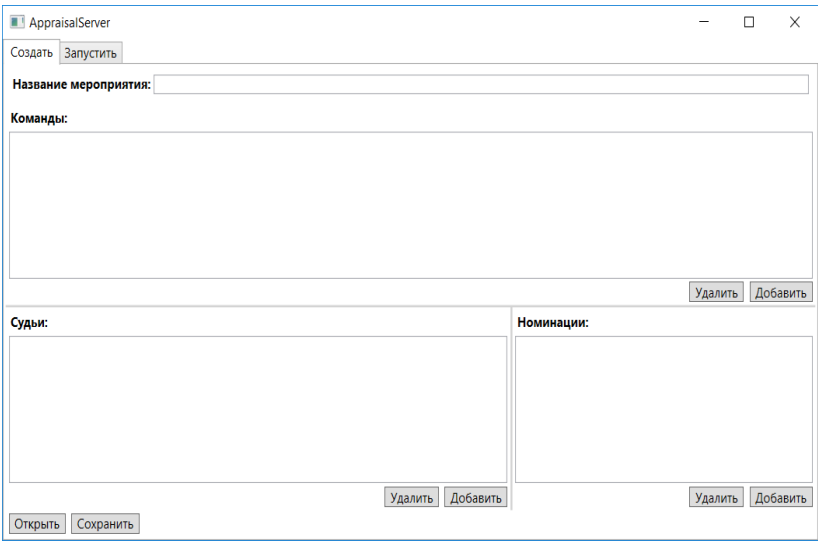


Рисунок 13 – Головне меню десктопного додатку

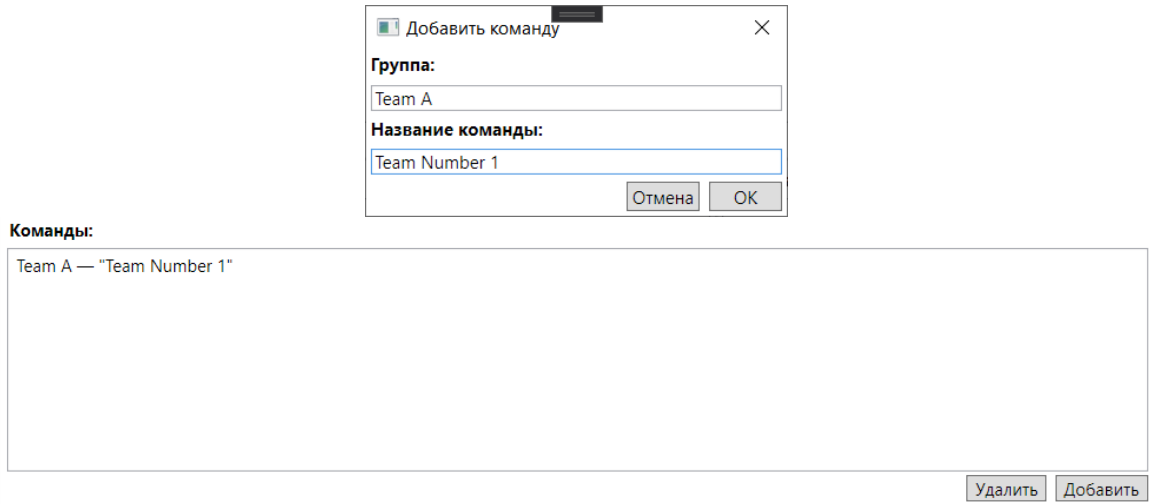


Рисунок 14 – Додавання нової команди

Наступним елементом додавання слугує вікно «Додати журі» (рис. 15), яке допомагає записувати прізвище, ім'я та по-батькові члена журі. Далі ці облікові записи знадобляться для входу із програми – клієнта.

Для того, щоб додати до переліку члена журі, достатньо натиснути лівою кнопкою миші на «ОК».

The image shows two parts of a software interface. On the left is a dialog box titled 'Добавить судью' (Add Judge) with a close button (X). It contains three text input fields: 'Фамилия:' (Surname) with 'Johnson', 'Имя:' (Name) with 'John', and 'Отчество:' (Patronymic) with 'A.'. At the bottom are 'Отмена' (Cancel) and 'ОК' (OK) buttons. On the right is a list box titled 'Судьи:' (Judges) containing the text 'Johnson John A.'. Below the list box are 'Удалить' (Delete) and 'Добавить' (Add) buttons.

