**АНОТАЦІЯ**

У представленій бакалаврській кваліфікаційній роботі спроектовано і розроблено систему для підтримки групової рольової гри під назвою «Мафія». Реалізовані програмні рішення надають великий спектр дій, які потрібні для якісного проведення групової гри. Серверна частина дає можливість створити нову кімнату, до якої можуть підключитись гравці з інших програм. Також вона дає змогу обчислити відповідну кількість карт певного типу та автоматично розіслати їх на клієнтські машини випадковим чином. Система на стороні користувача має наступні можливості: голосування, отримання ролі, контроль фолів, контроль відведеного часу, відображення статистики.

Серверна і клієнтська частини системи розроблені на мобільній платформі Android за допомогою мови програмування Java.Також розроблено сервіс ASMX для отримання статистичних даних. База даних з результатами розроблялась на мові [Transact-SQL](https://uk.wikipedia.org/w/index.php?title=Transact-SQL&action=edit&redlink=1), що є стандартом ANSI/ISO.

В економічній частині роботи детально розраховано рентабельність розробленої системи, а також обчислено собівартість та чистий дохід від реалізації запроектованої системи.

Подана бакалаврська кваліфікаційна робота складається із переліку умовних скорочень, вступу, семи розділів, висновків, списку використаних джерел та додатків. Загальний обсяг роботи 100 сторінок, робота ілюстрована 43 рисунком, а список проаналізованих літературних джерел містить 25 найменувань.

**ABSTRACT**

In the bachelor`s thesis a support system for role play game “Mafia” was constructed and developed. These program solutions provide wide range of activities that are needed for quality gaming.

Server part enables creating a new room and possibility for participants to connect using other applications. It also calculates the cards of a certain type and automatically sends it to client`s devices at random.

Client`s part includes: voting, getting a role, foul control, time control and statistic display. Server and client’s part are developed on Android mobile platform using Java. Service ASMX was developed for doing statistics. Results data base was developed on Transact-SQL which is an ANSI/ISO standard.

The thesis also includes economical part which analyzes cost-effectiveness, profit and cost of created system.

The given bachelor`s thesis consists of notations, introduction, seven chapters, references and three appendices. The length is one hundred pages. The thesis is illustrated with fourty three images and references include twenty five points.

ЗМІСТ

[ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ, АБРЕВІАТУР І ТЕРМІНІВ 8](#_Toc453329310)

[ВСТУП 9](#_Toc453329311)

[РОЗДІЛ 1. ХАРАКТЕРИСТИКА ОБ’ЄКТУ ПРОЕКТУВАННЯ ТА ПОСТАНОВКА ЗАДАЧІ 11](#_Toc453329312)

[РОЗДІЛ 2. ОГЛЯД ЛІТЕРАТУРИ 15](#_Toc453329313)

[2.1 Розгляд систем та інструментальних засобів 15](#_Toc453329314)

[2.1.1 Огляд мобільних операційних платформ 15](#_Toc453329315)

[2.1.2 Огляд мов програмування 18](#_Toc453329316)

[2.1.3 Огляд середовищ розробки 20](#_Toc453329317)

[2.1.4 Аналіз поширених типів БД 21](#_Toc453329318)

[2.2 Огляд аналога – програми MafiaProject 22](#_Toc453329319)

[РОЗДІЛ 3. СИСТЕМНИЙ АНАЛІЗ 26](#_Toc453329320)

[3.1 Побудова дерева проблеми та деревацілей 26](#_Toc453329321)

[3.1.1 Побудова дерева проблем 26](#_Toc453329322)

[3.1.2 Побудова дерева цілей 28](#_Toc453329323)

[3.2 Аналіз та вибір методів, алгоритмів та засобів розв’язання задачі 31](#_Toc453329324)

[3.2.1 Статуси в грі та їх обчислення 31](#_Toc453329325)

[3.2.1 Клієнт-серверна архітектура 31](#_Toc453329326)

[3.2.2 Soket 32](#_Toc453329327)

[3.2.3 Вибір мови програмування 33](#_Toc453329328)

[3.2.5 Вибір серверного рішення 36](#_Toc453329329)

[РОЗДІЛ 4. РОЗРОБКА ТА ВІДОБРАЖЕННЯ АЛГОРИТМІВ РОБОТИ КОМПОНЕНТІВ ТА СИСТЕМИ 38](#_Toc453329330)

[4.1 Побудова моделей бізнес-процесів за допомогою діаграм IDEF0 38](#_Toc453329331)

[4.3 Моделювання видів діяльності 42](#_Toc453329332)

[4.4 Діаграма класів 44](#_Toc453329333)

[4.5 Cхема бази даних 45](#_Toc453329334)

[РОЗДІЛ 5. РОЗРОБКА ПРОГРАМНОГО РІШЕННЯ 47](#_Toc453329335)

[5.1 Загальна структура програмного проекту (діаграма компонентів) 47](#_Toc453329336)

[5.2 Опис використаних сторонніх бібліотек та модулів 48](#_Toc453329337)

[5.2.1 Web-сервіс 48](#_Toc453329338)

[5.2.3 Клієнтський та серверний(локальний) додаток 50](#_Toc453329339)

[5.3 Розробка та опис інтерфейсу користувача 50](#_Toc453329340)

[5.3.1 Розробка веб сервісу 50](#_Toc453329341)

[5.3.2 Інтерфейси мобільних додатків 53](#_Toc453329342)

[Розділ 6. ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА 56](#_Toc453329343)

[6.1 Інструкції для користувачів 56](#_Toc453329344)

[6.2 Інструкції адміністратору 58](#_Toc453329345)

[6.3 Вимоги до апаратного забезпечення 61](#_Toc453329346)

[6.4 Тестування 62](#_Toc453329347)

[6.5 Опис проведених експериментів 63](#_Toc453329348)

[РОЗДІЛ 7. ЕКОНОМІЧНА ЧАСТИНА 66](#_Toc453329349)

[7.1 Економічна характеристика проектного рішення 66](#_Toc453329350)

[(програмного продукту) 66](#_Toc453329351)

[7.2 Інформаційне забезпечення та формування гіпотези щодо потреби розроблення продукту. 66](#_Toc453329352)

[7.3 Оцінювання та аналізування факторів зовнішнього та внутрішнього середовищ. 67](#_Toc453329353)

[7.4 Формування стратегічних альтернатив. 70](#_Toc453329354)

[7.5 Бюджетування. 72](#_Toc453329355)

[7.6 Вибір стратегії. 78](#_Toc453329356)

[ВИСНОВКИ 80](#_Toc453329357)

[ЛІТЕРАТУРА 81](#_Toc453329358)

[Додаток А 83](#_Toc453329359)

[Додаток В 85](#_Toc453329360)

[Додаток С 87](#_Toc453329361)

# ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ, АБРЕВІАТУР І ТЕРМІНІВ

БД – база данних;

ІС – інформаційнасистема;

КПК – кишеньковий портативний компютер.

Android – операційна система для мобільних пристроїв.

Android Studio — інтегроване середовище розробки(IDE) для платформи Android, представлене 16 травня 2013 року на конференціїGoogle I/O менеджером по продукції корпораціїGoogle.

ASMX Service – веб-сервіс, який надає можливість доступу до сховища, використовуючи відповідний інтерфейс користувацьких запитів.

IIS (Internet Information Services) — це набір серверів для декількох служб Інтернету від компанії Microsoft. IIS поширюється з операційними системами родини Windows NT.

Java – об’єктно-орієнтована мова програмування, випущена компанією Sun Mscrisystems у 1995 році як основний компонент платформи Java.

Soket – програмний інтерфейс для забезпечення обміну даних між різними процесами.

MS SQL – комерційна система керування базами даних, що розпов­сюджується корпорацією Microsoft.

Transact-SQL (T-SQL) – процедурне розширення мови SQL, створене компанією Microsoft (для Microsoft SQL Server) і Sybase (для Sybase ASE).

UML (unifiedmodellinglanguage) – мова графічного опису для об’єктного моделювання в сфері розробки програмного забезпечення.

WiFi – загальновживана назва для стандарту ІЕЕЕ 802.11 передачі цифрових потоків даних по радіоканалах.

XML (ExtensibleMarkupLanguage) — запропонований консорціумом World Wide Web(W3C) стандарт побудови мов розмітки ієрархічно структурованих даних для обміну між різними застосунками, зокрема, через Інтернет.

# ВСТУП

Стрімкий розвиток комп’ютерних наук та інформаційних технологій відкрив новий етап розвитку систем, які дають змогу полегшити процеси управління у різних сферах суспільства. Ще тільки вчора мобільні телефони та планшети були для нас чимось новим і дуже новітнім, а зараз цей пристрій більш поширений за портативні комп’ютери.

Комп’ютерні технології змінюють наше повсякденне життя, тому на даний момент дуже популярною є тенденція інтеграції комп’ютерних технологій у всі сфери людської діяльності.

Важко навіть уявити всю користь, яку надає комп’ютеризація. Оскільки вона дає змогу пришвидшити різноманітні процеси в системах різного роду, перевести їх на якісно новий рівень, а також вирішити задачі, які раніше були недоступними. Використовуючи такі системи, ми економимо велику кількість ресурсів, як людських, так і матеріальних. Відповідно, в рази підвищується якість управління в різноманітних сферах.

Автоматизація значною мірою увійшла і в світ розваг. Тому досить поширеною стала автоматизація різноманітного роду групових рольових ігор, що дає змогу людям у різних куточках планети проводити свої улюблені ігри, перебуваючи біля екранів комп’ютера. Зараз ми можемо знайти дуже велику кількість Web-орієнтованих програмних рішень для цієї сфери, але відомі рішення є однобічними і неповністю функціональними, що дає нам змогу розвинути ідею з креативністю та інноваційністю.

Метою бакалаврської кваліфікаційної роботи є створення програмного рішення для підтримки групових рольових ігор. Це дасть змогу групі людей, які мають мобільні пристрої і не мають виходу в Інтернет, створити локальну мережу, в якій вони зможуть обмінюватись різноманітними повідомленнями та здійснювати легке управління процесом гри, а при наявності Інтернет-з’єднання зберегти результати у встановленому сховищі даних, для ведення статистики. Мої додатки спростять проведення гри, як в часі, так і в фінансовому плані.

Обєктом дослідження є групові рольові ігри, які проводяться з розважаль­ною та інтелектуальною метою. Всі вони мають певні труднощі, які стосуються їх проведення за умови великою кількості учасників. Тому розроблена система дасть змогу відкрити повний потенціал цього роду ігор і приверне до себе ще більше охочих та бажаючих.

Предметом дослідження є рольова гра під назвою «Мафія». Ця гра за досить короткий термін завоювала велику кількість прихильників по всьому світі. І все це завдяки використанню основних найцікавіших можливостей індивідуума, а саме: комунікація, аналіз, підтримка, вміння приховувати емоції, вміння працювати в команді.

Ця гра дуже подібна на свого роду шоу, яке веде за собою боротьбу за виживання. Тому можна виділити два різних типи ігор: змагання і вистава. «Мафія» досить добре об’єднує ці два типи ігор. Вона не пов’язана з грошима, не потребує хорошої фізичної підготовки чи великих фінансових витрат. Гравцям вона приносить інтелектуальне задоволення колективної роботи.

Ця рольова гра є унікальною, оскільки базується на спілкуванні та обговоренні між учасниками різних команд – червоними і чорними. У цій грі завжди відбувається об’єднання у певні альянси. Обговорення проходить шляхом безупинних суперечок. Завдання гри – перемога власною командою.

Необхідність у використанні мобільних додатків для проведення гри проявляється гостро, коли ми маємо досить невелике приміщення, не маємо ігрових карт, відсутня стереосистема, оскільки цю гру часто проводять у темряві, що не дає можливості гучно вмикати музику, яка повинна приховувати рухи та жести, а при тісному контакті з іншими гравцями це взагалі неможливо.

Практична цінність розробки полягає в тому, що створення програмних рішень і автоматизація процесу проведення є дуже потрібними для всіх любителів рольових ігор. Через зміну деяких конфігурацій розроблена система може бути перенесена для підтримки іншої рольової гри, оскільки основні процеси гри залишаються незмінними.

# РОЗДІЛ 1. ХАРАКТЕРИСТИКА ОБ’ЄКТУ ПРОЕКТУВАННЯ ТА ПОСТАНОВКА ЗАДАЧІ

Вважається, що гру придумав студент Московського державного університету Дмитро Давидов, який навчався на факультеті психології. З самого початку у цю гру грали у гуртожитках і навіть в аудиторіях і коридорах цього університету. Гра дуже стрімко почала поширюватись і в інших навчальних закладах СРСР, оскільки деякі студенти проводили літні канікули в студентських таборах, в яких вони розповідали всім про цю цікаву рольову гру. Сам автор цієї гри після закінчення навчання в університеті залишився викладати психологію для іноземних студентів, і його студенти, які в той час були з різних країн, а саме: [Словенії](https://uk.wikipedia.org/wiki/%D0%A1%D0%BB%D0%BE%D0%B2%D0%B5%D0%BD%D1%96%D1%8F), [Польщіа](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%8C%D1%89%D0%B0), [Угорщини](https://uk.wikipedia.org/wiki/%D0%A3%D0%B3%D0%BE%D1%80%D1%89%D0%B8%D0%BD%D0%B0), [Словаччини](https://uk.wikipedia.org/wiki/%D0%A1%D0%BB%D0%BE%D0%B2%D0%B0%D1%87%D1%87%D0%B8%D0%BD%D0%B0), [Чехії](https://uk.wikipedia.org/wiki/%D0%A7%D0%B5%D1%85%D1%96%D1%8F), [Румунії](https://uk.wikipedia.org/wiki/%D0%A0%D1%83%D0%BC%D1%83%D0%BD%D1%96%D1%8F), [Бельгії](https://uk.wikipedia.org/wiki/%D0%91%D0%B5%D0%BB%D1%8C%D0%B3%D1%96%D1%8F) - після повернення на батьківщину також поширили її на своїх теренах. В США вперше поява цієї гри документується 1989-м роком у літньому таборі штату Пенсильванія.

Найчастіше проводять класичний тип гри мафії - вона більш зрозуміла для початківців.

Ведучий випадковим чином роздає учасникам гри [карти](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D0%BE%D0%B4%D0%B0_%D0%BA%D0%B0%D1%80%D1%82), які в свою чергу, є ролями на гру.

* Гравці, яким належать червоні карти, відносяться до команди «чесних мешканців міста», вони між собою не знайомі (існує інша назва - «мирні жителі»). Один із цих гравців отримує унікальну червону карту, яка присвоює гравцеві роль «комісар».
* Інші гравці, які отримали чорні карти, відносяться до команди *«мафії»*.

Весь сценарій гри побудований на двох фазах. Перша фаза - «день», а   
друга - «ніч».

Всі гравці закривають очі. Потім ведучий оголошує всім членам мафії, щоб вони відкрили очі і визнали своїх спільників. Члени мафії вибирають "жертву", мовчки жестом вказуючи послідовність цифр, і потім закривають очі знову.

Аналогічний процес відбувається для інших ролей з нічними діями. У тому випадку коли гравець отримав роль детектива, ведучий має правдиво відповісти детективу стосовно кольору гравця за столом.

Ніч може супроводжуватися легким постукуванням, щоб замаскувати звуки, що видаються під час жестикуляції.

Цю рольову гру варто проводити в кількості гравців від 6 до 15 і навіть більше, зрозуміло, якщо кількість гравців перевищує 10, то варто створювати новий стіл з новим набором гравців. І тут вимальовується знову нова проблема: турнір. Ми можемо проводити з однією колодою карт лише одну гру. Це можна уявити на прикладі однопоточності. Багатопотоковість в моєму випадку – це проведення паралельних ігор за різними столами незалежно від кількості наборів для гри.

Нам потрібно забезпечити зв'язок між користувачами - це все повинно відбуватись через сервер ведучого.

Для зв’язку нам потрібно створити локальну мережу. Це можна зробити як за допомогою комп’ютера, так і через мобільний пристрій, використовуючи вбудовану у більшість мобільних пристроїв функцію «Hotspot».

Тому мені потрібно використати технологію зв’язку. Оскільки у моїх місцях проведення даної гри відсутній Інтернет, то я повинен забезпечити зв'язок трішки по-іншому. В даному випадку можна використати технологію Socket для зв'язку додатків у моїй мережі.

Перевагою буде те, що мені доступ до Інтернету буде потрібен лише у випадку занесення результатів на сервер. Це буде таблицяю з результатами ігор, щоб побудувати рейтинг корисності кожного гравця.

Далі це буде виглядати так: гравці займають свої місця за ігровим столом, а ведучий створює гру і чекає на підключення всіх гравців. Після того він налаштовує гру - це дуже важлива частина, яка формує ролі, які будуть присутні у грі, це все повинно виконуватись автоматизовано, оскільки не завжди формула, за якою обчислюється кількість гравців певного типу, підійде нам з першого разу- її, відповідно, потрібно буде корегувати. І цю можливість повинна забезпечити програма.

Також нам потрібно створити можливість формування певного набору ролей, які в кінцевому варіанті ведучий випадковим чином повинен розіслати на пристрої користувачів. Авже гравці зможуть переглянути роль на своєму пристрої, і зрозуміти, яку роль на цю гру він отримав.

Залежно від отриманої ролі, гравцям буде доступний певний функціонал. Наприклад, розглянемо сценарій гри за роль Комісар.

Ця роль відповідає класу мирних гравців. Вона дає можливість кожної ночі перевірити одного з гравців, аби отримати інформацію про те, чи являється даний гравець на стороні «поганих», або ж «добрих».

Також є ряд інших додаткових ролей на гру, а саме:

[*Лікар*](https://uk.wikipedia.org/wiki/%D0%9B%D1%96%D0%BA%D0%B0%D1%80) чи *Доктор –* може зберегти жителю життя*.*

[*Журналіст*](https://uk.wikipedia.org/wiki/%D0%96%D1%83%D1%80%D0%BD%D0%B0%D0%BB%D1%96%D1%81%D1%82)—не перевіряє, а порівнює двох жителів на «одинаковість» статусів.

Також відомі інші типи ігор, в яких використовуються додаткові ролі, які можна використати один раз протягом гри. Цей варіант мені дуже подобається, оскільки додає до гри певний колорит. Поки що мені відомі такі додаткові ролі:

*Заміна ролі* - якщо ти отримав цю карту, то можеш помінятися ролями з будь-яким гравцем (помінятися картами), але не знаючи, хто він.

*Останній постріл*- якщо під час гри тебе вбили, ти маєш право останнього пострілу - можеш убити кого-небудь з гравців. Він теж вважається вбитим і виходить з гри.

*Допомога* - покажи цю карту, коли когось із членів твоєї команди вб'ють, і спаси його від смерті.

*Несподіваний постріл* - покажи цю карту і вкажи на того, кого ти хочеш вбити. Гравець, на якого ти вказав, відкриває свою карту і виходить з гри.

