МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ВИПУСКНА РОБОТА

на тему:

«Інформаційна система тестування із ідентифікацією та авторизацією студентів»

Завідувач випускаючої кафедри — **Довбиш А.С.**

Керівник роботи — **Власенко О.В.**

Студента групи КБ-61-8 — **Нога Є.Ю.**

СУМИ 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютерних наук

Затверджую_____
Зав. кафедрою Довбиш А.С.
"_____" _____2020 р.

ЗАВДАННЯ
до випускної роботи

Студента четвертого курсу, групи КБ-61-8 спеціальності "Кібербезпека" денної форми навчання Ноги Євгена Юрійовича.

Тема: "Інформаційна система тестування із ідентифікацією та авторизацією студентів"

Затверджена наказом по СумДУ
№_____ від _____2020 р.

Зміст пояснювальної записки: 1) інформаційний огляд; 2) постановка завдання й формування чітких умов для системи; 3) вибір вирішення поставленої задачі; 4) розробка програмного забезпечення інформаційної системи; 5) аналіз виконаної роботи.

Дата видачі завдання "_____"_____2020 р.
Керівник випускної роботи _____ Власенко О.В.
Завдання прийняв до виконання _____ Нога Є.Ю.

РЕФЕРАТ

Записка: 35 сторінок, 9 рисунків, 1 додаток, 9 джерел.

Мета роботи – створення інформаційної системи тестування із ідентифікацією та авторизацією студентів.

Об’єкт дослідження – процес розробки інформаційної системи.

Предмет дослідження — платформи Android та iOS, їх основні інструменти та їх рекомендоване використання.

Методи дослідження — пошук необхідних формул та значень основних показників; розробка Android-додатку.

Результати — інстальовано та налаштовано середовище розробки Visual Studio; розроблена інформаційна система, що дозволяє зручно та швидко реєструватися, виконувати вхід, проходити тестування, створювати тестування (функція доступна для викладачів).

XAMARIN-ДОДАТОК, C#, XAMARIN, OS, IOS, ANDROID,
ОПИТУВАННЯ, АВТОРИЗАЦІЯ, РЕЄСТРАЦІЯ, ТЕСТУВАННЯ

ЗМІСТ

ВСТУП 4
1 ІНФОРМАЦІЙНИЙ ОГЛЯД 5
1.1 Поняття інформаційної системи 5
1.2 Організаційна структура 6
1.3 Постановка задачі 7
2 ВИБІР МЕТОДІВ ВИРІШЕННЯ 8
2.1 Огляд існуючих операційних систем 8
2.2 Вибір мови програмування 11
2.3 Огляд середовища розробки 16
3 ПРОГРАМНА РЕАЛІЗАЦІЯ 20
3.1 Проектування додатку 20
3.2 Проектування бази даних 24
3.3 Структура проекту 24
3.4 Опис основних функцій та процедур 26
ВИСНОВКИ 28
СПИСОК ЛІТЕРАТУРИ 29
ДОДАТКИ 30

ВСТУП

У час стрімкого розвитку комп'ютерних технологій та популярності мобільних пристроїв важко уявити сучасну людину без смартфону. Провідні компанії з кожною версією додають все більше можливостей в телефони. Попит на смартфони завжди високий, багато людей слідкують за новинками. Це спричинює високу популярність мобільних додатків. Адже не завжди вдається швидко скористатися ноутбуком чи персональним комп'ютером для пошуку необхідної інформації. Смартфони дають цю можливість. Створення онлайн-оплати також додало переваг програмуванню для мобільних пристроїв.

Розробники це розуміють і при створенні певного веб-сайту або десктопного додатку одразу продумують варіант створення мобільної версії продукту. Існує чимала кількість способів це зробити.

Багато мов програмування вже має власні фреймворки для розробки мобільних версій продукту: React має React Native, Angular – ionic, C# – Xamarin, Mono, Java – Kotlin та багато інших.

У найближчі роки така тенденція – перевага використання мобільних додатків – навряд чи зміниться. Користувачів передусім буде цікавити мобільна версія продукту, яким вони користуються.

Саме тому додаток, який буде розроблюватися в рамках випускної кваліфікаційної роботи, буде виконано для смартфонів. Основне призначення додатку – тестування студентів на розуміння вивченого матеріалу. Це актуально, бо освіта – невід'ємна частина життя кожної людини. Всі проходять навчання, отримують теоретичний та практичний матеріал, для закріплення проходять тестування. Замість того, щоб давати відповіді на питання на папері, буде можливість проходження на власному телефоні. Це підвищить якість перевірки, адже відповіді на запитання будуть зберігатися в базі даних.

1 ІНФОРМАЦІЙНИЙ ОГЛЯД

В наш час смартфони відіграють важливу роль в повсякденному житті людини. Ноутбуки та персональні комп’ютери – речі, які не завжди зручно з собою носити. Саме тому швидкої популярності набирають мобільні додатки, адже вони є зручними і не поступаються програмному забезпеченюю для ноутбуків та комп’ютерів.