Рисунок 15 – Додавання судді

Останнє модальне вікно у додатку «AppraisalServer» призначено для додавання у перелік нової номінації (рис. 16).

The image shows two parts of a software interface. On the left is a dialog box titled 'Добавить номинацию' (Add Nomination) with a close button (X). It contains a text input field for 'Название номинации:' (Nomination Name) with 'The Most Effective Team'. Below it are two spinners for 'Минимальная оценка:' (Minimum Rating) and 'Максимальная оценка:' (Maximum Rating), with values 0 and 10 respectively. At the bottom are 'Отмена' (Cancel) and 'ОК' (OK) buttons. On the right is a list box titled 'Номинации:' (Nominations) containing the text 'The Most Effective Team (0 ... 10)'. Below the list box are 'Удалить' (Delete) and 'Добавить' (Add) buttons.

Рисунок 16 – Додавання номінації

Номінація — це сутність, яка являє собою назву розділу конкурсу із мінімальною та максимальною оцінкою. Вона потрібна для того, щоб можна було сформувати таблицю результатів та вибрати переможців за їх середнім балом.

Також в наступних розділах буде розглянуте якими чином здійснюється вибір переможців. Іншою можливістю додатка «AppraisalServer» є збереження створеного проекту в директорії, яку вибере оператор.

Кожен проект зберігається завдяки серіалізації об'єктів в оперативній пам'яті. Розширення кожного з проектів є json, а назва збігається із введеною інформацією в полі «Назва конкурсу».

Наприклад, наступний проект матиме такі характеристики:

- команди («Team A», «Team B», «Team C»);
- журі (Johnson John A.);
- номінації (The Most Effective Team).

Завдяки цим параметрам на сервері формується новий проект, який можна зберегти, якщо натиснути на «Зберегти» лівою кнопкою миші. Також можна відкрити збережений раніше проект завдяки кнопці «Відкрити» (рис. 17).

The screenshot shows the 'AppraisalServer' application window. It contains several sections for creating a project:

- Buttons:** 'Создать' (Create) and 'Запустить' (Run) at the top left.
- Название мероприятия:** A text input field containing 'New Event Name'.
- Команды:** A list box containing 'Team A — "Team Number 1"', 'Team B — ""', and 'Team C — ""'. Below the list are 'Удалить' (Delete) and 'Добавить' (Add) buttons.
- Судьи:** A text input field containing 'Johnson John A.'. Below it are 'Удалить' and 'Добавить' buttons.
- Номинации:** A text input field containing 'The Most Effective Team (0 ... 10)'. Below it are 'Удалить' and 'Добавить' buttons.
- Bottom Buttons:** 'Открыть' (Open) and 'Сохранить' (Save) at the bottom left.

Рисунок 17 – Заповнений проект

Наступна вкладка на сервері «AppraisalServer» потрібна для запуску механізмів на локальній машині (рис. 18). Для того щоб це можливо було зробити необхідно бути підключеним до локальної мережі.

Кнопка поруч із IP-адресою потрібна для того, щоб в разі зміни локальної мережі можна було оновити її.