*Бронежилет* - якщо тебе вб'ють, ти не відкриєш свою карту і не показуйєш, хто ти. Просто покажи цю карту - і залишайся в грі.

*Шпигун* - використовуй цю карту в кінці раунду і подивися на карту одного з гравців - дізнайся, хто він - мафіозі або поліцейський.

На стороні ведучого потрібно забезпечити вкладки, в яких буде видно всіх гравців і ті ролі, які вони отримали. Відповідно, потрібно забезпечити таку функцію, як штрафний бал для певного гравця. І одразу повідомити гравця про це на його пристрої.

Також секундомір – аби не переключатися між програмами під час гри, що є дуже незручно.

І, що є досить вагомим, є ще одна досить цікава функція, яку неодмінно потрібно реалізувати, це голосування. Рахувати підняті руки буде не дуже ефективно. І це займає зайвий час.

І насамкінець, ведучий, при наявності Інтернет-з’єднання, повинен занести результати ігор на сервер з результатами.

Насправді, функціонал можна розширювати до безмежності, але, я думаю, все-таки доцільніше реалізувати головні функції, без яких обійтись буде важко.

# РОЗДІЛ 2. ОГЛЯД ЛІТЕРАТУРИ

## 2.1 Розгляд систем та інструментальних засобів

### 2.1.1 Огляд мобільних операційних платформ

**SymbianOS** — мобільна операційна система (ОС) і обчислювальна платформа, яка спочатку була розроблена як проект із закритим вихідним кодом ОС для КПК, в 1998 році SymbianLtd. SymbianOS був нащадком EPOCPsion і працює виключно на процесорах ARM, хоча існувала (але не випущена) x86 версія. Symbianі використовується багатьма великими брендами мобільних телефонів, такими як Samsung, Motorola, SonyEricsson і, перш за все, Nokia. Як першовідкривач, який встановив смартфон промисловості, він був найпопулярнішим ОС для смартфонів по всьому світу в середньому до кінця 2010 року.

**BlackBerryOS (RIM)** — мобільна операційна система, розроблена компанією BlackBerry Ltd для своєї лінії портативних пристроїв. Операційна система забезпечує багатозадачність і підтримує спеціалізовані пристрої введення, які були прийняті на BlackBerry Ltd. для використання в своїх кишенькових комп'ютерах, зокрема коліщатком навігації, трекболом, а зовсім недавно - сенсорної панелі і сенсорного екрану.

**iPhoneOS** — це мобільна операційна система, створена і розроблена Apple Inc. і поширюється виключно для обладнання Apple. Ця операційна система, яка в даний час використовується в багатьох мобільних пристроях компанії, в тому числі iPhone, IPad і IPod Touch. Це друга найпопулярніша мобільна операційна система в світі за обсягом продажів після Android. IPad- обгортка системи є також другою за популярністю, за обсягом продажів проти Android[1] починаючи з 2013 року, коли продажі Android-планшетів виросли на 127%.

**WindowsMobile**і**WindowsCE** — сімейство мобільних операційних систем, розроблених Microsoft для смартфонів і КПК. Його походження сходить до Windows CE в 1996 році, хоча сама Windows Mobile вперше з'явилась в 2000 році як PocketPC 2000. Вона була перейменована в "Windows Mobile" в 2003 році, після чого вона пройшла у кількох версіях (за аналогією з настільними версіями Windows) і була спрямована на бізнес і корпоративних споживачів. До 2007 року вона була найпопулярнішим програмним забезпеченням в США, але ця популярність була втрачена в наступні роки. У лютому 2010 року, перед натиском з боку конкурента OSs (включаючи прошивку і Android) Microsoft оголосила про Windows Phone, що замінює Windows Mobile. В результаті, Windows Mobile є застарілою операційною системою. Windows Phone не сумісна з Windows Mobile -пристроями і програмним забезпеченням. Остання версія Windows Mobile випущена після анонсу Windows Phone [2]. Після цього Microsoft припинила розробку на Windows Mobile для того, щоб зосередитися на Windows Phone.

**Android** — це мобільна операційна система у даний час розробляється Google, заснована на ядрі Linux і призначена, в першу чергу, для сенсорних мобільних пристроїв, таких як смартфони і планшети. Вона була найбільш продаваною ОС на планшетних ПК і смартфонахів з 2013 року і має найбільшу базу встановленого обладнання [3]. Інтерфейс Android, в основному, базується на прямій маніпуляції, використовуючи жести, які вільно відповідають реальним діям, таких як натиск пальцем, щипання, щоб маніпулювати об'єктами на екрані. Також є сервіс для рознесення додатків для Android - це Google Play, понад мільйон опублікованих додатків для Android і 50 мільярдів завантажень станом на липень 2013 р. [4]. На додаток до сенсорних пристроїв компанія Google удосконалила Android для телевізорів, автомобілів і наручних годинників, кожен із спеціалізованим інтерфейсом. Варіанти і роздвоєні версії Android також використовуються на ноутбуках, ігрових приставках, цифрових камерах і інших платформах для смартфонів, з відкритим вихідним кодом. Платформа базується на Linux. В 2012 році, за даними доля цієї ОС склала 75%.

Інші, менш відомі мобільні операційні системи:

**PalmOS** — це мобільна операційна система, розроблена компанією Palm, Inc., INITIALLY, для персональних цифрових помічників (PDA). В 1996 Palm OS була розроблена для простоти використання з сенсорним екраном на основі графічного інтерфейсу користувача. Вона забезпечена набором базових додатків для управління персональною інформацією. Пізніші версії ОС були розширені для підтримки смартфонів. Кілька інших ліцензіатів виготовили пристрої під управлінням Palm OS.

**PalmwebOS** — Linux в якості ядра, на основі багатозадачної операційної системи для смарт-пристроїв, таких як телевізори і smartwatches та смартфонів. Раніше вона була мобільною операційноюї системою. Спочатку розроблена Palm, (яка була придбана компанією Hewlett-Packard), HP зробив платформу з відкритим вихідним кодом. Операційна система згодом була продана компанії LG Electronics. HP і раніше займав багато патентів, пов'язаних з WebOS. У січні 2014 року компанія Qualcomm оголосила, що вона придбала патенти на технології від HP, які включають в себе всі патенти WebOS.

**Linux** — не змогла широко розповсюдитись, але традиційно цей напрямок вважається перспективним. В Азії смартфони на базі Linux є дуже поширені . Платформа Maemo використовується в планшетах Nokia770-N810 і смартфонах NokiaN900, OpenMoko використовується в Neo1973 та NeoFreeRunner.

**Bada** — операційна система для мобільних пристроїв, таких як смартфони і планшетні комп'ютери. Вона була розроблена Samsung Electronics. Його назва походить від слів, що означають "океан" чи "море" на корейській мові.

**AliyunOS** — дистрибутив Linux, призначений для смартфонів на базі проекту Open Source Android. Вона розробляється AliCloud - дочірня компанія китайської компанії Alibaba Group. AliyunOS був випущений в Китаї 28 липня 2011р. Перший пристрій для запуску був K-Touch W700. Для AliCloud потрібно було три роки та 1600 інженерів, щоб розробити Yun ОС від коду бази проекту Android Open Source. Станом на травень 2012 року 1 мільйон Aliyun-смартфонів було продано.

В таблиці 2.1 наведено статистику по кількості проданих смартфонів на певній платформі за певний рік.

*Таблиця 2.1*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Year** | **Android (Google)** | **iOS (Apple)** | **Windows Mobile/Phone (Microsoft)** | **BlackBerry (formerly RIM)** | **Symbian (Nokia)** | **Palm/WebOS (Palm/HP)** | **Bada (Samsung)** | **Other** |
| **2007** |  | **3,20** | **14,70** | **11,78** | **77,67** | **1,75** |  |  |
| **2008** |  | **11,41** | **16,50** | **23,16** | **72,94** | **2,53** |  |  |
| **2009** | **6,70** | **24,88** | **15,03** | **34,36** | **80,87** | **1,39** |  |  |
| **2010** | **67,32** | **46,61** | **12,39** | **47,46** | **111,57** |  |  |  |
| **2011** | **229,42** | **89,16** | **8,78** | **51,53** | **93,43** |  | **9,62** | **14,25** |
| **2012** | **452,61** | **130,23** | **16,95** | **34,22** |  |  | **15,91** | **47,21** |
| **2013** | **768,71** | **151,79** | **30,85** | **18,62** |  |  |  | **18,82** |
| **2014** | **1,003,67** | **191,42** | **35,14** | **7,92** |  |  |  | **5,76** |

За таблицею явно відстежується лідерство мобільної платформи Android, яка за кількістю продажів випереджає найближчого конкурента в 5 раз.

### 2.1.2 Огляд мов програмування

**C#** (вимовляється*Сі-шарп*) — об'єктно-орієнтована мова програмування, розроблена командою на чолі з Андерсом Гейлсбергом для Microsoft.

Програма, написана на цій мові, компілюється Common Intermediate Language (CIL) - це спеціальний проміжний код, який виконується в таких середовищах як NET Framework, Mono або DotGNU. Виконання компіляції в операційній системі без такого середовища є неможливим.

**Java** - мова програмування загального призначення. Вона призначена для того, щоб розробники додатків могли написати один раз, а програма працювала будь-де. Java-додатки зазвичай компілюються в байт-код, і це може працювати на будь-який віртуальній машині Java (JVM), незалежно від комп'ютерної архітектури. Станом на 2016 рік Java є одним з найпопулярніших мов програмування в використанні, зокрема для клієнт-серверних веб-додатків. Java спочатку була розроблена Джеймсом Гослінг в Sun Microsystems (яка з тих пір придбана корпорацією Oracle) і випущена в 1995 році в якості одного з основних компонентів платформи Java компанії Sun Microsystems. Мова отримує більшу частину його синтаксису від C і C++.

**Objective-C** — являє собою універсальну, об'єктно-орієнтовану мову програмування з Smalltalk-стиль, який додає до мови програмування C обмін повідомленнями. Це була основна відмінність мови програмування, яка використовується Apple для OS X і операційних систем IOS, а також їх відповідні інтерфейси прикладного програмування (API): Cocoa і Cocoa Touch до введення Swift.

Мова програмування Objective-C спочатку була розроблена на початку 1980-х років. Вона був обрана в якості основної мови, використовуваного NeXT для своєї операційної системи NeXTSTEP, з якої OS X і ОС IOS є похідними. Портативні програми Objective-C, які не використовують бібліотеки Cocoa або Cocoa Touch, або ті, які використовують частини, можуть бути перенесені для інших систем, можуть бути скомпільовані для будь-якої сучасної системи, підтримуваної GNU Compiler Collection (GCC) або Clang.

**C++** (Сі-плюс-плюс) — це мова програмування загального рівня призначення. Вона має імперативний, об'єктно-орієнтований і узагальнений вид, забезпечуючи при цьому також об'єктами низького рівня для маніпулювання пам'яттю.

Вона була розроблена з ухилом в бік системного програмування та вбудованих, обмежених ресурсів і великих систем, з продуктивністю, ефективністю і гнучкістю. C ++ Ключовими складниками мови є програмне забезпечення інфраструктури і обмежені ресурсами додатки, в тому числі додатки для настільних комп'ютерів, серверів (наприклад, електронної комерції, веб-пошуку або серверів SQL) і продуктивність критично важливих додатків (наприклад, телефонних комутаторів або космічних зондів). C ++ являє собою скомпільовану мову з реалізаціями на багатьох платформах.

**2.1.3 Огляд середовищ розробки**

**Microsoft Visual Studio** — інтегроване середовище розробки (IDE) від Microsoft. Воно використовується для розробки комп'ютерних програм для Microsoft Windows, а також веб-сайтів, веб-додатків і веб-сервісів. Visual Studio використовує платформи розробки програмного забезпечення Microsoft, такі як Windows API, Windows Forms, Windows Presentation Foundation, Windows Store і Microsoft Silverlight. Він може виробляти як машинний код і керований код.

Visual Studio включає в себе редактор коду, що підтримує технологію IntelliSense (компонент завершення коду), а також рефакторінга коду. Інші вбудовані інструменти включають в себе конструктор форм для побудови GUI- додатків, веб-дизайнер, клас дизайнера і проектувальника схеми бази даних. Середовище приймає велику кількість плагінів, які розширюють функціональність майже на кожному рівні, в тому числі надає підтримку для систем джерела управління (наприклад, Subversion) і додає новиі набори інструментів, такі як редактори та візуальні конструктори для предметно-орієнтованих мов або набори інструментів для інших аспектів життєвого циклу розробки програмного забезпечення.

**AndroidStudio** — є офіційним інтегрованим середовищем (IDE) для розробки на платформу Android.

Це було оголошено 16 травня 2013 року на конференції GoogleI/O. AndroidStudio знаходиться у вільному доступі під ApacheLicense 2.0.

Середовище було в ранній стадії попереднього доступу, починаючи з версії 0.1 в травні 2013 року, а потім вступила стадія бета-тестування, починаючи з версії 0.8, яка була випущений в червні 2014 г. Перша стабільна збірка була випущена в грудні 2014 року, починаючи з версії 1.0.

На основі програмного забезпечення JetBrainsIntelliJIDEA, AndroidStudio розроблений спеціально для розробки Android. Він доступний для завантаження на Windows, MacOSX і Linux і може замінити інструменти Eclipse.

**2.1.4 Аналіз поширених типів БД**

**База даних** являє собою організовану сукупність даних. Це колекція схем, таблиць, запитів, звітів, уявлень і інших об'єктів. Дані, як правило, організовуються для моделювання аспектів реальності, таким чином підтримує процеси, що вимагають інформації.

Система управління базами даних (СКБД) являє собою додаток - комп'ютерна програма, яка взаємодіє з користувачем, іншими додатками, а також бази даних сама по собі для збору і аналізу даних. СУБД загального призначення розроблена, щоб дозволити визначення, створення, обробку запитів, оновлення і адміністрування баз даних. Добре відомі СУБД включають MySQL, PostgreSQL, Microsoft SQL Server, Oracle, Sybase, SAP HANA і IBM DB2. База даних не як правило, тому різні СУБД можуть взаємодіяти за допомогою стандартів, таких як SQL і ODBC або JDBC. Системи управління базами даних часто класифікуються відповідно до моделі бази даних, яку вони підтримують; найпопулярніші системи управління базами даних з 1980 року підтримали реляційну модель, яку представлено на мові SQL. Іноді СУБД вільно згадується як база даних.

Бази даних класифікують за різними критеріями. За моделлю організації даних розрізняють такі бази даних:

* **Ієрархічна**. Ієрархічна база даних може бути представлена як дерево, що складається з об'єктів різних рівнів. Між об'єктами існують зв'язки типу «предок-нащадок». При цьому можлива ситуація, коли об'єктне має нащадків або має їх декілька, тоді як у об'єкта-нащадка обов'язково тільки один предок.
* **Мережна**.Така база даних подібна до ієрархічної, за винятком того, що кожен об'єкт може мати більше одного предка.
* **Реляційна.** Реляційна база даних зберігає дані у вигляді таблиць. Найуживаніші СКБД використовують реляційну модель даних.
* **Об'єктно-орієнтована.** У базі даних цього виду дані оформляють у вигляді моделей об'єктів.

За розміщенням даних виділяють такі види баз:

* **Локальна**, або **централізована**. Така база даних підтримується на одному комп'ютері.
* **Розподілена.** Частини такої бази даних розміщують на різних комп'ютерах мережі.

## 2.2 Огляд аналога – програми MafiaProject

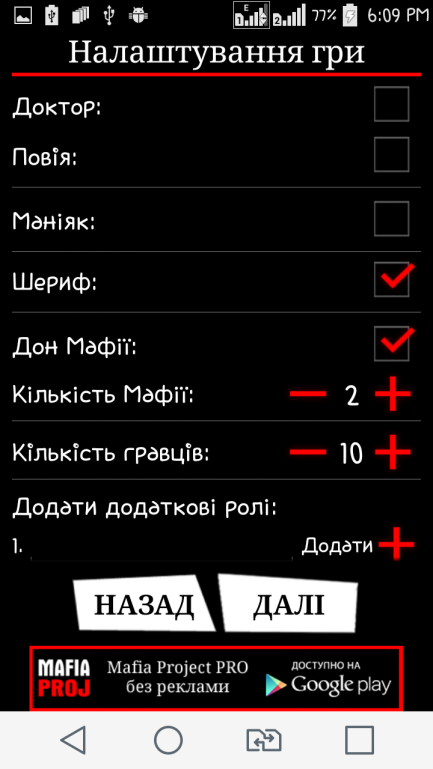
Насправді знайти подібні прототипи програми, яку я описав вище, не вдалося,оскільки такі системи ще не розроблялися. Але зовсім недавно я натрапив на один додаток, який дозволяє спростити проведення гри з боку ведучого. Цей додаток має назву MafiaProject. Вигляд запуску програми наведено на мал.2.1.



*Мал. 2.1 Вигляд програми MafiaProject*

Програма має досить приємний дизайн. В першому вікні нас чекає 4 пункти меню, з яких ми можемо потрапити на інші активності.

Додаток дає змогу створити нову гру, в якій вона дає широкий діапазон налаштувань. Вікно налаштувань наведено на мал.. 2.2.



*Мал. 2.2 Вигляд вікна налаштувань*

Цей додаток мітить версію типу PRO, яка працюватиме без реклами. Невеликим мінусом цієї програми є реклама, яка переслідує користувача на кожному кроці.

Наступним етапом після налаштування є ігрове поле з картами. Кожен учасник повинен випадковим чином обрати одну з карт на столі. Після цього для гравця відобразиться його роль у поточній грі. Карта пропадає з ігрового поля, а для ведучого програма зберігає інформацію про кожного гравця і роль, яку він обрав. Вигляд ігрового поля наведено на мал. 2.3.



*Мал. 2.3 Вигляд ігрового поля*

Як було сказано раніше, при натисканні на будь яку карту для користувача відображається випадковим чином його роль на гру. Це можна побачити на мал. 2.4.

 **

*Мал. 2.4 Відображення ролі для користувача (Громадянин)*

Після відображення ролей всім користувачам додаток буде потрібний лише для ведучого, який матиме певний функціонал. Для користувачів цей додаток вже зробив все, що міг, оскільки він не лише допоміг вибрати їм ролі. А для ведучого залишились збережена вся інформація про кожного гравця. Вигляд списку гравців наведено на *мал.* 2.5.



*Мал. 2.5 Відображення списку гравців*

У цьому вікні ведучий може проводити власні замітки, стежити за часом, виставляти на голосування і виганяти гравців з гри. Але це все лише для ведучого.

Розглянувши це програмне рішення я можу сказати наступне: чудово, що існує такий додаток. Хоча б одну проблему він вирішує - це контроль з боку ведучого. Але якраз всі основні проблеми залишились без розгляду, тому потреба в написанні більш розгорнутого додатку є дуже гостра.

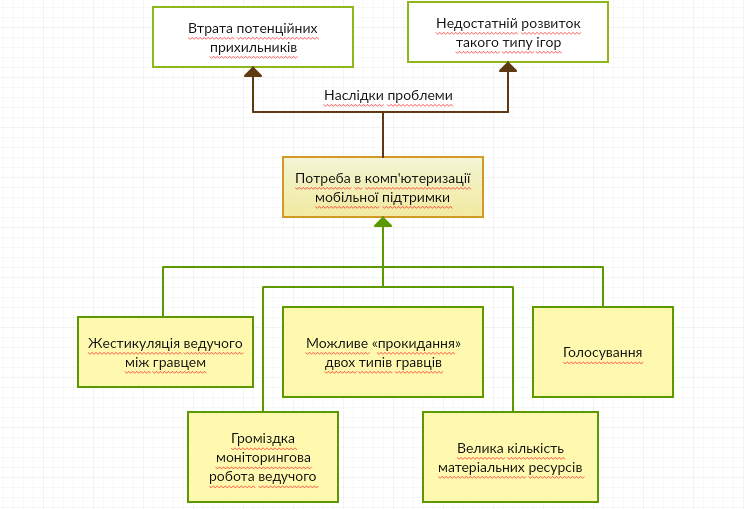
# РОЗДІЛ 3. СИСТЕМНИЙ АНАЛІЗ

## 3.1 Побудова дерева проблеми та деревацілей

### 3.1.1 Побудова дерева проблем

В процесі виконання цієї роботи необхідно було розглянути предметну область дослідження і чітко визначити проблему, яку треба було вирішити створенням програмного продукту. Для того, щоб детальніше описати проблему і проаналізувати її, я побудував дерево проблеми, що відображає основну проблему, яку необхідно було розв’язати, причини її появи і наслідки, котрі вона спричиняє.

В загальному вигляді дерево проблеми можна представити наступним чином мал. 3.1:



Мал. 3.1 Дерево проблеми

Із попереднього малюнку видно, що в корені дерева проблеми лежить наступна проблема – потреба в комп’ютеризації мобільної підтримки.

В процесі побудови дерева проблеми я виділив такі основні причини, які приводять до складності проведення цієї гри:

* Жестикуляція ведучого між гравцем – ця причина є явною. Вона відслідковується у всіх іграх, оскільки ведучий не може явно сказати і спілкуватись з гравцем в певні моменти гри, і це призведе до надмірної жестикуляції між цими особами, яка, рано чи пізно, може бути розкрита, якщо гравець випадковим чином доторкнеться до сусіда, відповідно його роль на гру буде відкрита, що призведе до нецікавості подальшого розвитку подій за столом.
* Громіздка моніторингова робота ведучого – ведучий, який проводить гру, повинен проводити безліч дій для відслідковування подій за столом, він повинен зафіксовувати всі ролі, які належать всім гравцям. З самого початку гри він повинен скласти колоду з відповідних ролей, які повинні підлягати балансу. Також повинен робити відмітки про фоли, гравців, які були виставлені на голосування, гравців, які пропускають свою промову. Моніторинг часу промови і т.д.
* Голосування – дужа нелегким завданням є підрахунок піднятих рук під час голосування, оскільки в один момент часу ми можемо мати велику кількість кандидатів на «виселення з міста».
* Можливість прокидання двох типів гравців – ця ситуація досить часто виникає через неуважність гравців. Ця ситуація призводить до фатальних наслідків. Гра після такої ситуації припиняється і починається заново, це дуже неприємне явище для цієї гри.
* Велика кількість матеріальних ресурсів – аби забезпечити процес проведення кожної групової рольової гри нам потрібна велика кількість матеріальних ресурсів, а саме: велике приміщення, стереосистема, набір пов’язок на очі, відповідний набір ігрових карт, які, в свою чергу, відповідають за ролі гравцям.

Названі вище проблеми призводять до таких наслідків:

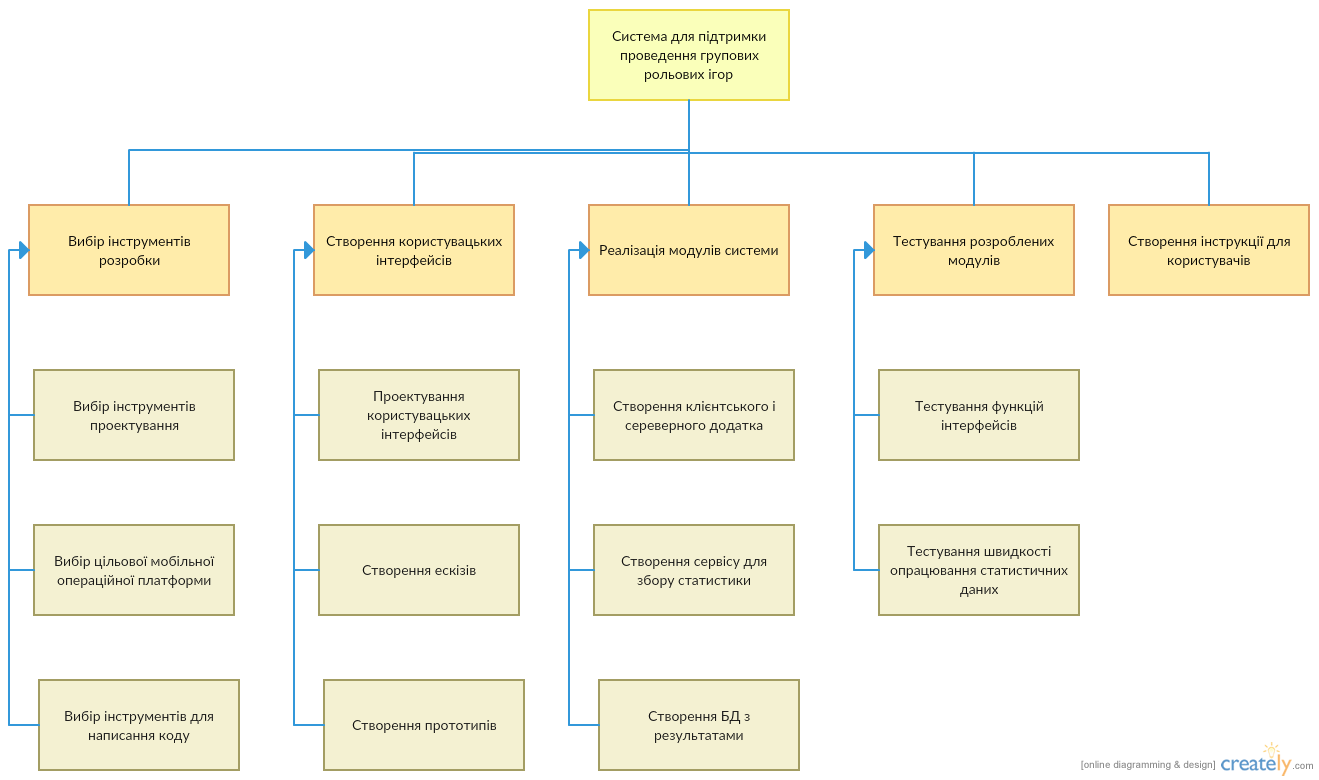
* Втрата потенційних користувачів – з першого разу привернути увагу до цієї гри є дуже важко, оскільки потрібно забезпечити максимальну якість під час проведення гри. І якщо в нас немає відповідних засобів для цього, то гравцям, які тільки починають грати у цю гру, буде нецікаво грати у неї наступного разу.
* Недостатній розвиток такого типу ігор – якщо проблеми залишаються і надалі, і щоразу більше гравців відмовляються грати в неї, то сама гра перестане існувати. І розвиток таких ігор буде тільки реальним у досить великих і відомих закладах і в турнірах, але між звичайними любителями вона стане нецікавою і недоступною.

Проаналізувавши усі причини поточної проблеми і наслідки, які вона спричиняє, можемо зробити висновок про те, що сфера рольових ігор потребує системи, яка зможе вирішити вказані вище проблеми та задачі.

### 3.1.2 Побудова дерева цілей

Для того, щоб продовжити системний аналіз, необхідно визначити і поставити перед собою чіткі цілі, які потрібно буде досягнути в процесі розробки системи. Це потрібно зробити для систематизації роботи над системою і можливості для планування роботи.

В процесі побудови дерева цілей, було створене дерево, наведене на мал.3.2



Мал.3.2 Дерево цілей

В корені дерева лежить головна ціль, яку треба досягнути в процесі роботи, – створення системи для підтримки проведення групових рольових ігор.

Для того, щоб досягнути дану ціль, було проведено декомпозицію цієї цілі на дрібніші. На першому рівні(стратегічному) було визначено 5 підцілей – вибір інструментів розробки, створення користувацьких інтерфейсів, реалізація модулів системи, тестування модулів системи і створення інструкцій для користувачів.

**Вибір інструментів розробки передбачає:**

* Вибір інструментів проектування;
* Вибір цільової мобільної операційної платформи;
* Вибір інструментів для написання коду.

**Створення користувацьких інтерфейсів передбачає наступні підцілі:**

* Проектування користувацьких інтерфейсів;
* Створення ескізів;
* Створення прототипів.

**Реалізація модулів системи передбачає наступні підцілі:**

* Створення клієнтського і серверного додатка;
* Створення сервісу для збору статистичних даних;
* Створення БД з результатами

**Для тестування клієнтської частини потрібно виконати наступні підцілі:**

* Провести тестування функцій інтерфейсу;
* Тестування швидкодії опрацювання статистичних даних.

## 3.2 Аналіз та вибір методів, алгоритмів та засобів розв’язання задачі

**3.2.1 Статуси в грі та їх обчислення**

Велике значення для групової рольової гри має баланс. Цим значенням ми повинні поставити в рівні умови гравців обох сторін. Нам потрібно поставити обмеження на кількість гравців та вміти збалансувати ролі між гравцями, аби поставити їх в рівні умови [5].

Якщо обчислити це значення невірно, то при недостатній кількості гравців гра стає нецікавою і швидко закінчується, а при великій дуже важко зосередитись на всіх гравців і гра починає ділитись на групи співрозмовників.

Формула (1) для обрахунку кількості ролей мафії:

(1),

де М — кількість ролей мафії, \! N —кількість гравців, \! k —[коефіцієнт](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B5%D1%84%D1%96%D1%86%D1%96%D1%94%D0%BD%D1%82) розрахунку.

* Для гри за одним столом розрахунковий коефіцієнт вибирається з проміжку від 3 до 4.
* В іграх по мережі приймається підвищене значення для k.

### 3.2.1 Клієнт-серверна архітектура

Як правило, комп’ютери і програми, які входять до складу ІС, не є рівноправними, деякі з них володіють ресурсами (файлова система, процесор, БД і тд.), інші мають можливість доступатись до цих ресурсів[6].

Програму чи процесс, який виконує роль управління ресурсами, називають сервером, клієнт і сервер будь-якого ресурсу можуть знаходитись як на одному комп’ютері, так і на різних комп’ютерах, зв’язаних по мережі.

Оскільки в основі лежить архітектура «клієнт - сервер», то будь яка ІС складається з декількох компонентів, кожен з яких реалізовує свій набір певних сервісів. Компоненти такої системи можуть виконуватись на різних комп’ютерах, виконуючи сервісні або клієнтські функції. Це дозволяє збільшити надійність, безпеку і продуктивність мережевих програм загалом.

### 3.2.2 Soket

На сьогоднішній день використання клієнтських служб швидкого обміну повідомленнями (instantmessanger) стали незамінним засобом для всіх користувачів Інтернету. Зараз існує дуже багато різноманітних протоколів і клієнтів (MSN,ICQ,Skype і тд.), які ми кожен день використовуємо. Але ми ніколи не задумувались, що покладено в основу їх функціонування.

Наприклад, ми завантажили і встановили один з цих сервісів обміну повідомленнями, і при підключенні до серверу успішно увійшли у систему. На одному пристрої може паралельно працювати безліч програм, уявімо, що ваш пристрій виконує п’ять різноманітних програм, які очікують на підключення з іншими пристроями по мережі. Оскільки наш пристрій має одну ІР-адресу, то не важко уявити собі, яким чином наші програми отримують повідомлення - саме так, через ІР. Але потрібно розуміти, що одним каналом нам не обійтись. А це означає те, що різні програми повинні мати якийсь додатковий ідентифікатор для стабільної роботи та ведення живого листування.

Такими унікальними номерами кожної програми є порти. І виявляється, що таким шаблоном (ІР-адреса + порт) ми можемо отримати унікальне значення програми в мережі.

В поняттях програмування, для забезпечення комунікації між різними комп’ютерами в мережі, нам будуть потрібні дві різні програми в одній мережі. В одній з них буде створюватись серверний сокет зв’язку, а в іншого - клієнтський. Різниця між ними є дуже простою. Серверний сокет вміє очікувати на підключення клієнтів. При його створенні потрібно вказати конкретно порт, який буде використовуватись для комунікації по мережі. І потрібно не забути про можливі колізії. Що я маю на увазі? Ми не можемо бути впевнені в тому, що наш порт на комп’ютері не використовує інша стороння програма, що призведе до помилки.

Варто нагадати, що для використання цього рішення нам не буде потрібно з’єднання з Інтернетом. Оскільки ми хочемо проводити групову рольову гру незалежно до місця, часу і можливості підключення до мережі. Але результати будуть зберігатись на віддаленому сервері, з якого ми зможемо отримати результати через ASMX-сервіс.

**3.2.3 Вибір мови програмування**

В другому розділі я розглянув основні мови програмування, які використовуються для розробки мобільних платформ. Я обрав мову програмування Java, за допомогою якої ми маємо можливість побудувати мобільні додатки, які будуть доступні для 70% мобільних пристроїв у світі[8].

Одна з основних переваг мови Java - незалежність від платформи, на якій виконуються програма, той самий код можна скомпілювати і запускати під управлінням різноманітних операційних систем, таких як: Windows, Solaris, Linux, Machintosh та ін.

Синтаксис мови Java схожий до інших мов програмування, що не скаладє труднощів в її вивченні.

Крім того, Java - повністю об'єктно-орієнтована мова, навіть більшою мірою, ніж C++. Всі сутності в мові Java є об'єктами, за винятком небагатьох основних типів (primitive types).

Творці мови виділяють такі особливості:

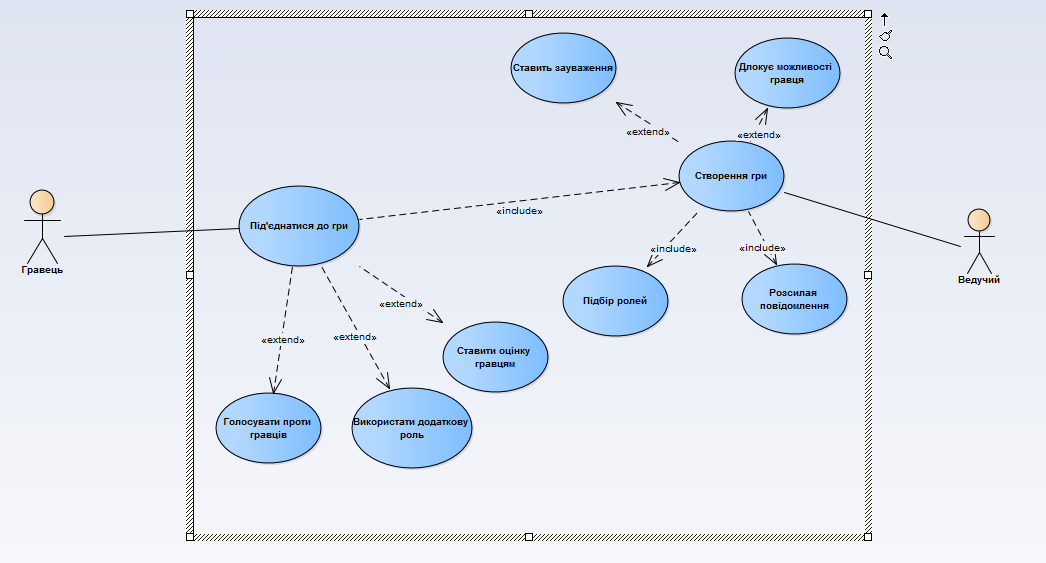
- висока надійність. Творці забезпечили мову Java засобами, що дозволяють виключити саму можливість створювати програми, в яких були б приховані найбільш поширені помилки. Для цього в мові Java зроблено наступне:

- виключена можливість явного виділення і звільнення пам'яті. Програмістам більше не потрібно думати про пам'ять, всю роботу за них виконає збірник сміття, який завжди відслідковує пам'ять і очищає дані у втрачених посиланнях.

- введені істинні масиви і заборонені вказівники. Тепер програміст забезпечений від стирання даних, використовуючи вказівники.

**3.2.4 Побудова Use Case-діаграми**

На мал.3.3 зображена діаграма прецедентів для нашої системи, яка відображає можливі випадки використання системи користувачами. Нижче, під діаграмою прецедентів, в таблиці наведено специфікацію прецедентів.