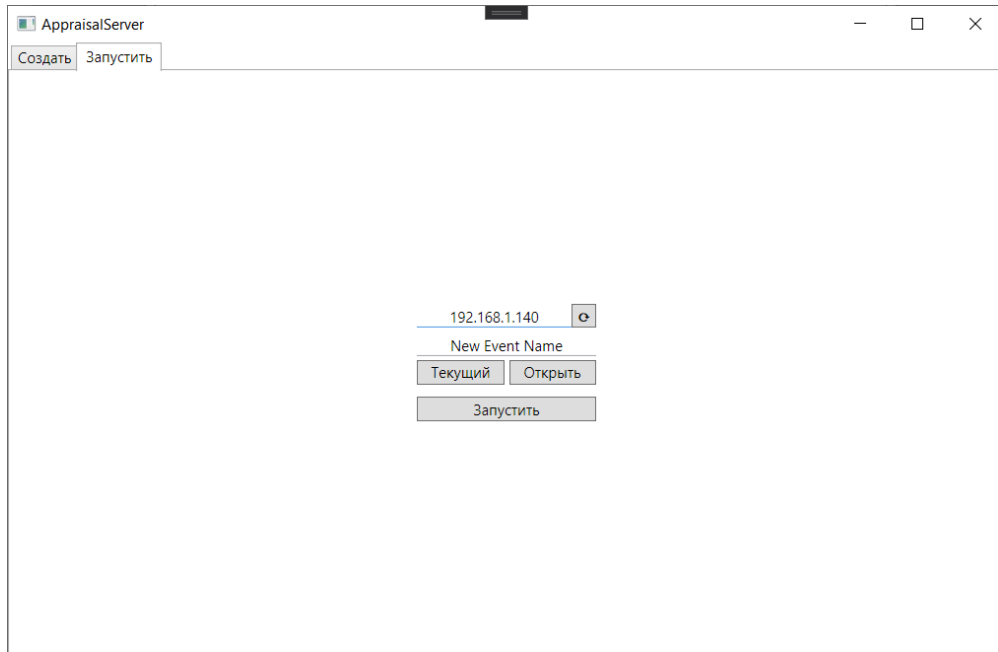


Рисунок 18 – Вкладка «Запустити»

Під IP-адресою розташована панель для вибору проекту (конкурсу), який буде запусканий. В оператора є дві можливості вибору проектів:

- 1) якщо проект був створений раніше, то він може його завантажити.

Для цього потрібно натиснути лівою кнопкою миші на «Відкрити», після цього буде запусканий проводник, який допоможе знайти необхідний файл із розширенням json.

Якщо файл є проектом він буде завантажений у оперативну пам'ять і буде використаний у наступних голосуваннях.

- 2) якщо проект потрібно завантажити, який щойно був створений у першій вкладці, користувачу потрібно натиснути на «Поточний».

В такому випадку усі введені дані будуть завантажені в оперативну пам'ять із попереднього розділу. Після запуску серверу він очікує підключення з боку клієнтів по локальній IP- адресі та TCP-протоколу. Після авторизації останнього члену журі буде завантажена сторінка розсилки голосування на девайси. Відправка розсилки здійснюється завдяки заповненим полям:

- номінація;
- група.

Нижче розташована таблиця результатів (рис. 17), в яку будуть вноситися оцінки суддів. Ця таблиця складається зі стовпців:

- номінація;
- команда;
- суддя;
- оцінка.

Завдяки цій гнучкій системі оператор може формувати список результатів після кожного виступу команди, щоб потім зробити точний графік результатів та прогнози на змагання. Заповнена таблиця результатів має такий вигляд.

The screenshot shows a window titled "AppraisalServer". At the top left, there are two buttons: "Создать" (Create) and "Запустить" (Run). The main area contains a form with two dropdown menus: "Номинация:" (Nomination) with the value "The Most Effective Team (0 ... 10)" and "Группа:" (Group) with the value "Team A — 'Team Number 1'". Below these are two buttons: "Отправить" (Send) and "Результаты" (Results). At the bottom, there is a table with four columns: "Номинация", "Команда", "Судья", and "Балл". The table is currently empty.

Рисунок 19 – Вікно таблиці оцінок суддів

Отже, кожний суддя має зробити оцінку кожної команди у кожній номінації. Наприкінці отримується декартова множина рядків (рис. 20).

The screenshot shows a window titled "AppraisalServer" with two tabs: "Создать" (Create) and "Запустить" (Run). The "Запустить" tab is active. Inside the window, there is a form with the following elements:

- A label "Номинация:" followed by a dropdown menu showing "The Most Effective Team (0 ... 10)".
- A label "Группа:" followed by a dropdown menu showing "Team C — """.
- A button labeled "Отправить" (Send).
- A button labeled "Результаты" (Results).

Below the form is a table with the following data:

Номинация	Команда	Судья	Балл
The Most Effective Team (0 ... 10)	Team A — "Team Number 1"	Johnson John A.	10
The Most Effective Team (0 ... 10)	Team B — ""	Johnson John A.	9
The Most Effective Team (0 ... 10)	Team C — ""	Johnson John A.	8

Рисунок 20 – Заповнена таблиця результатів

Для отримання результатів оператор сервера має визвати модальне вікно, натиснувши лівою кнопкою миші на «Результати».

Після цього буде відображене вікно, яке має такий вигляд. Воно складається з двох таблиць:

- таблиця місць;
- таблиця номінацій.

Перша таблиця дозволяє отримати перелік рядків, які являють собою:

- місце команди,
- назва команди та група,
- середній бал за виступ.