Мал.3.3.UseCase діаграма

Таблиця 3.1 Специфікація прецеденту «Створення гри»

|  |  |
| --- | --- |
| *Прецедент* | *Створення гри* |
| Короткий опис | Прецедент дає можливість ведучому створити гру |
| Суб'єкти | Ведучий, Система |
| Передумови | Ведучий налаштовує вимоги до гри |
| Основний потік | Початок прецеденту – ведучий обирає налаштування стосовно гри. Система пропонує декілька форм для введення інформації про гру, тобто її налаштування. |
| Альтернативний потік | Ведучий може отримати помилку налаштування і отримає інструкцію по її виправленню. |
| Постумови | Якщо прецедент був успішний, то в результаті система видає інформацію, згідно введених ведучим даних. Також повідомляє його про успішність з’єднання. |

*Таблиця 3.2 Специфікація прецеденту «Блокує можливості гравця»*

|  |  |
| --- | --- |
| *Прецедент* | *Блокує можливості гравця* |
| Короткий опис | Забороняє певні можливості гравцям. |
| Суб'єкти | Ведучий, Система |
| Передумови | Ведучий обирає гравця, якого хоче заблокувати. |
| Основний потік | Початок прецеденту – кінець процедури - заблокований гравець, який не має права обговорення. Прецедент дозволяє додавати та видаляти блокованих гравців із списку. |
| Альтернативний потік | З тих чи інших причин система повідомляє ведучому про помилку і пропонує повторити спробу. |
| Постумови | Якщо прецедент був успішний, система зберігає список блокованих користувачів |

*Таблиця 3.3 Специфікація прецеденту Робити замовлення*

|  |  |
| --- | --- |
| *Прецедент* | *Робити замовлення* |
| Короткий опис | Прецедент дає можливість ведучому робити замовлення. |
| Суб'єкти | Ведучий, Система, Гравець. |
| Передумови | Ведучий обирає гравця і ставить йому штрафний бал. |
| Основний потік | Прецедент настає після обирання гравця, якому буде врахований фол. |
| Альтернативний потік | З тих чи інших причин система повідомляє клієнту про помилку і пропонує повторити спробу. |
| Постумови | Якщо прецедент був успішний, система повідомляє ведучого. |

*Таблиця 3.4 Специфікація прецеденту «Підбір ролей»*

|  |  |
| --- | --- |
| *Прецедент* | *Підбір ролей* |
| Короткий опис | Прецедент дає можливість ведучому скорегувати дані стосовно ролей. |
| Суб'єкти | Ведучий, Система. |
| Передумови | Ведучий відібрав ролі |
| Основний потік | Початок прецеденту співпадає з рішенням ведучого стосовно ролей. |
| Альтернативний потік | З тих чи інших причин система повідомляє клієнту про помилку і пропонує повторити спробу. |
| Постумови | Якщо прецедент був успішний, сформовується база даних. В іншому випадку стан системи залишається незмінним. |

*Таблиця 3.5 Специфікація прецеденту «Під’єднання до гри»*

|  |  |
| --- | --- |
| *Прецедент* | *Під’єднання до гри* |
| Короткий опис | Прецедент дає можливість гравцю під’єднатись до гри |
| Суб'єкти | Гравець, Система |
| Передумови | Гравець вводить вхідні дані. |
| Основний потік | Прецедент надає змогу під’єднатися до гри, яку створив ведучий. |
| Альтернативний потік | З тих чи інших причин система повідомляє клієнту про помилку і пропонує повторити спробу. |
| Постумови | Якщо прецедент був успішний, система додає гравця в гру. |

**3.2.5 Вибір серверного рішення**

В будь-якому випадку вибір типу бази даних буде базуватись на трьох основних принципах, які зображені на мал.3.4, а саме:

* Узгодженність (Consistensy)
* Доступність (Availability;)
* Композиційність (Partitiontolerance).



*Мал. 3.4 CAP-діаграма*

На жаль, обрати такий тип БД, який би задовольняв ці всі 3 принципи, неможливо. А це означає те, що чимось ми повинні пожертвувати. Опираючись на своє розуміння БД та визначивши потреби, я вирішив обрати принцип, яким керуються реляційні БД, а це узгодженість і доступність.

Я обираю комерційний варіант СКБД Microsoft SQL Server. Мова, що використовується для запитів — Transact-SQL, створена спільно Microsoftта Sybase. Transact – SQL є реалізацією стандарту ANSI/ISO що до структурованої мови запитів (SQL) і з розширеннями. Використовується як для невеликих і середніх за розміром баз даних, так і для великих баз даних масштабу підприємства. Багато років вдало конкурує з іншими системами керування базами даних.

# РОЗДІЛ 4. РОЗРОБКА ТА ВІДОБРАЖЕННЯ АЛГОРИТМІВ РОБОТИ КОМПОНЕНТІВ ТА СИСТЕМИ

## 4.1 Побудова моделей бізнес-процесів за допомогою діаграм IDEF0

При проектуванні нашої системи я використовував структурний підхід. Суть цього підходу до розробки інформаційних систем полягає в тому, що її розділяють на функції: система розбивається на функціональні підсистеми, які в свою чергу діляться на дрібніші функції, що підрозділяються на завдання і так далі. При цьому автоматизована система зберігає цілісне представлення, у якому всі складові компоненти взаємопов'язані. Процес розбиття триває аж до конкретних процедур. При зворотній розробці системи "знизу-вверх" – від окремих завдань до всієї системи – цілісність втрачається, виникають проблеми інформаційного стикування окремих компонентів.

В процесі проектування ми використовували наступні види діаграм:

* IDEF0 - методологія функціонального моделювання і графічного опису процесів, призначена для формалізації і опису бізнес-процесів. Особливістю IDEF0 є її акцент на ієрархічне представлення об'єктів, що значно полегшує розуміння предметної області. В IDEF0 розглядаються логічні зв'язки між роботами, а не послідовність їх виконання в часі;
* DFD - модель проектування, графічне представлення «потоків» даних в інформаційній системі. Діаграма потоків даних також може використовуватись для візуалізації процесів обробки даних;

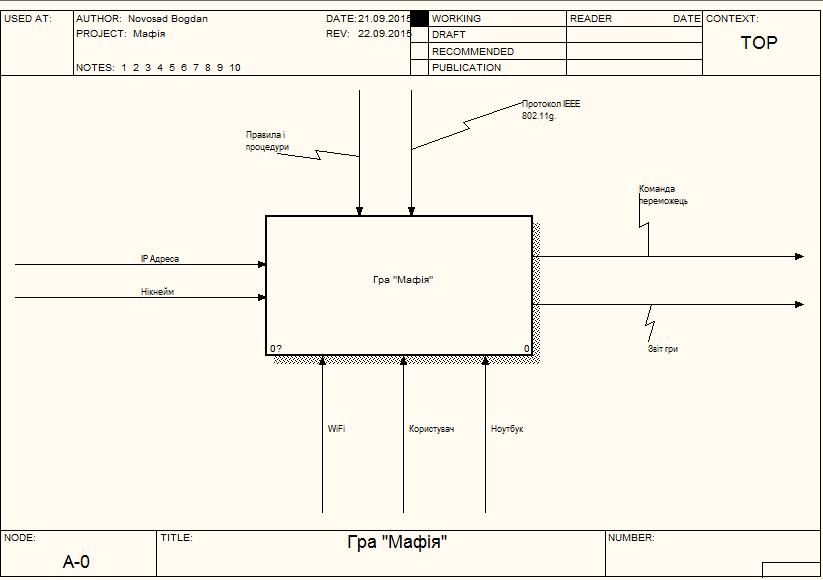
При розробці ми використовували спіральну модель, оскільки в проектуванні і розробці цієї системи є важливим швидкий результат, і важливо, щоб були відповідні прототипи.

Також при проектуванні цієї системи було використано засоби UML. UML – уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, називаної UML-моделлю. UML була створена для визначення, візуалізації, проектування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива генерація виконавчого коду.

При проектуванні системи я використовував AllFusion Process Modeler 7.

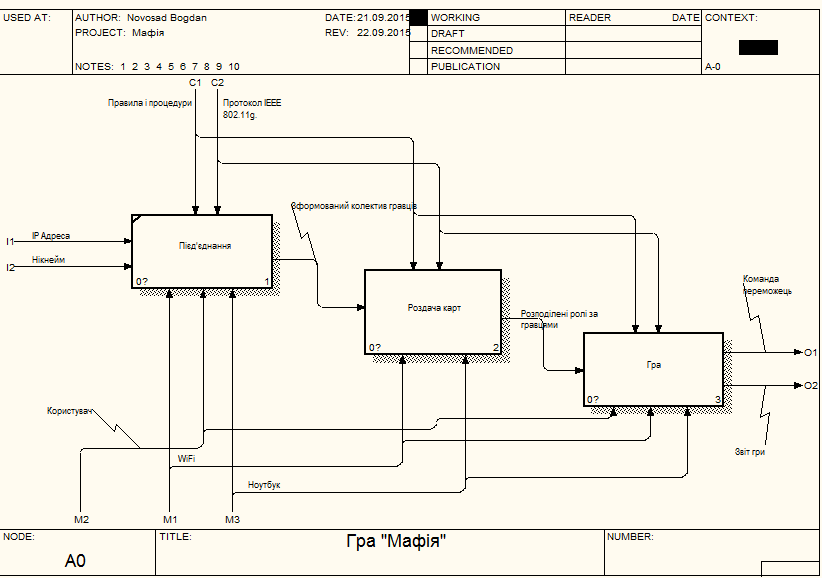
AllFusion Process Modeler 7 (раніше BPwin) - інструмент для моделювання, аналізу, документування та оптимізації бізнес-процесів. Його можна використовувати для графічного представлення бізнес-процесів.

Для функціонального моделювання і створення моделей основних бізнес-процесів використано IDEF0. На мал.4.1 зображена контекстна діаграма основного бізнес-процесу роботи нашої системи. Ця діаграма описує основний вид діяльності проектованої системи.

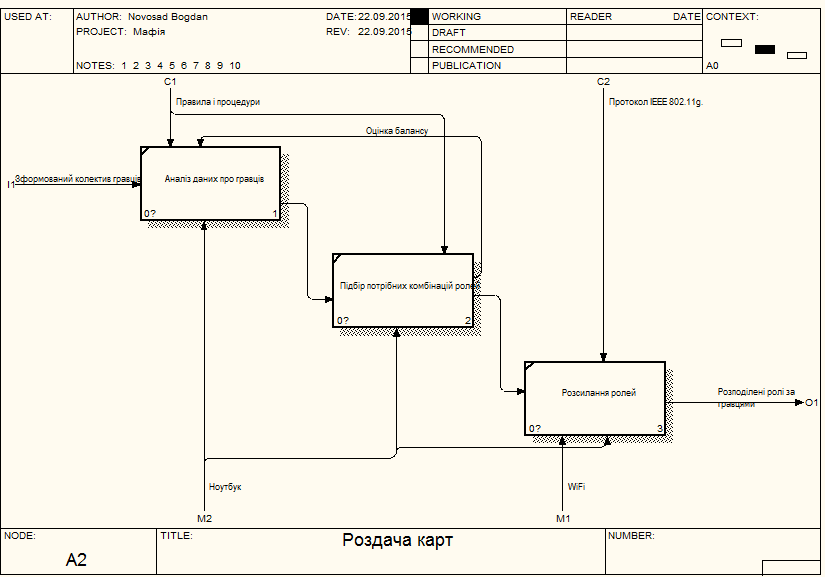


Мал.4.1Контекстна діаграма

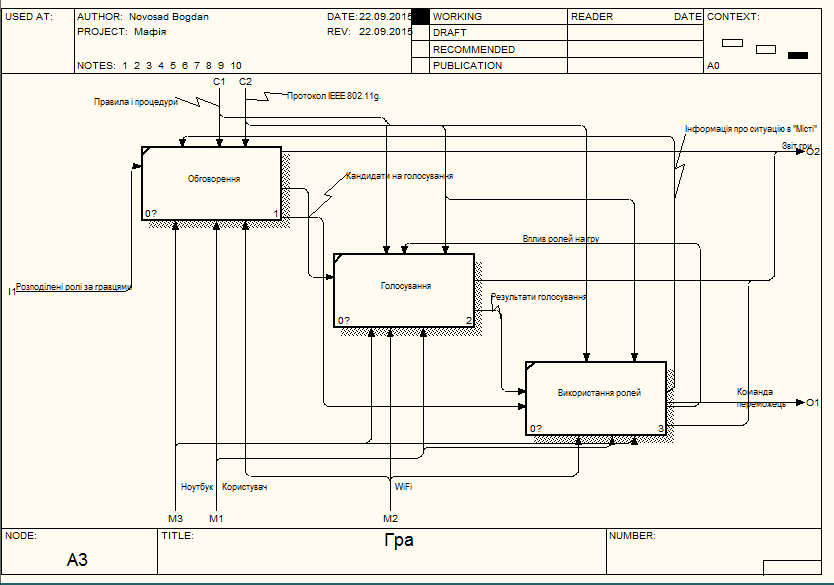
Наступні діаграми, котрі зображені на мал.4.2-мал.4.4, відображають декомпозицію контекстної діаграми і декомпозицію першого рівня «Роздача карт» та «Гра», відповідно.



*Мал. 4.2 Діаграма декомпозиції контекстної діаграми*



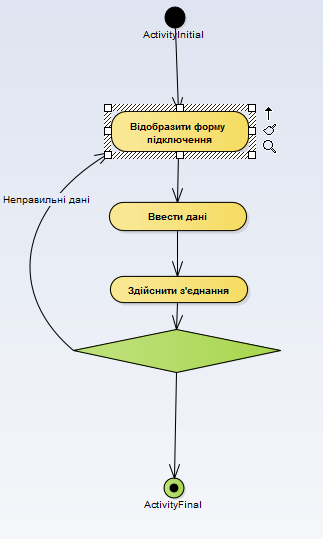
*Мал. 4.3 Діаграма декомпозиції першого рівня (для роботи Роздача карт)*



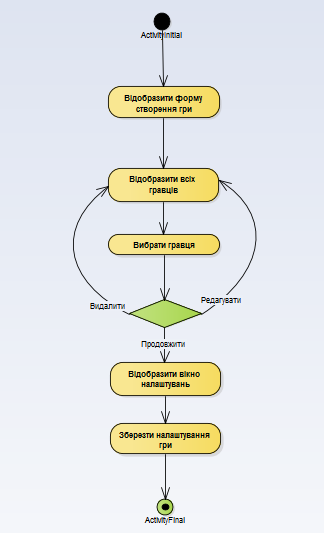
*Мал. 4.4 Діаграма декомпозиції другого рівня (для роботи Гра**)*

## 4.3 Моделювання видів діяльності

Моделювання видів діяльності (activity model) представляє в графічній формі потік подій для певного прецеденту. На наступних малюнках зображено діаграми видів діяльності – діаграма діяльності «Підключення гравця» (мал.4.5), діаграма створення гри та редагування налаштувань (мал.4.6).



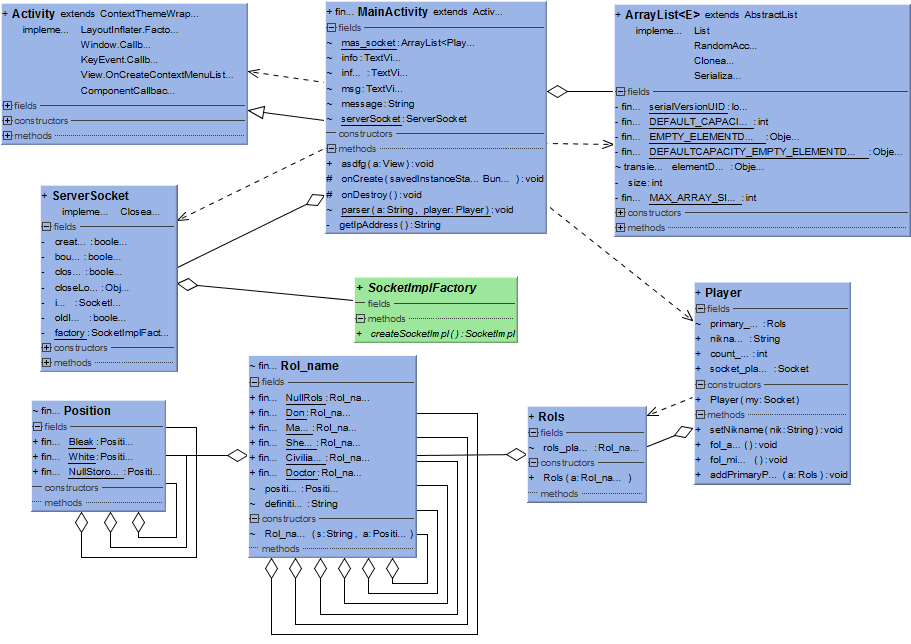
Мал.4.5 Діаграма підключення гравця



Мал. 4.6 Діаграма створення гри та редагування налаштувань

## 4.4 Діаграма класів

На рис.4.7 зображено діаграму класів системи, на якій показані основні класи які присутні в системі, методи роботи із ними і їхні атрибути.



Мал.4.7 Діаграма класів системи

**MainActivity** – Клас, наслідуваний від Activity, для відображення та зв’язку Ведучого або ж гравця з програмою.

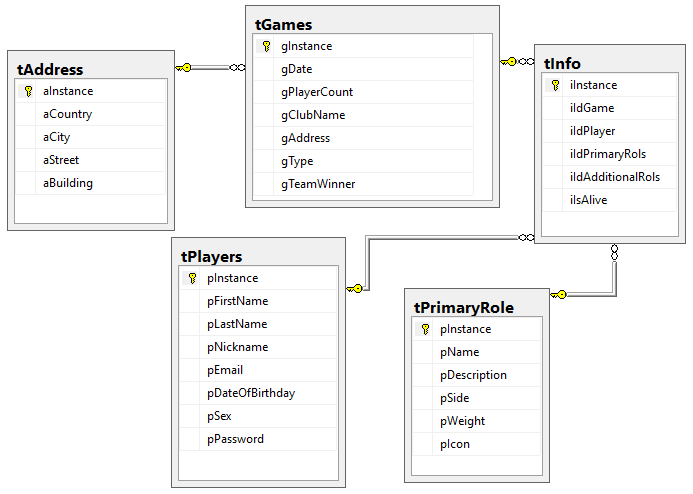
**Players -** Цей клас містить всю інформацію про гравця, його нікнейм, адрес, методи, які встановлюють значення ролі та активного підключення

**ServerSocket** - Цей клас реалізує серверні сокети. ServerSocket очікує запитів по мережі. Він виконує деякі операції на основі цього запиту, а потім повертає результат в запитуючої сторони.

**Rols -** Клас, який має набір ролей для гри, які вносяться як поле до кожного гравця

**4.5 Cхема бази даних**

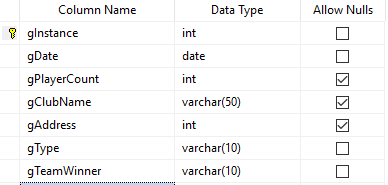
На рис.4.8 наведено логічну схему БД. Схема складається з 5 таблиць, пов’язаними зв’язками один до багатьох та багато до багато. Для реалізації зв’язку багато до багатьох було додано додаткову таблицю tInfo.



*Рис.4.8 Логічна схема БД*

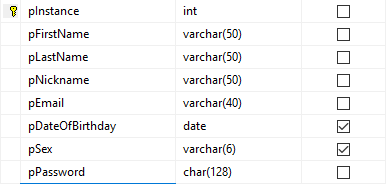
Таблиця tGames описує інформацію про одиницю гри, а точніше всю інформацію про час проведення, місце, команду переможців, тип гри.

Фізична структура таблиці наведена на рис.4.9.



*Рис.4.9 Фізична схема таблиці* tGames

Таблиця tPlayers містить інформацію про всіх гравців системи. Поле паролю зашифроване за допомогою методу SHA-512 - найнадійнішого шифрування на сьогоднішній день, яке доступне для використання скриптом TransactSQL. Фізична структура таблиці наведена на мал.4.10.



*Мал.4.10 Фізична схема таблиці* tPlayers

# РОЗДІЛ 5. РОЗРОБКА ПРОГРАМНОГО РІШЕННЯ

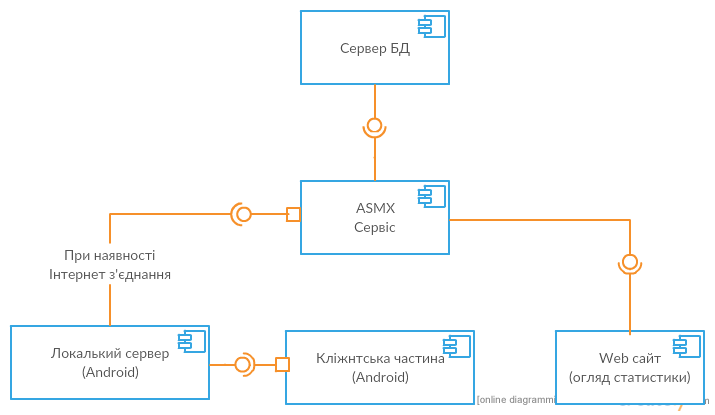
## 5.1 Загальна структура програмного проекту (діаграма компонентів)

Розроблена система складається з 5 компонентів. На малс. 5.1 зображено взаємозв’язки між ними.

Компоненти Локальний сервер та Клієнтська частина можуть існувати без інших і виконувати аматорську роль проведення групової рольової гри без збереження результатів проведення.

Також компоненти Сервер БД та ASMXсервіс працюють незалежно від інших компонентів, що робить нашу систему дуже гнучкою. Вони надають великий ряд послуг для збору та відображення статистичних даних.

Компонент Webсайт є компонентом презентаційного рівня, і це тільки один з прикладів. Насправді таких презентаційних компонентів може бути безмежна кількість. Сервіс надає інтерфейс для отримання статистичних даних, а це означає, що будь-які програми різних типів і написані на різних мовах зможуть отримати результат в зручному для відображення вигляді.



Мал. 5.1 Діаграма компонентів системи

## 5.2 Опис використаних сторонніх бібліотек та модулів

### 5.2.1 Web-сервіс

Для реалізації веб-сайту для огляду статистики я використовував наступні технології та мови програмування:

* HTML5 – найновіша ревізія мови гіпертекстової розмітки, яка є, де-факто, стандартом індустрії;
* CSS3 – найновіша версія мови опису каскадних таблиць стилів;
* JavaScript – об’єктно-орієнтована (прототипно-орієнтована) мова програмування, яка найчастіше використовується для створення програмних конструкцій на клієнтській стороні;
* jQuery – популярна JavaScript-бібліотека із відкритим вихідним кодом, яка значною мірою полегшує розробку клієнтських програм за допомогою мови JavaScript;
* LESS – препроцесор CSS, надбудова над стандартом мови CSS, додає логічні структури і програмні конструкції до мови CSS;

Також під час розробки цієї системи було використано такі інструменти:

* VisualStudioCode – засіб для створення, редагування та зневаження сучасних веб-застосунків і програм для хмарних систем. Visual Studio Code розповсюджується безкоштовно і доступний у версіях для платформ Windows, LinuxіOSX.
* Google Chrome – браузер від компанії Google. Використовувався для перевірки і тестування створених користувацьких інтерфейсів;
* GoogleChromeDevTools – набір інструментів, які надають великі можливості для аналізу HTML-документів, файлів із стилями CSSі клієнтських програм на мові JavaScript;
* GitHub – децентралізована система контролю версій, яка використовувалась мною для оптимізації роботи з кодом;
* NPM - менеджер пакетів для середовища виконання JavaScriptNode.js.
* **5.2.2 База даних та ASMX-сервіс**

Цей модуль розроблявся для збору інформації та ведення статистики. Він надає послуги для локальної серверної програми на мобільній операційній платформі Android для приймання результатів ігор і сайту для перегляду статистики гравців. Для розробки цього модуля я використовував наступні мови та технології.

* SQLServerManagementStudio (SSMS) - утиліта з MicrosoftSQLServer 2005 і пізніших версій для конфігурації, управління і адміністрування всіх компонентів MicrosoftSQLServer. Утиліта включає скриптовий редактор і графічну програму, яка працює з об'єктами і налаштуванням сервера.
* ASP.NET 5.0 – бібліотека, яка призначена для полегшення створення веб-сайтів, частина екосистеми .NET;
* MicrosoftVisualStudio 2015 – це популярне середовище програмування від Microsoft. Містить у собі багато корисного функціоналу, зокрема підтримує велику кількість мов програмування. Можливості цього програмного пакету можна розширити за допомогою додатків, доступних на веб-сайті цього інструменту;
* ReSharper 10.1 – додаток до MicrosoftVisualStudio 2015, який значно оптимізує і полегшує написання програм на мові C#;

### 5.2.3 Клієнтський та серверний(локальний) додаток

Компонент для проведення групової рольової гри «Мафія» складається з двох частин, а саме:

1. Клієнтська частина (Гравець);
2. Серверна частина (Ведучий).

Ці додатки розроблені для мобільної платформи Android, вони будуть основними інструментами для проведення гри. У процесі реалізації цього модуля я використовував такі мови та технології:

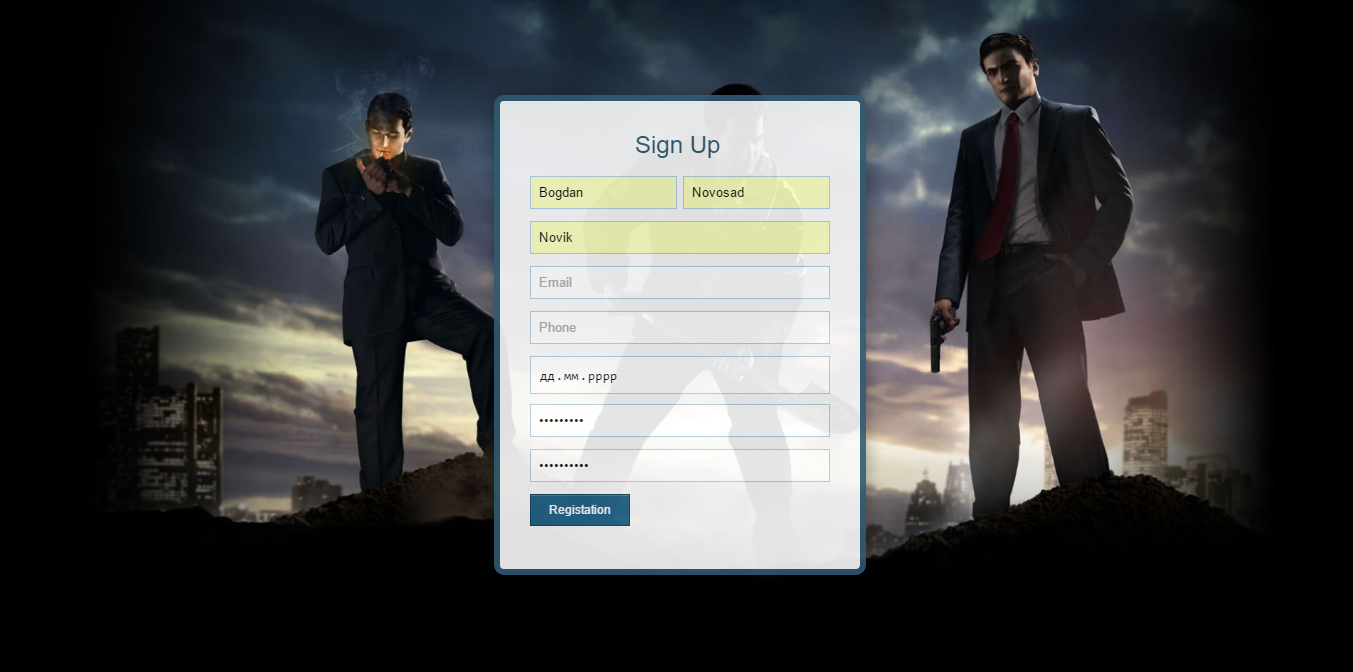
* Java –об’єктно-орієнтована мова, яка є основним компонентом платформи Java. Також вона лежить в основі операційної мобільної платформи Android;
* AndroidStudio 2.0 – середовище розробки для платформи Android;
* Genimotion – функціональний емулятор Androidдля PC.

## 5.3 Розробка та опис інтерфейсу користувача

### 5.3.1 Розробка веб сервісу

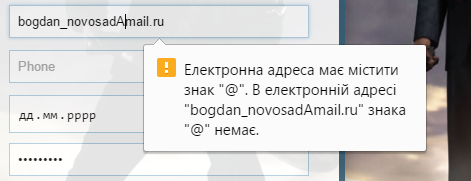
Для виконання веб-рішення цієї роботи було розроблено 3 HTML документи, з яких складався модуль огляду статистики. Були написані всі сценарії на мові JavaScript, і написано 90 рядків стилів.

На мал.5.2зображено сторінку реєстрації у системі. Як бачимо, користувач повинен заповнити всі поля для вдалої реєстрації в системі.



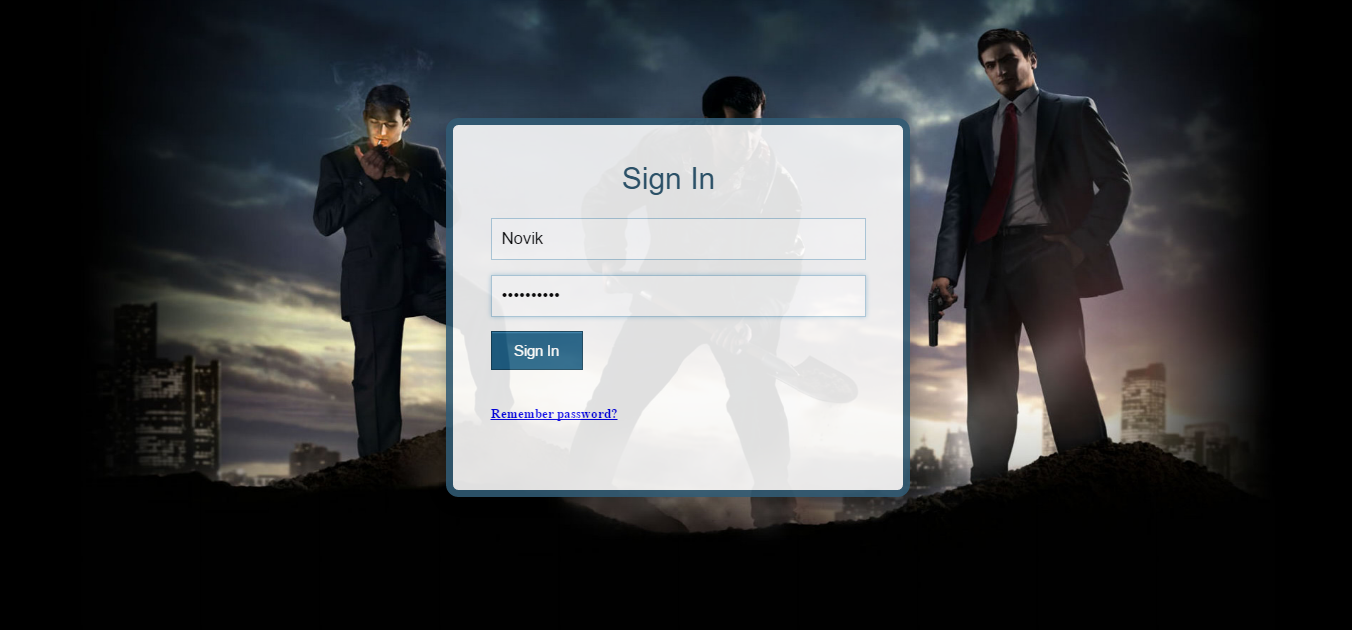
*Мал. 5.2 Сторінка реєстрації у системі*

Сторінка автоматично валідує ці поля і, у разі помилки, видає відповідне повідомлення. Приклад валідації введеної інформації в полі електронної скриньки наведене на мал.5.3.



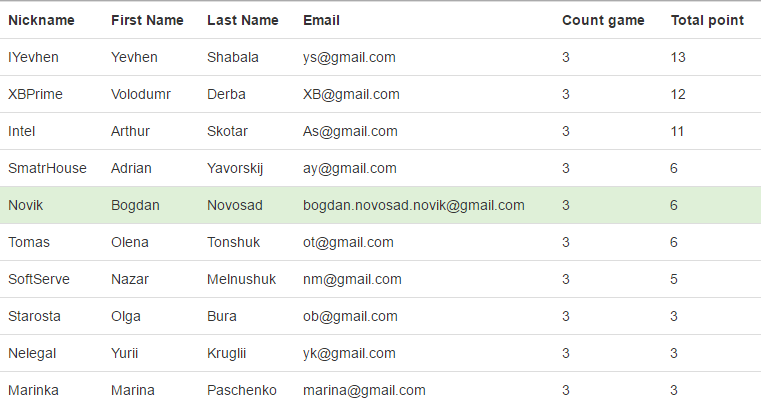
*Мал. 5.3 Валідація введеної інформації*

На мал. 5.4 зображено форму реєстрації, в якій потрібно ввести дані: логін та пароль, також форма надає можливість нагадати пароль користувачу.



*Мал. 5.4 Сторінка входу у систему*

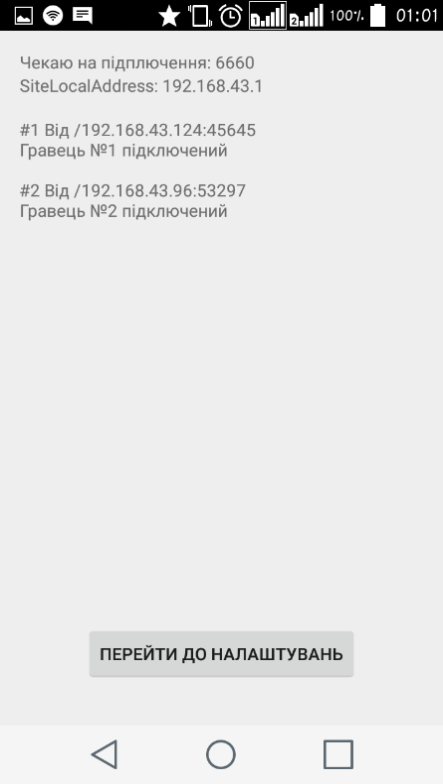
Після входу в систему користувачу відображається форма, в якій відображена статистика всіх гравців, а статистика гравця, який авторизувався, відображена іншим стилем - зеленим мал.5.5.



*Мал. 5.5 Сторінка статистики*

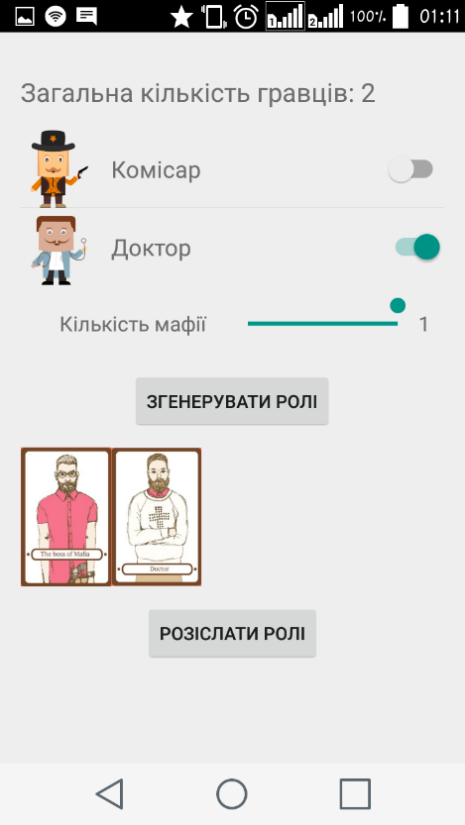
### 5.3.2 Інтерфейси мобільних додатків

Для реалізації локальної мережі з можливістю спілкування розроблено два додатки на платформі Android. Розглянемо інтерфейс серерного додатка. Ведучий запускає додаток і отримує інформацію для з’єднання, в подальшому у цьому вікні буде оновлятись інформація про гравців, які підключились до гри. Форма відображення цієї інформації зображена на мал. 5.6.



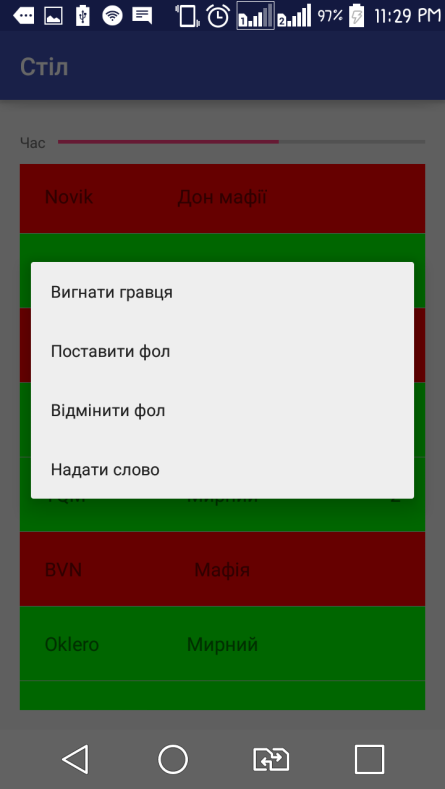
*Мал. 5.6 Підключення гравців*

Наступнийм крок - створення активності, в якій ведучий проводить налаштування і вибір ролей. Форма налаштування наведена на мал.5.7



*Мал. 5.7 Підбір ролей*

У ведучого буде присутня активність, в якій він буде мати можливість вибору різноманітних дій, а саме: вигнати гравця, поставити фол, відмінити фол, надати слово. Вигляд активності наведено на мал. 5.8.



*Мал. 5.8 Відображення панелі керування грою*

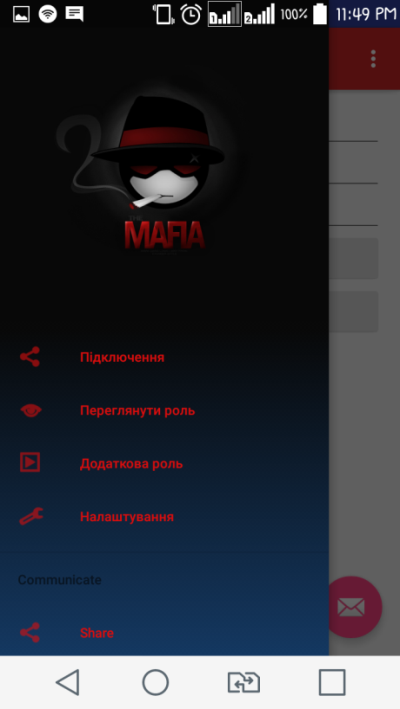
На стороні ведучого присутня форма з’єднання з сервером ведучого. У цій формі користувач вводить дані для з’єднання і підєднується до серверної частини.

Вікно підключення наведено на мал. 5.9



*Мал. 5.9 Вікно підключення*

Клієнтська частина програми має головне меню навігації мал. 5.10, при якому користувач може здійснювати навігацію по програмі.