Наприклад, перше місце займає команда Team A із середнім балом 10. Наступна таблиця – це таблиця команд, які отримали найвищий бал за номінацію.

Таким чином виконується кумулятивне голосування, яке наприкінці змагань виведе декількох переможців у різних номінаціях. Це дуже зручно на випадок проведення декількох конкурсів за одну сесію, так, судді отримають загальну оцінку проведення всіх номінацій і буде зручно провести аналітичні висновки.

Приклад заповненого вікна даними щодо всіх проведених конкурсів представлено на рисунку 21.

Результаты конкурса

Таблица мест:

№	Команда	Балл за выступление
1	Team A — "Team Number 1"	10
2	Team B — ""	9
3	Team C — ""	8

Таблица номинаций:

Номинация	Команда
The Most Effective Team (0 ... 10)	Team A — "Team Number 1"

OK

Рисунок 21 – Вікно з результатами конкурсу

3.5 Опис роботи з програмою-клієнтом

Для роботи із програмою-клієнтом необхідно завантажити та встановити мобільний додаток «AppraisalClient». Початкова сторінка цього додатку буде мати такий вигляд.

Як бачимо, посередині розташоване поле введення, яке запрошує посилання у такому вигляді: IP-адреса:ТСР-порт. Порти розташовані таким чином, що кожен слідуючий суддя має попередній порт збільшений на одиницю, початковий порт завжди починається з 0.

Наприклад, маємо три суді, тоді перший суддя має 0 порт, другий – 1, а третій – 2. Отже суддя з нульовим портом має ввести, наприклад, таку адресу: 192.168.1.140:0. Це свідчить про те, що ми підключаємося до локального серверу із IP-адресою 192.168.1.140 та за портом 29040 + x, де x – це номер порта, який був записаний суддею.

Після перевірки правильності введення посилання, користувач повинен натиснути на «Далі» (рис. 22).

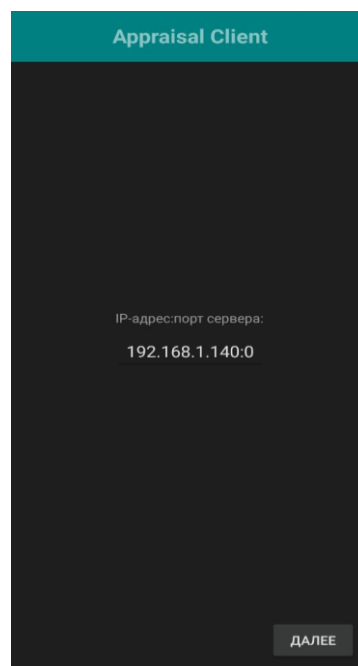


Рисунок 22 – Початкова сторінка «AppraisalClient»

Наступна сторінка потрібна для авторизації судді (рис. 23). Вона включає в себе випадające меню, яке складається з переліку суддей (які були відправлені з боку локального сервера).

Тут необхідно вибрати своє ім'я, у разі відсутності такого у списку – звернутися до адміністратору. Після вибору суддя буде направлений на сторінку очікування.

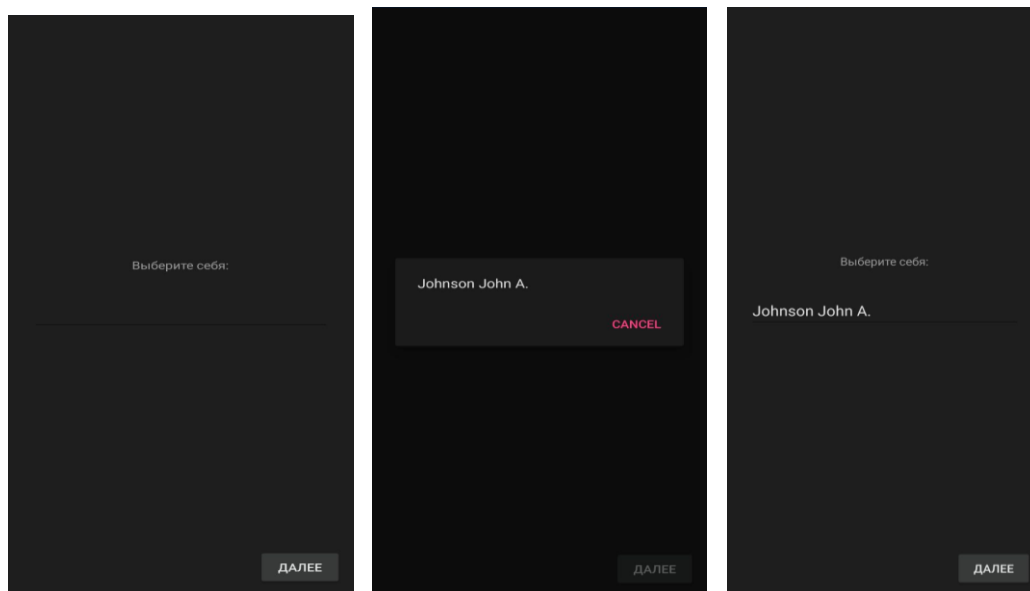


Рисунок 23 – Сторінки авторизації, вибору та очікування

Після розсилки з боку серверу клієнт отримує нову сформовану сторінку із елементом управління для вибору оцінки (рис. 24).

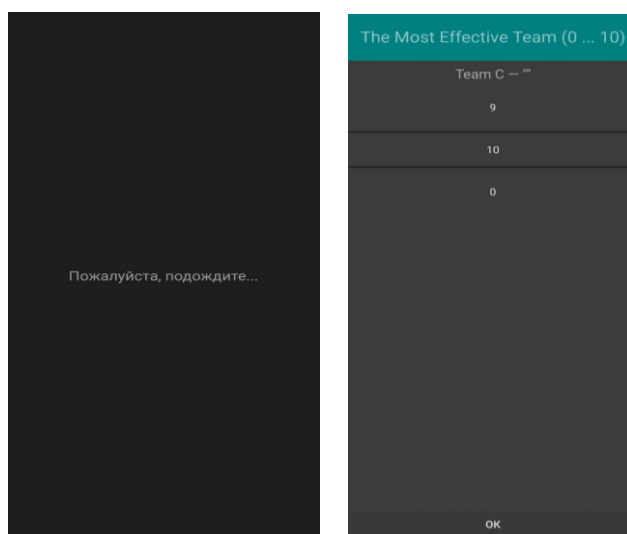


Рисунок 22 – Сторінки оцінювання та очікування інших суддів

На верхній панелі розташована інформація, щодо номінації. Після номінації, нижче іде назва команди та група, якій буде виставлятися оцінка.

Після натискання на кнопку «ОК» на сервер приходить інформація у вигляді строки-json, яка конвертується у рядок в таблиці результатів (рис. 24).

3.6 TCP сервер та його імплементація в проекті

Величезна кількість трафіку і взаємодії в мережі зараз відбувається по протоколу TCP (Transmission Control Protocol). Цей протокол гарантує доставку повідомлень і широко використовується в різних існуючих на сьогоднішній день програмах.

Для запуску сервера викликається метод Start(). Коли до сервера звертається клієнт, то ми можемо використовувати один з двох методів AcceptSocket() або AcceptTcpClient() для отримання відповідно об'єкта Socket або TcpClient, які будуть використовуватися для взаємодії з підключеним клієнтом.

```
public class ListenerObject
{
    private Timer timer;
    public Judge Judge { get; set; }
    public TcpClient TcpClient { get; private set; }
```

Клас TcpListener прослуховує вхідні підключення за певним портом. Для створення об'єкта слухача треба передати йому в конструктор об'єкт IPAddress і порт, по якому буде вестися прослуховування вхідних підключень.

```
public TcpListener TcpListener { get; private set; }
public bool isRegistered = false;
public event EventHandler<UpdateArgs> OnUpdate;
```

Для роботи з протоколом TCP в .NET призначені класи TcpClient і TcpListener. Ці класи будуються поверх класу System.Net.Sockets.Socket.

```

public ListenerObject(int port, string address)
{
    IPAddress ip = IPAddress.Parse(address);
    TcpListener = new TcpListener(ip, port);
    timer = new Timer { Interval = 1000 };
    timer.Tick += Timer_Tick;
}
private void Timer_Tick(object sender, EventArgs e)
{
    Task.Run(() =>
    {
        if (TcpClient is null) return;