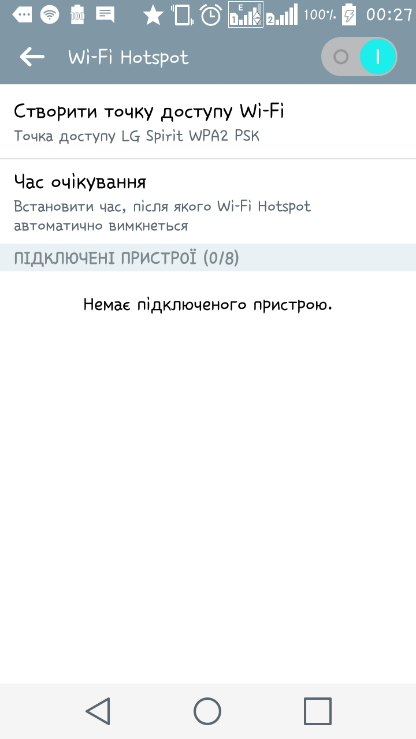
*Мал. 5.10 Основне меню*

Програмний код додатків наведено в додатках А, В та С.

# Розділ 6. ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА

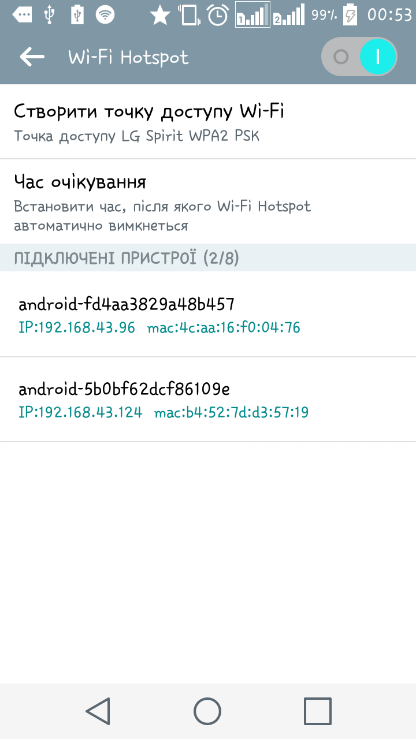
## 6.1 Інструкції для користувачів

Для того, аби розпочати використання наших додатків, потрібно створити локальну мережу. Це можна виконати декількома способами. Одним з таких способів є створеня точки доступу безпосередньо з мобільного пристрою за допомогою функції Хостпот. Створення точки доступу наведено на мал. 6.1.



*Мал. 6.1 Створення точки доступу*

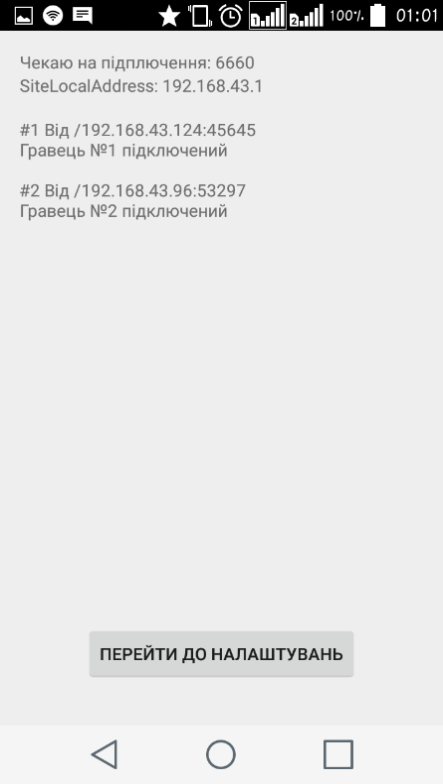
Тепер умовою підключення гравців є те, що вони повинні приєднатись до цієї мережі. І ввести коректні дані для підключення. На мал.6.2 наведено два під’єднаних пристрої.



*Мал. 6.2 Підключені пристроїв*

Як бачимо, під’єднані пристрої відображуються в відповідній панелі. Для кожного мобільного пристрою резервується своя адреса, за якою можуть комуні кувати пристрої.

Після того як всі користувачі системи будуть знаходитись в одній мережі ведучий може переходити до етапу створення столу в якому він повинен буде створити однорангову мережу яка складатиметься тільки з гравців які братимуть участь у грі. Ведучому на ерані виведе інформацію про підключення, це інформація про ІР адресу та порт рис.6.3.



*Рис.6.3 Інформація про налаштування зєднання*

Тепер користувачі зможуть отримати інформацію для того аби під’єднатись до столу і розпочати гру.

## 6.2 Інструкції адміністратору

Windows Server 2012, під кодовою назвою Windows Server 8, це шостий випуск Windows Server. Дві попередні версії і бета-версія, були випущені в процесі розробки. Програмне забезпечення в цілому доступні для клієнтів, починаючи з 4 вересня 2012 року.

Перед тим, як почати інсталювання сервера потрібно переконатись, що ваша машина відповідає мінімальним вимогам для встановлення (вимоги Windows сервера наведені в пункті 6.3) Windows Server.

Вставте Windows Server 2012 DVD диск, і як тільки ви отримаєте повідомлення яке зображено на рис.6.4, натисніть Enter для завантаження з диску

**C:\Users\Bogdan\Downloads\8836.1.png**

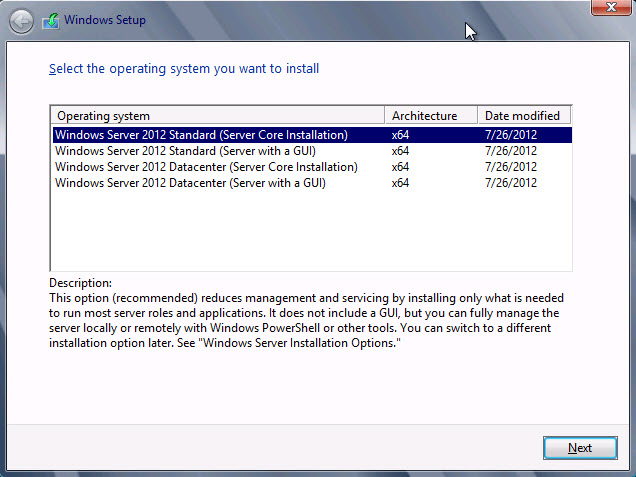
*Рис. 6.4 Підтвердження завантаження образу диску*

Почекайте деякий час, поки програма установки скопіює всі необхідні файли (в залежності від вашої машини, це займе кілька хвилин).Після того, як будуть завантажені файли, установка почнеться з екрану зображеного на рис.6.5. Ви можете змінити їх, щоб задовольнити ваші потреби.

****

*Рис.6.5 Вікно мовних налаштувань*

Ви побачите екран з загрузкою, почекайте, поки він закінчить завантаження. У наступному екрані зображеному на рис.6.7 настройки, ви побачите чотири варіанти. Виберіть Windows Server 2012 DataCenter Evaluation (Server With GUI).

****

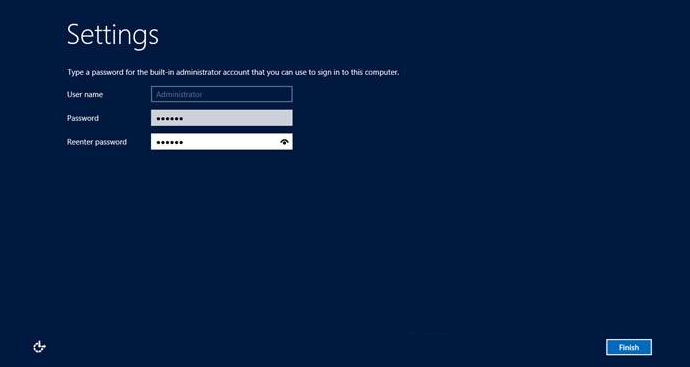
*Рис.6.7 Вибір операційної системи*

Після натискання кнопки Next від попереднього екрана, прочитайте умови ліцензійної угоди, відзначте: " I accept the license terms " і натисніть кнопку Next. Тепер програма інсталяції проситиме вас вибрати диск на який потрібно буде встановити дане ПЗ рис.6.8.

****

*Рис.6.8 Вибір розділу диску*

Тепер, як тільки ми вибрали наш розділ, натиснувши на наступний з попереднього екрану то програма почне інсталяцію. Цей процес може зайняти деякий час. Після того як інсталяція буде завершена, сервер буде перезавантажений і запустить Windows Server 2012 в перший раз. Він попросить вас встановити пароль для користувача Administrator рис.6.9.

****

*Рис.6.9 Встановлення паролю для адміністратора*

Установка завершить налаштування, це може зайняти кілька хвилин. Після цього нам потрібно скрипт схеми БД. Ми повинні запустити його на нашому сервері для використання сервера.

## 6.3 Вимоги до апаратного забезпечення

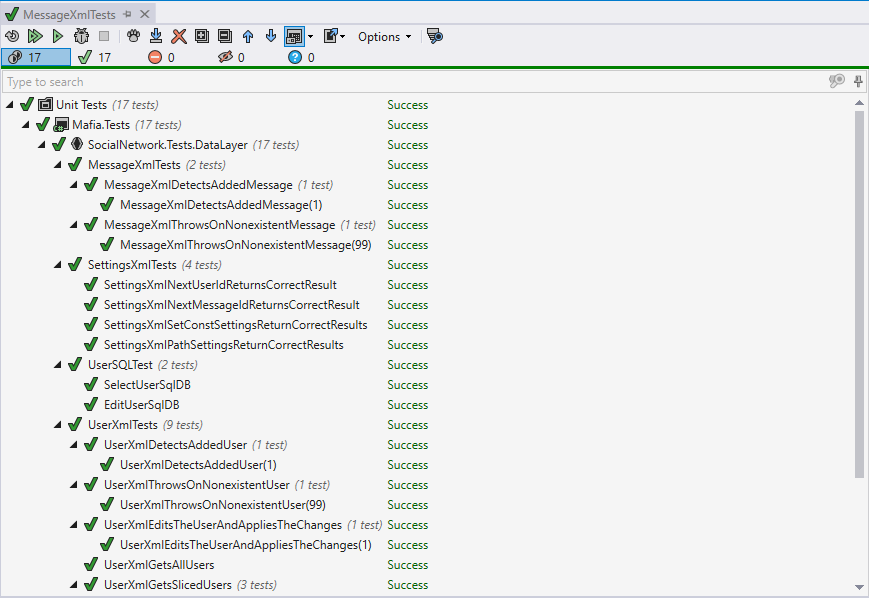
* Windows Server
* Процесор: Мінімум: 1,4 ГГц 64-розрядний процесор
* Оперативна пам'ять: Мінімум: 512 МБ
* Дисковий простір: Мінімум: 32 ГБ

Інші вимоги:

* DVD-привід
* Super VGA (800 х 600) або з більш високою роздільною здатністю
* Клавіатура і миша Microsoft® (або інший сумісний вказівний пристрій)
* Доступ в Інтернет
* Для комунікації між пристроями потрібні мобільні пристрої на платформі Android версії 4.0 і вище з вбудованим WiFi модулем.
* Будь який з сучасних браузерів для перегляду статистики.

## 6.4 Тестування

Для тестування модулю БД я використовував функціональні тести, та юніт тести. Тести написані для бази даних розроблялись на мові С# так як тестування проводилось на рівні сервісу. Взагалом було написано 17 тестів які виконували різноманітні операції над БД. Структура модуля тестування наведена на рис.6.10.



*Рис.6.10 Структура проекту з тестами*

Як бачимо всі тести виконались вірно що вказує прапорець навпроти кожного класу чи методу для якого він виконувався.

Детально розглянемо один тест який написаний для перевірки коректності роботи функціх оновлення користувача в базі данних

[Test]

public void EditUserSqlDB()

{

User testUser = new User("TestF2", "TestL", "aaasd", "Novik");

UserSQL connectUserSql = new UserSQL();

connectUserSql.EditUser(testUser);

Assert.AreEqual(1, 1, 1);

}

public void EditUser(User editedUser)

{

SqlCommand editSqlCommand = new SqlCommand();

editSqlCommand.CommandText = "UPDATE tUsers SET " +

"uFirstName = @FirstName, " +

"uLastName = @LastName, " +

"uEmail = @Email, " +

"uPassword = CONVERT(char(128),HashBytes('SHA2\_512',@Password),2) " +

"Where " +

"uEmail = @Email";

editSqlCommand.CommandType = CommandType.Text;

editSqlCommand.Connection = connection;

editSqlCommand.Parameters.AddWithValue("@FirstName", editedUser.FirstName);

editSqlCommand.Parameters.AddWithValue("@LastName", editedUser.LastName);

editSqlCommand.Parameters.AddWithValue("@Email", editedUser.Email);

editSqlCommand.Parameters.AddWithValue("@Password", editedUser.Password);

editSqlCommand.ExecuteNonQuery();

}

## 6.5 Опис проведених експериментів

Приклад запиту в рядку браузера

http://localhost:61295/WebService1.asmx/GetAllUSers

Результат

<User>

<ID>2</ID>

<FirstName>Bogdan</FirstName>

<LastName>Novosad</LastName>

<Email>bogdan.novosad.novik@gmail.com</Email>

<Password>

BC547750B92797F955B36112CC9BDD5CDDF7D0862151D03A167ADA8995AA24A9AD24610B36A68BC02DA24141EE51670AEA13ED6469099A4453F335CB239DB5DA

</Password>

</User>

<User>

<ID>3</ID>

<FirstName>Volodumr</FirstName>

<LastName>Derba</LastName>

<Email>XB@gmail.com</Email>

<Password>

92A891F888E79D1C2E8B82663C0F37CC6D61466C508EC62B8132588AFE354712B20BB75429AA20AA3AB7CFCC58836C734306B43EFD368080A2250831BF7F363F

</Password>

</User>

<User>

<ID>4</ID>

<FirstName>Marina</FirstName>

<LastName>Paschenko</LastName>

<Email>marina@gmail.com</Email>

<Password>

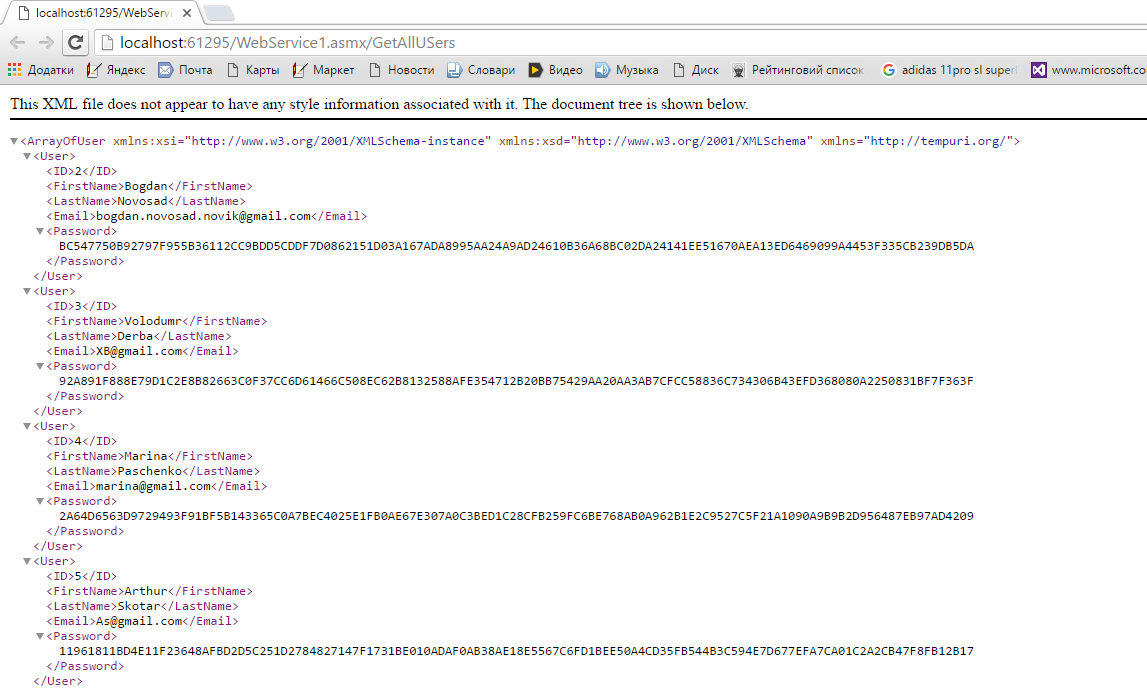
2A64D6563D9729493F91BF5B143365C0A7BEC4025E1FB0AE67E307A0C3BED1C28CFB259FC6BE768AB0A962B1E2C9527C5F21A1090A9B9B2D956487EB97AD4209

</Password>

</User>

На рис.6.11 зображено приклад запиту з вікна браузера Chrome по функції

GetAllUsers яка повертає список всіх користувачів



*Рис 6.11 Запит типу get за допомогою браузера Chrome*

Запит з програми Postman рис.6.12.

Рядок запиту - http://localhost:61295/WebService1.asmx/GetUSerById?userId=4

Результат:

<?xml version="1.0" encoding="utf-8"?>

<User xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://tempuri.org/">

<ID>4</ID>

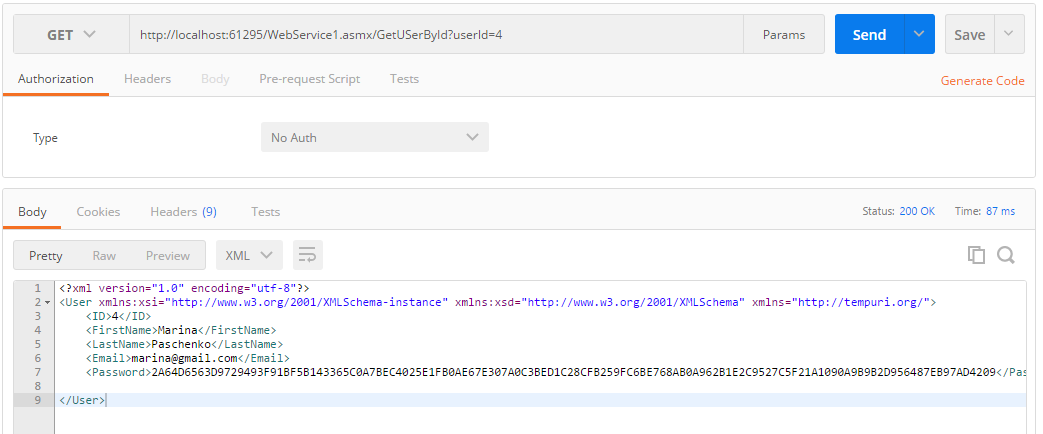
<FirstName>Marina</FirstName>

<LastName>Paschenko</LastName>

<Email>marina@gmail.com</Email>

<Password>2A64D6563D9729493F91BF5B143365C0A7BEC4025E1FB0AE67E307A0C3BED1C28CFB259FC6BE768AB0A962B1E2C9527C5F21A1090A9B9B2D956487EB97AD4209</Password>

</User>



*Рис 6.12. Запит типу get за допомогою програми Postman*

**РОЗДІЛ 7. ЕКОНОМІЧНА ЧАСТИНА**

**7.1 Економічна характеристика проектного рішення**

**(програмного продукту)**

Метою цієї бакалаврської кваліфікаційної роботи є створення автоматизованої системи підтримки рольової гри на прикладі гри «Мафія». Ця система працює на основі трьохрівневої архітектури «клієнт-сервер», а результати ігор зберігаються у вільному доступі в мережі Інтернет. Ця база надає статистичну інформацію для гравців.