```

Створюємо потік TCP-клієнта для «рукопотискання» з сервером:

```

NetworkStream stream = TcpClient.GetStream();
if (!isRegistered)
{
    Register(stream);
    isRegistered = true;
    return;
}
byte[] data = new byte[1024];
StringBuilder builder = new StringBuilder();
int bytes = 0;
do
{
    bytes = stream.Read(data, 0, data.Length);
builder.Append(Encoding.Unicode.GetString(data, 0, bytes));
}
while (stream.DataAvailable);
}

```

Передача по каналу зв'язку:

```

public void Register(Stream stream)
{
    string update = String.Empty;
for (int i = 0; i < MainWindow.Competition.Judges.Count; i++)
{
    update += MainWindow.Competition.Judges[i].ToString();
    if (i < MainWindow.Competition.Judges.Count - 1)
    {
        update += '\n';
    }
}
}

```

```
var data = Encoding.Unicode.GetBytes(update);
stream.Write(data, 0, data.Length); }
```

Сервер переходить у режим прослуховування каналу зв'язку:

```
public void WaitForClient()
{
    TcpListener.Start();
    TcpClient = TcpListener.AcceptTcpClient();
}
public void Start()
{
    timer.Start();
}
```

Метод відправки повідомлення:

```
public void Send(string message)
{
    byte[] data =Encoding.Unicode.GetBytes(message);
    TcpClient.GetStream().Write(data,
0,data.Length);
}
```

Методи `AcceptSocket()` і `AcceptTcpClient()` блокують потік, поки сервер не обслужить підключеного клієнта. Потім через методи, визначені в класах `TcpClient` і `Socket`, можна взаємодіяти з підключеним клієнтом: отримувати від нього дані або, навпаки, відправляти йому.

В кінці по завершенню роботи сервера треба викликати метод `Stop()`.

В даному випадку сервер прослуховує всі підключення по певному порту. У нескінченному циклі він отримує вхідні підключення.

У вхідних об'єктів `TcpClient` отримує мережевий потік через метод `client.GetStream()` і потім його використовує для відправки повідомлення.

Також можна поєднати клієнта з цим сервером. Тоді сервер буде відправляти клієнтам дані, а клієнти – отримувати їх.

3.7 TCP клієнт та його імплементація в проекті

Для створення клієнтської програми, яка працює по протоколу TCP, призначений клас `TcpClient`. Щоб з'єднатися з сервером TCP, в цьому класі визначено метод `Connect()`, якому передається назва хоста і порт. Щоб взаємодіяти з сервером `TcpClient` визначає метод `GetStream()`, який повертає об'єкт `NetworkStream`.

Через даний об'єкт можна передавати повідомлення сервера або, навпаки, отримувати дані з сервера. Після закінчення роботи з `TcpClient` його треба закрити методом `Close()`.

Для відправки даних при потоці використовується метод `Write()`. Для читання використовується метод `Read()` класу `NetworkStream`.

При цьому в обох випадках нам треба враховувати тип кодування: він повинен збігатися з кодуванням, в якій сервер відправляє або отримує повідомлення.

В даному випадку використовується кодування UTF8.

Повноцінний код клієнта, виконує читання з потоку:

```
public class ClientObject
{
    private Timer timer;
    public TcpClient TcpClient { get; set; }
    public event EventHandler<UpdateArgs> OnUpdate;
```

Визиваємо конструктор для клієнту:

```
    public ClientObject()
    {
        TcpClient = new TcpClient();
        var timerCallback = new TimerCallback(Tick);
        timer = new Timer(timerCallback);
    }
    public ClientObject(string server, int port) : this()
    {
        Connect(server, port);
    }
    private void Tick(object sender)
    {
```

```
if (TcpClient is null) return;
```

Тут мається на увазі, що по локальній адресі на певному порту запущений tcp-сервер. Ми очікуємо, що сервер буде передавати рядок в кодуванні UTF8, і для створення рядка визначається об'єкт `StringBuilder`.

```
Task.Run(new Action(() =>
{
    var stream = TcpClient.GetStream();
    var data = new byte[1024];
    StringBuilder builder = new StringBuilder();
    int bytes = 0;

    do
    {
        bytes = stream.Read(data, 0, data.Length);
        builder.Append(Encoding.Unicode.GetString(data, 0, bytes));
    }
    while (stream.DataAvailable);
    var update = builder.ToString();
    OnUpdate?.Invoke(this, new UpdateArgs(update));
}));
```

Визиваємо метод `Execute()` для зчитування запиту, якщо він поступив – у роботу вступає метод `Connect()`:

```
public void Execute(string request)
{
    var data = Encoding.Unicode.GetBytes(request);
    TcpClient.GetStream().Write(data, 0,
data.Length);
}
public void Connect(string server, int port)
{
    int _start = 28040;
    TcpClient.Connect(server, _start+port);
    timer.Change(0, 5000);}}
```

Для зчитування даних створюємо масив байтів:

```
byte [] data = new byte [256];
```

Однак оскільки дані можуть мати більший розмір, то зчитуємо їх з потоку в циклі `do..while`. При цьому перевіряється значення `stream.DataAvailable` на наявність даних в потоці.

ВИСНОВКИ

У дипломній роботі реалізовано дистрибутивний клієнт-серверний застосунок для віддаленого проведення локальних голосувань і референдумів. Для зручності використання та підключення користувачів проект реалізовано у десктопній версії та у вигляді мобільного додатку для клієнтів.

Робота над проектом проходила у декілька етапів. Спочатку було розглянуто предметну область та виконано аналітичний огляд існуючих аналогів. Все це допомогло поставити завдання та головну мету проекту.

На етапі проектування побудовані діаграми варіантів використання для опису завдань і функцій програмного додатку та його користувачів. Для роботи додатку було прийнято рішення використовувати протокол TCP.

Як подальший розвиток проекту можна додати збереження у базі даних графічної інформації щодо анкет учасників та процесів самих змагань.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ТОП-5 сервисов для проведения онлайн опросов и анкетирования. URL: <https://habr.com/ru/post/297686/> (дата звернення 21.03.2021).
2. ТОП-8 сервисов для создания онлайн опросов. URL: <https://coba.tools/compilation/top-8-servisov-dlya-sozdaniya-onlain-oprosoy> (дата звернення 21.03.2021).
3. Event. URL: <https://event.ru/overviews/10-poleznyih-prilozheniy-dlya-vedeniya-event-proektov/> (дата звернення 23.03.2021).
4. Самая популярная в мире система проведения опросов. URL: <https://www.surveymonkey.ru/> (дата звернення 28.03.2021).
5. Удобный конструктор опросов и форм обратной связи. URL: <https://simpoll.ru/> (дата звернення 02.04.2021).
6. Для чего нужен язык C#. URL: <https://thecode.media/c-sharp/> (дата звернення 02.04.2021).
7. SQLite. URL: <https://lecturesdb.readthedocs.io/databases/sqlite.html/> (дата звернення 10.04.2021).
8. Начало работы с WPF. URL: <https://metanit.com/sharp/wpf/1.2.php> (дата звернення 12.04.2021).
9. Битва C# JSON сериализаторов для .NET Core 3. URL: <https://habr.com/ru/company/otus/blog/515880/> (дата звернення 12.04.2021).
10. Как манипулировать и использовать JSON с C # и WinForms. URL: <http://helpexe.ru/programmirovanie/kak-manipulirovat-i-ispolzovat-json-s-s-i-winforms> (дата звернення 12.04.2021).
11. Xamarin и кросс-платформенная разработка. URL: <https://metanit.com/sharp/xamarin/1.1.php> (дата звернення 16.04.2021).

12. Как разработать мобильное приложение. URL: <https://overcoder.net/q/3324948/как-разработать-мобильное-приложение> (дата звернения 16.04.2021).
13. Почему C# программисты скоро будут нарасхват. URL: <https://proglib.io/p/c-sharp-popularity> (дата звернения 20.04.2021).
14. Что такое XML. URL: <https://javarush.ru/groups/posts/2287-что-такое-xml> (дата звернения 20.04.2021).
15. Веб-технологии для разработчиков. XML Введение. URL: https://developer.mozilla.org/ru/docs/Web/XML/XML_introduction (дата звернения 22.04.2021).
16. Диаграмма вариантов использования (UseCase diagram). URL: https://flexberry.github.io/ru/fd_use-case-diagram.html (дата звернения 25.04.2021).
17. UML Use Case Diagram Tutorial. URL: <https://www.lucidchart.com/pages/uml-use-case-diagram> (дата звернения 29.04.2021).
18. Диаграмма деятельности UML. Нотация диаграмм. URL: <https://planerka.info/item/diagrammy-deyatelnosti-uml/> (дата звернения 29.04.2021).