Потреба у таких системах є дуже великою, оскільки автоматизовані системи ще не були створені у цій галузі. Такі типи рольових ігор тільки набирають обороти і з кожним днем захоплюють все більше і більше прихильників.

Для створення цієї системи був залучений один розробник,один який відповідав за три програмні рішення. Робота над цією системою тривала по 5 годин протягом 45 днів. Під час роботи використовувались один мобільний пристрій на платформі Android та 2 комп’ютери, сумарно ці пристрої використовували 300 ват на годину електроенергії

**7.2** **Інформаційне забезпечення та формування гіпотези щодо потреби розроблення продукту.**

Комп’ютеризація є невід’ємним процесом для будь-якої галузі в сучасному світі. Тому всі заклади, які проводять такий тип ігор, і взагалі всі бажаючі любителі цієї рольової гри матимуть велику перевагу, використовуючи цю систему по всьому світі.

Наша система надає ряд переваг як і для гравців, так і для організацій, які проводять цей тип ігор. І це призведе до більшої популяризації тих закладів, які будуть використовувавати подібні системи.

Кожен гравець рольових ігор має великий ряд правил та обмежень. Це використовується як зі сторони сервера, так і зі сторони гравця. А найбільш цінною є статистика, яка зберігатиметься на сервісі і дасть змогу кожному користувачу відслідковувати свою активність.

Ця система складатиметься з багатьох модулів. Серверна частина - для ведучого, клієнтська - для гравців, БД - для збереження даних, сервіс - для поширення результатів, сайт- для відображення.

Якщо розглянути аналог моєї системи, який зараз присутні на ринку – це додаток «Mafia Project»,- то можна сказати наступне. Дана система не надає ті послуги які є потрібні для проведення рольової гри.

**7.3** **Оцінювання та аналізування факторів зовнішнього та внутрішнього середовищ.**

Цей розділ передбачає оцінювання та аналіз факторів зовнішнього та внутрішнього середовищ групою експертів.

Фактори зовнішні оцінюються за шкалою [-5;5], при цьому межі шкали відображають максимальний негативний та позитивний вплив факторів на організацію, 0 демонструє, що фактор впливає на організацію нейтрально.

Фактори внутрішні оцінюються за шкалою [0;5], при цьому 0 демонструє недостатній розвиток, відсутність чи катастрофічний стан фактору внутрішнього середовища, оцінка 5 демонструє високий рівень розвитку даного фактору.

Сума ваг усіх факторів становить одиницю, тобто рівень вагомості для кожного фактору визначається за допомогою коефіцієнтів. Зважений рівень впливу факторів розраховується як добуток впливу фактору у балах та рівня вагомості. Результати експертних оцінок впливу факторів зовнішнього середовища наведено у табл. 7.1.

*Таблиця 7.1*

**Результати експертного оцінювання впливу факторів зовнішнього та внутрішнього середовищ**

|  |  |  |  |
| --- | --- | --- | --- |
| **Фактори** | **Середня експертна оцінка, бали** | **Середня вагомість факторів** | **Зважений рівень впливу, бали** |
| **1** | **2** | **3** | **4** |
| **Фактори зовнішнього середовища** |  | | |
| Споживачі | 4 | 0,11 | 0,44 |
| Постачальники | 2 | 0,10 | 0,20 |
| Конкуренти | -3 | 0,10 | -0,30 |
| Державні органи влади | 0 | 0,05 | 0,00 |
| Інфраструктура | -1 | 0,06 | -0,06 |
| Законодавчі акти | 0 | 0,10 | 0,00 |
| Профспілки, партії та інші громадські орган. | 0 | 0,05 | 0,00 |
| Система економічних відносин в державі | 0 | 0,06 | 0,00 |
| Організації-сусіди | 2 | 0,01 | 0,02 |
| Міжнародні події | 1 | 0,01 | 0,01 |
| Міжнародне оточення | 1 | 0,03 | 0,03 |
| Науково-технічний прогрес | 3 | 0,07 | 0,21 |
| Політичні обставини | 0 | 0,06 | 0,00 |
| Соціально-культурні обставини | 0 | 0,05 | 0,00 |
| Рівень техніки та технологій | 3 | 0,04 | 0,12 |

*Продовження табл. 7.1*

|  |  |  |  |
| --- | --- | --- | --- |
| Особливості міжнародних економічних відносин | 0 | 0,02 | 0,00 |
| Стан економіки | -1 | 0,08 | -0,08 |
| Загальна сума | 11 | 1 | 0,59 |
| **Фактори внутрішнього середовища** |  | | |
| Цілі | 3 | 0,11 | 0,33 |
| Структура | 4 | 0,16 | 0,64 |
| Завдання | 2 | 0,07 | 0,14 |
| Технологія | 4 | 0,20 | 0,80 |
| Працівники | 4 | 0,21 | 0,84 |
| Ресурси | 2 | 0,25 | 0,50 |
| Загальна сума | 19 | 1,00 | 3,25 |
|  |  |  |  |

Згідно з результатами аналізу експертного оцінювання впливу зовнішніх факторів можна зробити висновок, що найбільш впливовими в позитивному плані є наступні фактори: споживачі, науково-технічний прогрес, рівень техніки та технологій. Споживачі(користувачі) системи зацікавлені у користуванні і розвитку системи, оскільки вона передбачає спрощення процесу рольових ігор. Науково-технічний прогрес дає можливість розробки ефективного програмного продукту за мінімальні періоди часу. Рівень техніки та технологій дає велику базу для користувачів в плані розвитку системи та підтримки її на більш сучасному рівні технологій. Найбільш негативний вплив на проект мають зовнішній фактор конкуренти.

Підсумувавши результати аналізу експертного оцінювання впливу внутрішніх факторів можна зробити висновок, що найбільш позитивний вплив на проект мають наступні фактори: структура, працівники і технології. Структура системи створює хороші умови для її розробки . Працівники позитивно впливають на систему, оскільки вони є основними носіями ідеї створення даної системи. Сучасні технології створюють сприятливі умови для розробки системи, оскільки вони відкривають великі опції реалізації системи.

**7.4** **Формування стратегічних альтернатив.**

Перша група стратегічних альтернатив. Критеріями поділу альтернативних стратегій розвитку є існуючий продукт (програмне забезпечення) та новий, а також супутні послуги.

**Існуюче**

**Нове**

**Стратегія розроблення нового продукту**

**Стратегія нового продукту з супутніми послугами**

**Стратегія розвитку існуючого продукту**

**Стратегія розвитку існуючого продукту з супутніми послугами**

**Продукт (проектне рішення)**

**Додаткові послуги**

**Наявні**

**Відсутні**

*Мал. 7.4.1. Стратегічні альтернативи*

Стратегія розроблення нового продукту (проектного рішення) характеризується створенням абсолютно нового програмного забезпечення, яке дає змогу вирішити новоутворені потреби людини, суспільства, економіки тощо.

Стратегія розвитку існуючого продукту (проектного рішення) означає модифікацію програмного забезпечення, його якісних характеристик.

Стратегія розвитку існуючого продукту (проектного рішення) з супутніми послугами означає пропонування на ринку модифікованого програмного забезпечення із додатковими послугами (встановлення, супроводження, коригування, адаптування до специфіки конкретного підприємства тощо).

Стратегія нового продукту (проектного рішення) з супутніми послугами означає розроблення нового програмного забезпечення та пропонування при його експлуатації додаткових послуг.

Друга група стратегічних альтернатив. Критеріями поділу альтернативних стратегій розвитку є існуючий ринок та продукт, новий ринок та продукт.

**Існуючий**

**Новий**

**Стратегія розвитку продукту**

**Диверсифікація**

**Стратегія глибокого проникнення продукту**

**Стратегія**

**розвитку**

**ринку**

**Продукт (проектне рішення)**

**Ринок**

**Новий**

**Існуючий**

*Мал. 7.4.2. Стратегічні альтернативи*

Глибше проникнення на ринок полягає в використанні існуючого продукту (проектного рішення) для збільшення частки на існуючому ринку. Якщо фірма володіє достатніми ресурсами та потужностями для виготовлення існуючого продукту, то ця стратегія є найменш ризикованою. Однак активне зростання на існуючому ринку призведе до зростання конкуренції. Стратегія буде успішною за умови обмежень у ресурсах та потужностях конкурентів або стрімкому розвитку самого ринку. Слід зазначити, що кожен ринок за обсягом має свій ліміт і, якщо підприємство буде прагнути розвиватись, то воно повинно використовувати інші запропоновані стратегії.

Стратегія розвитку ринку полягає в використанні існуючого продукту (програмного забезпечення) або незначній його модифікації для виходу на новий сегмент ринку, весь ринок або іноземний ринок. Ця стратегія є з вищим рівнем ризику, оскільки необхідно виходити на новий ринок, де можуть бути інші правила гри, вимоги та смаки споживачів тощо.

Стратегія розвитку продукту полягає у створенні нового продукту (програмного забезпечення) для існуючого сегменту ринку. Ця стратегія є досить ризиковою, оскільки вимагає створення нового продукту (програмного забезпечення) для існуючого сегменту споживачів. Однак, якщо ринок починає зменшувати обсяги та існуючий продукт є на етапі зрілості та падіння, тоді доцільно застосовувати стратегію розвитку продукту.

Стратегія диверсифікації реалізується шляхом виходу на нові сфери бізнесу, тобто розширення номенклатури товарів, послуг тощо.

Розроблюваний програмний продукт є новим. Аналогічно ідентичних програмних продуктів, що вирішують обрану проблему, не існує, тому ринок, на який виходить розроблюваний продукт, є новим. Супутніх послуг не передбачається. Для першої групи стратегічних альтернатив обрано стратегію розробки нового продукту. Для другої групи стратегічних альтернатив обрано стратегію диверсифікації.

**7.5** **Бюджетування.**

Бюджетування є комплексно обґрунтованою системою розрахунку витрат, пов’язаних з виготовленням та реалізацією продукту, яка дає можливість здійснити аналіз витрат та розробити заходи щодо підвищення рентабельності виробництва. На даному етапі необхідно визначити собівартість продукту, який розробляється, та економічно обґрунтувати доцільність вибору однієї із стратегій. Результати розрахунків узагальнено у табл. 7.2 – табл. 7.8

*Таблиця 7.2*

**Бюджет витрат матеріалів та комплектуючих виробів**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Назва матеріалів та комплектуючих** | **Марка, тип, модель** | **Фактична кількість, шт.** | **Ціна за одиницю, грн.** | **Разом, грн.** |
| **1** | **2** | **3** | **4** | **5** |
| Ноутбук | Asus K55VJ | 1 | 16 500,00 | 16 500,00 |
| Мишка | Asus Seashell | 2 | 150,00 | 300,00 |
| Клавіатура | Gemix KB-160 | 1 | 122,00 | 122,00 |
| Мобільний телефон | LG Spirit | 1 | 3 000,00 | 3 000,00 |
| ПК | Artline Business B23 | 1 | 5 864,00 | 5 864,00 |
| Антивірусне ПЗ | ESET Smart Security 5 | 2 | 600,00 | 1 200,00 |
| Стіл офісний | - | 2 | 1 600,00 | 3 200,00 |
| Стілець офісний | - | 4 | 250,00 | 1 000,00 |
| Шафа | - | 1 | 2 570,00 | 2 570,00 |
| Разом: |  | 13 |  | 33 756,00 |
|  |  |  |  |  |

Витрати на оплату праці включають заробітну плату (ЗП) всіх категорій працівників, безпосередньо зайнятих на всіх етапах проектування. Розмір ЗП обчислюється на основі трудоємності відповідних робіт у людиноднях та середньої ЗП відповідних категорій працівників.

Для розробки програмного продукту необхідні чотири спеціалісти, а саме: керівник проекту, розробник, працівник адміністративної сфери, менеджер зі збуту.

Розрахунок витрат на оплату праці працівників проекту вводимо в таблицю 6.3.

*Таблиця 7.3*

**Бюджет витрат на оплату праці**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Посада, спеціальність** | **Кількість працівників, осіб** | **Час роботи, дні** | **Денна заробітна плата працівників, грн.** | **Сума витрат на оплату праці, грн.** |
| **1** | **2** | **3** | **4** | **5** |
| Основна заробітна плата |  | | | |
| Керівник проекту | 1 | 45 | 450,00 | 20 250,00 |
| Розробник | 1 | 45 | 600,00 | 27 000,00 |
| Працівник адміністративної сфери | 1 | 15 | 300,00 | 4 500,00 |
| Менеджер зі збуту | 1 | 7 | 250,00 | 1 750,00 |
| Разом: | 4 |  |  | 53 500,00 |

*Таблиця 7.4*

**Бюджет обов’язкових відрахувань та податків**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Посада,спеціальність** | **Витрати на оплату праці** | **Сума податку з доходів фізичних осіб\*, грн.** | **Сума військового збору, грн.**  **(1,5%)** | **Заробітна плата до видачі** | **Нарахуван-ня на заробітну плату\*\*, грн.** |
| **1** | **2** | **3** | **4** |  | **5** |
| Керівник проекту | 20 250,00 | 3 645,00 | 303,75 | 16 301,25 | 4 455,00 |
| Розробник | 27 000,00 | 4 860,00 | 405,00 | 21 735,00 | 5 940,00 |
| Працівник адміністративної сфери | 4 500,00 | 810,00 | 67,50 | 3 622,50 | 990,00 |
| Менеджер зі збуту | 1 750,00 | 315,00 | 26,25 | 1 408,75 | 385,00 |
| Разом: | 53 500,00 | 9 630,00 | 431,25 |  | 11 770,00 |

\* ставка податку з доходів фізичних осіб відповідає діючій на момент виконання випускної роботи (18 %);

\*\* ставка нарахувань на фонд оплати праці відповідає діючій на момент виконання роботи (22%).

*Таблиця 7.5*

**Бюджет загальновиробничих витрат**

|  |  |
| --- | --- |
| **Статті витрат** | **Сума, грн.** |
| **1** | **2** |
| *Змінні загальновиробничі витрати, у т.ч.:* |  |
| * заробітна плата допоміжного персоналу | 0,00 |
| * витрати на МШП | 450,00 |
| * витрати на електроенергію | 500,00 |
| * витрати на ремонт | 500,00 |

*Продовження табл. 7.5*

|  |  |
| --- | --- |
| * інші змінні витрати | 0,00 |
| Разом змінних витрат: | 1 450,00 |
| *Постійні загальновиробничі витрати, у т.ч.:* |  |
| - комунальні послуги; | 1 500,00 |
| - витрати на оренду; | 7 000,00 |
| - інші постійні витрати (амортизація); | 450,00 |
| Разом постійних витрат: | 8 950,00 |
| *Разом загальновиробничих витрат:* | 10 400,00 |

*Таблиця 7.6*

**Бюджет адміністративних витрат на збут**

|  |  |
| --- | --- |
| **Статті витрат** | **Сума, грн.** |
| **1** | **2** |
| Адміністративні витрати, у т.ч.: |  |
| * заробітна плата адміністративного персоналу | 450,00 |
| * відрахування від заробітної плати | 990,00 |
| * витрати на МШП | 450,00 |
| * итрати на відрядження | 0,00 |
| * витрати на ремонт | 500,00 |
| * витрати на паливно-мастильні матеріали | 0,00 |
| * витрати на сплату податків, зборів | 0,00 |
| * знос адміністративного обладнання | 0,00 |
| * інші адміністративні витрати | 0,00 |
| Разом адміністративних витрат: | 6 440,00 |
| Витрати на збут, у т.ч.: |  |
| * заробітна плата менеджерів зі збуту | 1 750,00 |
| * відрахування від заробітної плати | 385,00 |
| * витрати на гарантійний ремонт | 0,00 |
| * витрати на відрядження; | 0,00 |
| * витрати на гарантійне обслуговування | 0,00 |
| * витрати на налагодження і експлуатацію | 0,00 |
| * витрати на паливно-мастильні матеріали | 0,00 |
| * витрати на рекламу | 0,00 |
| * інші витрати на збут | 0,00 |
| Разом витрат на збут: | 2 035,00 |

*Таблиця 7.7*

**Зведений кошторис витрат на розробку проектного рішення (продукту)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Статті витрат** | **Одиниці виміру** | **Фактична кількість** | **Ціна одиниці, грн.** | **Разом, грн.** |
| **1** | **2** | **3** | **4** | **5** |
| Сировина і матеріали | шт. | 13 | - | 33 756,00 |
| Купівельні напівфабрикати та комплектуючі вироби | - | - | - | - |
| Зворотні відходи (вираховуються) | - | - | - | - |
| Основна заробітна плата | грн. | - | - | 53 500,00 |

*Продовження таб.7.7*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Додаткова заробітна плата |  | - | - | - |
| Витрати на утримання та експлуатацію устаткування | - | - | - | - |
| Загальновиробничі витрати, у т.ч.: |  | | | |
| * змінні; | грн. | - | - | 1 450,00 |
| * постійні; | грн. | - | - | 8 950,00 |
| Разом виробничих витрат: | грн. | - | - | 97 656 |
| Адміністративні витрати | грн. | - | - | 6 440 |
| Витратити на збут | грн. | - | - | 2 035 |
| Інші операційні витрати | - | - | - | - |
| Разом виробничих і операційних витрат: |  |  |  | 106 131 |

Для визначення фінансових результатів, необхідно розрахувати вартість (ціну) продукту (проектного рішення), який розробляється. Ціна визначається на основі суми виробничих і операційних витрат з врахуванням рентабельності виробництва.

Ц= СБ \* Р + СБ (5.1),

де Ц - ціна проектного рішення (програмного продукту), грн.

СБ - собівартість проектного рішення (програмного продукту), грн.

Р - рентабельність, %.

Отож, при заданій рентабельності 44%, ціна проектного рішення:

Ціна = 97 656\* 0,44 + 97 656 = 140 624,64 (грн.)

Ціна з ПДВ = Ціна \* 1.2 = 140 624.64 \* 1.2 = 168 749,57 (грн.);

Дохід від реалізації продукції = Ціна з ПДВ \* кількість реалізованої продукції = 168 749,57 \* 1 = 168 749,57 (грн.);

ПДВ = Дохід від реалізації продукції / 1 = 168 749,57 \* 1 = 168 749,57 (грн.);

Чистий дохід від реалізації продукції = Дохід від реалізації продукції – ПДВ = 168 749,57 – 28 124,93= 140 624,64 (грн.);

Собівартість реалізованої продукції = Собівартість проектного рішення \* кількість реалізованої продукції = 97 656\* 1 = 97 656 (грн.);

Валовий прибуток = Чистий дохід від реалізації продукції - Собівартість реалізованої продукції = 140 624,64 - 97 656 = 42 968,64 (грн.);

Фінансовий результат від операційної діяльності = Валовий прибуток - Адміністративні витрати - Витрати на збут - Інші операційні витрати = 42 968,64 – 6 440 – 2 035 – 0.00 = 34 493,64 (грн.);

Податок на прибуток = Фінансовий результат від операційної діяльності \* розмір податку на прибуток = 34 493,64 \* 0.18 = 6 208,85 (грн.);

Чистий прибуток (збиток) = Фінансовий результат від операційної діяльності - Податок на прибуток = 34 493,64 – 6 208,85 = 28 284,78 (грн.);

*Таблиця 7.8*

**Бюджет фінансових результатів**

|  |  |
| --- | --- |
| **Показники** | **Сума, грн.** |
| **1** | **2** |
| Дохід від реалізації (10 шт) | 168 749,57 |
| Податок на додану вартість (20 %) | 28 124,93 |
| Чистий дохід від реалізації продукції | 140 624,64 |
| Собівартість реалізованої продукції | 97 656 |
| Валовий прибуток | 42 968,64 |
| Операційні витрати: |  |
| * адміністративні витрати | 6440 |
| * витрати на збут | 2035 |
| * інші операційні витрати | 0,00 |
| Фінансовий результат від операційної діяльності | 34 493,64 |
| Податок на прибуток (18%) | 6 208,85 |
| Чистий прибуток (збиток) | 28 284,7 |

**7.6 Вибір стратегії.**

Виконавши відповідний аналіз можна зробити висновок про те, що розробляти даних програмний продукт є доцільно. Наша система розробляється для ринку, на якому подібних систем ще не існує. Тому вона посяде ключове місце у цій сфері розваг.

Оскільки для цього продукту обрано стратегію мінімально життєздатного продукту, то його запуск дасть змогу швидко отримати зворотній зв’язок від потенційних користувачів, допоможе дізнатись їхню конкретну думку про важливість та зручність такої системи, і в подальшому ми зможемо розширювати функціонал даної системи.

Фактори зовнішнього середовища непогано впливають на проектування і реалізацію системи, серед негативних факторів доцільно виділити споживачі, науково-технічний прогрес, рівень техніки та технологій.

Підсумувавши результати аналізу експертного оцінювання впливу внутрішніх факторів можна зробити висновок, що найбільш позитивний вплив на проект мають наступні фактори: структура, працівники і технології.

Оскільки система є повністю розробленою наново, тому було обрано стратегію розроблення нового продукту та стратегію диверсифікації.

З обчислень, наведених вище, бачимо, що собівартість продукту дорівнює 97 656 грн., для рентабельності 44% та виведення на ринок програмного продукту чистий дохід від реалізації системи складе 140 624,64 грн., а чистий прибуток - 28 284,7 грн.

# ВИСНОВКИ

У результаті виконання бакалаврської кваліфікаційної роботи розроблено програмний комплекс для підтримки групових рольових ігор. Цей комплекс складається з декількох програмних рішень, які є зручним інструментом для всіх любителів гри «Мафія», оскільки забезпечують велику кількість функцій, що спрощують проведення та підтримку таких ігор.

Під час виконання роботи проведено детальний аналіз предметної області і виявлено основні проблеми. Визначено основний функціонал системи, за предмет дослідження і реалізації обрана групова рольова гра «Мафія», яка має значну аудиторію прихильників по всьому світі. Проаналізовано аналог системи, яка існує на даний момент і визначено її основні переваги і недоліки.

Розроблена система дає змогу створити мобільну локальну мережу для проведення гри, автоматично генерувати ролі учасників, проводити голосування, стежити за поточним статусом в грі та інші функціонали. Результати гри збері­гаються в базі даних на веб ресурсі, яка також містить статистику всіх учасників.

Розроблено діаграму компонентів і архітектурно визначено всі модулі та складові, системи що розробляється. Для кожного з компонентів був запроектовано окремий модуль, який відповідав за конкретні дії в системі. Користувацький інтерфейс розроблявся таким чином, щоб бути інтуїтивно зрозумілим і забезпечувати легку навігацію по користувацьких історіях.

Очікується, що у разі поширення розробленої системи, вона зможе привернути увагу не тільки прихильників гри «Мафія» але й інших, оскільки перенесення системи для підтримки іншої групової рольової гри не вимагає принципових змін у функціоналі системи і може бути виконане її переналаштуванням. Модуль статистики системи може зберігати і опрацьовувати результати інших довільних групових рольових ігор.

Розвиток розробленої системе може здійснюватися у напрямках. Один з них доповнення функціоналу групової рольової гри «Мафія», а інший - це розвиток платформи для побудови систем підтримки інших улюблених групових рольових ігор. Ці два напрямки можуть існувати і разом.

# ЛІТЕРАТУРА

1. Rivera, J., and R. van der Meulen. "Gartner Says Worldwide Tablet Sales Grew 68 Percent in 2013, With Android Capturing 62 Percent of the Market."*Garnter Inc* (2014).
2. Klaver, C. "Windows Mobile advanced forensics."*digital investigation*6.3 (2010): 147-167.
3. Victor, H. "Android's Google Play beats App Store with over 1 million apps, now officially largest."*Retrieved January*16 (2013): 2014.
4. Брюс Эккель. Философия Java. 3-е издание. Издательство «Питер». Петербург 2003. ISBN 5-88782-105-1
5. https://uk.wikipedia.org/wikiМафія\_(гра)
6. Макарчук, Т. А., В. Ф. Минаков, and А. В. Артемьев. "Мобильное обучение на базе облачных сервисов." *Современные проблемы науки и образования*2 (2013).
7. Майер, Рето.*Android 2. Программирование приложений для планшетных компьютеров и смартфонов*. Litres, 2014.
8. Ken Arnold, James Gosling, David Holmes Java(TM) Programming Language, The 4th Edition
9. Wolfson, Mike, and Donn Felker.Android Developer Tools Essentials: Android Studio to Zipalign. " O'Reilly Media, Inc.", 2013.
10. Agrawal, Sanjay, et al. "Database tuning advisor for microsoft SQL server 2005: demo."*Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. ACM, 2005.
11. Snell, James, Doug Tidwell, and Pavel Kulchenko.*Programming Web services with SOAP*. " O'Reilly Media, Inc.", 2001.
12. Charland, Andre, and Brian Leroux. "Mobile application development: web vs. native."*Communications of the ACM* 54.5 (2011): 49-53.
13. Lowy, Juval. Programming WCF services. " O'Reilly Media, Inc.", 2007.
14. VanLengen, Craig A., and John D. Haney. "Creating web services using asp. net." Journal of Computing Sciences in Colleges 20.1 (2004): 262-275.
15. Riad, Alaa M., Ahmed E. Hassan, and Qusay F. Hassan. "Design of SOA-based Grid Computing with Enterprise Service Bus." AISS 2.1 (2010): 71-82.
16. Nielsen, Paul, and Uttam Parui. Microsoft SQL server 2008 bible. Vol. 607. John Wiley & Sons, 2011.
17. Larson, Brian. Delivering Business Intelligence with Microsoft SQL Server 2012. McGraw-Hill Osborne Media, 2012.
18. Sayers, Eric W., et al. "Database resources of the national center for biotechnology information." Nucleic acids research 40.Database issue (2012): D13.
19. Crockford, Douglas. "The application/json media type for javascript object notation (json)." (2006).
20. Graham, Ian S. The HTML sourcebook. John Wiley & Sons, Inc., 1995.
21. Raman, T. V. "Cascaded speech style sheets." Computer networks and ISDN systems 29.8 (1997): 1377-1383.
22. Libby, Alex. Instant LESS CSS Preprocessor How-to. Packt Publishing Ltd, 2013.
23. Stothard, Paul. "The sequence manipulation suite: JavaScript programs for analyzing and formatting protein and DNA sequences." Biotechniques 28.6 (2000): 1102-1104.
24. Edmonds, Alan Robert. Angular momentum in quantum mechanics. Princeton University Press, 1996.
25. Crockford, Douglas. "The application/json media type for javascript object notation (json)." (2006).

# Додаток А

**Клас основних ролей**

package com.example.abc.server;

/\*\*

\* Created by Богдан on 17.12.2015.

\*/

enum Position{

Bleak,

White,

NullStorona

}

enum Rol\_name{

NullRols("Порожня роль",Position.NullStorona),

Don("Don",Position.Bleak),

Mafia("Mafia",Position.Bleak),

Sheriff("Sheriff",Position.White),

Civilians("Civilians",Position.White),

Doctor("Doctor",Position.White);

Position position;

String definition;

Rol\_name(String s,Position a) {

position = a;

definition = s;

}

}

public class Rols {

Rol\_name rols\_player = Rol\_name.NullRols;

public Rols(Rol\_name a){

rols\_player = a;

}

}

# Додаток В

**Клас дадаткових ролей**

package com.example.abc.ыукмук;

/\*\*

\* Created by Богдан on 17.12.2015.

\*/

enum time{

Police\_Round,

Throughout\_the\_game,

Beginning\_of\_the\_game

}

enum Second\_rols\_name{

Neutral\_card("Ця карта ні на що не впливає. Не відкривайте її до кінця гри.",time.Throughout\_the\_game),

Spy("Використовуй цю карту в кінці раунду Поліції і подивися на карту одного з гравців - дізнайся, хто він, - мафіозі або поліцейський.",time.Police\_Round),

The\_vest("Якщо тебе вб'ють, не відкриєш свою карту і не показуй, хто ти. Просто покажи цю карту і залишайся в грі. Карту можна використовувати тільки один раз.",time.Throughout\_the\_game),

Ringleader("Покажи цю карту всім на початку гри. Якщо під час голосування проти когось буде рівну кількість голосів - твій голос буде вирішальним.",time.Beginning\_of\_the\_game),

Unexpected\_shot("Покажи цю карту і вкажи на того, кого ти хочеш вбити . Гравець, на якого ти вказав, відкриває свою карту і виходить з гри.",time.Police\_Round),

Help("Покажи цю карту, коли когось із членів твоєї команди вб'ють, і спаси його від смерті . Карту можна використовувати тільки один раз.",time.Police\_Round),

The\_last\_shoot("Якщо під час гри тебе вбили, ти маєш право останнього пострілу - можеш убити кого-небудь з гравців. Він теж вважається вбитим і виходить з гри.",time.Throughout\_the\_game),

Changing\_roles("Якщо ти отримав цю карту, можеш помінятися ролями з будь-яким гравцем ( помінятися картами), але не знаючи, хто він.",time.Beginning\_of\_the\_game);

String defin;

time when;

Second\_rols\_name(String s, time when\_1) {

defin = s;

when = when\_1;

}

}

public class SecondRols {

Position position = Position.NullStorona;

Second\_rols\_name secondRols = Second\_rols\_name.Neutral\_card;

public SecondRols(Second\_rols\_name a){

secondRols = a;

}

}

# Додаток С

**Локальний сервер гри**

package com.example.abc.s;

import android.app.Activity;

import android.content.Intent;

import android.os.AsyncTask;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.TextView;

import org.json.\*;

import java.io.ByteArrayOutputStream;

import java.io.DataInputStream;

import java.io.DataOutput;

import java.io.DataOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.io.PrintStream;

import java.net.InetAddress;

import java.net.NetworkInterface;

import java.net.ServerSocket;

import java.net.Socket;

import java.net.SocketException;

import java.net.UnknownHostException;

import java.util.ArrayList;

import java.util.Enumeration;

final public class MainActivity extends Activity {

static ArrayList<Player> mas\_socket = new ArrayList<Player>();

TextView info, infoip, msg;

String message = "";

static ServerSocket serverSocket;

public void asdfg(View a) throws IOException {

/\*SocketServerStringThread socketServerString = new SocketServerStringThread(

mas\_socket.get(0).socket\_player,"Rol:Civilian;");x

socketServerString.run();\*/

Reader\_all\_Thread readerall = new Reader\_all\_Thread();

readerall.run();

Intent config\_windows = new Intent(this,Config\_Windows.class);

startActivity(config\_windows);

/\*

if(mas\_socket.get(0).socket\_player!=null){

MyWriterServerTask ms = new MyWriterServerTask(mas\_socket.get(0),"Rol:Cuvilian;");

ms.execute();

}\*/

}

/\*public class MyWriterServerTask extends AsyncTask<Void, Void, Void> {

Player player;

DataOutputStream outputStream;

String mesage;

MyWriterServerTask(Player a, String mes){

player = a;

mesage = mes;

}

@Override

protected void onPreExecute() {

super.onPreExecute();

}

@Override

protected Void doInBackground(Void... arg0) {

try {

outputStream = new DataOutputStream(player.socket\_player.getOutputStream());

outputStream.writeUTF(message);

//outputStream.write(0);

} catch (UnknownHostException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}finally{

}

return null;

}

@Override

protected void onPostExecute(Void result) {

super.onPostExecute(result);

}

}\*/

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

info = (TextView) findViewById(R.id.info);

infoip = (TextView) findViewById(R.id.infoip);

msg = (TextView) findViewById(R.id.msg);

infoip.setText(getIpAddress());

Thread socketServerThread = new Thread(new SocketServerThread());

socketServerThread.start();

}

@Override

protected void onDestroy() {

super.onDestroy();

if (serverSocket != null) {

/\*try {

//serverSocket.close();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}\*/

}

}

private class Reader\_socket\_one extends AsyncTask<Void, Void, Void> {

Player players;

InputStream inputStream;

String response = "";

public Reader\_socket\_one(Player a){

players = a;

}

/\*\*

\* Override this method to perform a computation on a background thread. The

\* specified parameters are the parameters passed to {@link #execute}

\* by the caller of this task.

\* <p/>

\* This method can call {@link #publishProgress} to publish updates

\* on the UI thread.

\*

\* @param params The parameters of the task.

\* @return A result, defined by the subclass of this task.

\* @see #onPreExecute()

\* @see #onPostExecute

\* @see #publishProgress

\*/

@Override

protected Void doInBackground(Void... params) {

if(players!=null)

try {

inputStream = players.socket\_player.getInputStream();

ByteArrayOutputStream byteArrayOutputStream =

new ByteArrayOutputStream(1024);

byte[] buffer = new byte[1024];

int bytesRead;

while ((bytesRead = inputStream.read(buffer)) != -1) {

if (buffer[bytesRead] == ';') {

byteArrayOutputStream.write(buffer, 0, bytesRead);

response += byteArrayOutputStream.toString("UTF-8");

parser(response, players);

byteArrayOutputStream.reset();

}

}

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

}

static void parser(String a,Player player){

String mas\_str[] = a.split(":");

switch (mas\_str[0]){

case "Nik":{

player.setNikname(mas\_str[1]);

}

break;

}

}

private class Reader\_all\_Thread extends Thread{

@Override

public void run() {

for(int i=0;i<mas\_socket.size();i++){

Reader\_socket\_one a = new Reader\_socket\_one(mas\_socket.get(i));

a.execute();

}

}

}

public class SocketServerThread extends Thread {

Socket socket;

static final int SocketServerPORT = 6660;

int count = 0;

String [] inetaddr = new String[10] ;

@Override

public void run() {

try {

serverSocket = new ServerSocket(SocketServerPORT);

MainActivity.this.runOnUiThread(new Runnable() {

@Override

public void run() {

info.setText("Чекаю на підплючення: "

+ serverSocket.getLocalPort());

}

});

while (true) {

socket = serverSocket.accept();

//Додаю нового клієнта

mas\_socket.add(new Player(socket));

//Отримую адресу

String temp = socket.getInetAddress().toString() ;

//Відкидаю порт

String []temp1 = temp.split(":");

//Перевыряю чи вже э подыбне підключення

Boolean f = false;

for(int i=0;i<count;i++){

if(inetaddr[i]!=null){

if(inetaddr[i].equals(temp)){

f=true;

}

}

}

//Якщо нема то запамятовую його

if(!f){

inetaddr[count] = temp1[0];

count++;

message += "#" + count + " Від " + socket.getInetAddress()

+ ":" + socket.getPort() + "\n";

MainActivity.this.runOnUiThread(new Runnable() {

@Override

public void run() {

msg.setText(message);

}

});

SocketServerReplyThread socketServerReplyThread = new SocketServerReplyThread(

socket, count);

socketServerReplyThread.run();

}

}

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

private class SocketServerReplyThread extends Thread {

private Socket hostThreadSocket;

int cnt;

SocketServerReplyThread(Socket socket, int c) {

hostThreadSocket = socket;

cnt = c;

}

@Override

public void run() {

OutputStream outputStream;

String msgReply = "Привіт, ви #" + cnt+";";

try {

outputStream = hostThreadSocket.getOutputStream();

PrintStream printStream = new PrintStream(outputStream);

printStream.print(msgReply);

//

//printStream.close();

message += "Гравець №" + cnt + " підключений\n\n";

MainActivity.this.runOnUiThread(new Runnable() {

@Override

public void run() {

msg.setText(message);

}

});

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

message += "Виникла помилка! " + e.toString() + "\n";

}

MainActivity.this.runOnUiThread(new Runnable() {

@Override

public void run() {

msg.setText(message);

}

});

}

}

private String getIpAddress() {

String ip = "";

try {

Enumeration<NetworkInterface> enumNetworkInterfaces = NetworkInterface

.getNetworkInterfaces();

while (enumNetworkInterfaces.hasMoreElements()) {

NetworkInterface networkInterface = enumNetworkInterfaces

.nextElement();

Enumeration<InetAddress> enumInetAddress = networkInterface

.getInetAddresses();

while (enumInetAddress.hasMoreElements()) {

InetAddress inetAddress = enumInetAddress.nextElement();

if (inetAddress.isSiteLocalAddress()) {

ip += "SiteLocalAddress: "

+ inetAddress.getHostAddress() + "\n";

}

}

}

} catch (SocketException e) {

// TODO Auto-generated catch block

e.printStackTrace();

ip += "Something Wrong! " + e.toString() + "\n";

}

return ip;

}

private String getIpAddress() {

String ip = "";

try {

Enumeration<NetworkInterface> enumNetworkInterfaces = NetworkInterface

.getNetworkInterfaces();

while (enumNetworkInterfaces.hasMoreElements()) {

NetworkInterface networkInterface = enumNetworkInterfaces

.nextElement();

Enumeration<InetAddress> enumInetAddress = networkInterface

.getInetAddresses();

while (enumInetAddress.hasMoreElements()) {

InetAddress inetAddress = enumInetAddress.nextElement();

if (inetAddress.isSiteLocalAddress()) {

ip += "SiteLocalAddress: "

+ inetAddress.getHostAddress() + "\n";

}

}

}

} catch (SocketException e) {

// TODO Auto-generated catch block

e.printStackTrace();

ip += "Something Wrong! " + e.toString() + "\n";

}

return ip;

}

}