1.1 Поняття інформаційної системи

Інформаційна система – інтегрований набір компонентів для збору, зберігання та обробки даних, а також для надання інформації, знань та цифрових продуктів [1]. Бізнес-фірми та інші організації покладаються на інформаційні системи для здійснення своїх операцій та управління ними, взаємодії зі своїми клієнтами та постачальниками та конкурують на ринку. Інформаційні системи використовуються для управління міжорганізаційними ланцюгами поставок та електронними ринками. Наприклад, корпорації використовують інформаційні системи для обробки фінансових рахунків, управління своїми людськими ресурсами та залучення своїх потенційних клієнтів за допомогою рекламних акцій в Інтернеті. Багато великих компаній будуються повністю навколо інформаційних систем. До них належить eBay, значною мірою ринок аукціонів; Amazon, розширюється електронний торговий центр і постачальник послуг хмарних обчислень. Уряди впроваджують інформаційні системи для надання економічно ефективних послуг громадянам. Цифрові товари – такі як електронні книги, відеопродукти та програмне забезпечення – та онлайн-сервіси, такі як ігри та соціальні мережі, постачаються разом із інформаційними системами. Люди велику частину свого особистого життя покладаються на інформаційні системи, як правило, на базі Інтернету: для спілкування, навчання, шопінгу, банківської діяльності та розваг.

Першою широкомасштабною механічною інформаційною системою була таблиця перепису Германа Холлеріта. Придумана вчасно для перепису населення США 1890 року, машина Холлеріта являла собою головний крок у автоматизації, а також натхнення для розробки комп'ютеризованих інформаційних систем.

Глобальне проникнення Інтернету дозволило отримати доступ до інформації та інших ресурсів та сприяло формуванню відносин між людьми та організаціями в безпрецедентних масштабах. Прогрес електронної комерції через Інтернет призвів до різкого зростання цифрових міжособистісних комунікацій (за допомогою електронної пошти та соціальних мереж), розповсюдження продуктів (програмного забезпечення, музики, електронних книг та фільмів) та ділових операцій (купівля, продаж та реклама в Інтернеті). З широким поширенням у всьому світі смартфонів, планшетів, ноутбуків та інших комп'ютерних мобільних пристроїв, всі вони пов'язані бездротовими мережами зв'язку, інформаційні системи були розширені для підтримки мобільності як природного стану людини.

1.2 Організаційна структура

Під організаційною структурою управління розуміється цілісна сукупність з'єднаних між собою інформаційними зв'язками елементів об'єкта і органу управління, об'єднана в рамках досягнення певних цілей.

Вона відображає будову системи управління, змістом якої є функції управління, вертикальну та горизонтальну співвідношення рівнів управління, а також кількість і взаємозв'язок структурних підрозділів в межах кожного рівня .

Так як елементи у всіх вузівських системах управління багато в чому схожі, то основним критерієм відмінності структур вважають організацію взаємозв'язків.

Одним з принципів побудови складних організаційних систем є ієрархічність. Багаторівневі системи управління, засновані на концепції ієрархічної структури, функціонують в організаціях різних галузей.

1.3 Постановка задачі

1. Розробити мобільний додаток для тестування
2. Обрати необхідні інструменти для розробки додатка.
3. Функціональність додатка має забезпечити можливість тестувати студентів та викладачів
4. Забезпечити можливість створювати питання з однією правильної відповіддю та з декількома. Загальна кількість питань в рамках тесту буде 10. Якщо для обної області не вистачає питань, ця область буде виключена з вибору. Якщо питання передбачає декілька правильних відповідей, за правильну відповідь користувач отримуватиме 1 бал (фінальну оцінку за 10-бальною шкалою треба буде перевести особі, яка здійснює перевірку знань, без використання додатку). Якщо питання передбачає декілька правильних відповідей, то бал буде ділитися на кількість правильних відповідей у питанні (далі – ЧАСТКА) і далі кількість обраних правильних відповідей буде множитися на ЧАСТКУ. Якщо було обрано неправильну відповідь то від порахованої оцінки за відповідь на дане питання буде відніматися ЧАСТКА.

2 ВИБІР МЕТОДІВ ВИРІШЕННЯ

2.1 Огляд існуючих операційних систем

Android – операційна система і платформа для мобільних телефонів та планшетних комп'ютерів, створена компанією Google на базі ядра Linux. Підтримується альянсом Open Handset Alliance (OHA) [2].

Пристрої Android не просто працюють, а й роблять життя користувачів легше. Можливості цієї операційної системи дозволяють уникати пробок за допомогою GPS-навігатора, відправляти SMS на годиннику і отримувати відповіді на питання від Асистента. Під управлінням Android працюють приблизно два з половиною мільярди різних пристроїв – від телефонів з підтримкою 5G до найпередовіших планшетів.

Android – це операційна система з відкритим кодом та ОС Linux для мобільних пристроїв, таких як смартфони та планшетні комп'ютери. Android був розроблений Альянсом відкритих телефонів, який очолює Google та інші компанії.

Android пропонує уніфікований підхід до розробки додатків для мобільних пристроїв, що означає, що розробникам потрібно розробляти тільки для Android, і їх додатки повинні мати можливість працювати на різних пристроях, що працюють на Android.

Перша бета-версія Android Software Development Kit (SDK) була випущена Google у 2007 році, де в вересні 2008 року була випущена перша комерційна версія Android 1.0.

Android – потужна операційна система, яка конкурує з Apple 4GS і підтримує чудові функції. Деякі з них перераховано нижче:

1. Красивий інтерфейс користувача: основний екран ОС Android забезпечує прекрасний та інтуїтивний інтерфейс користувача.

2. Підключення: GSM / EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, NFC і WiMAX.

3. Зберігання: легка реляційна база даних SQLite використовується для цілей зберігання даних.

4. Медіа-підтримка: H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, AAC 5.1, MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF та BMP .

5. Повідомлення: SMS та MMS

6. Веб-браузер: на основі механізму макета WebKit з відкритим кодом, поєднаного з двигуном JavaScript V8 Chrome, що підтримує HTML5 та CSS3.

7. Мультитач: Android має вбудовану підтримку мультитач, яка спочатку була доступна в телефонах, таких як HTC Hero.

8. Багатозадачність: користувач може переходити з однієї задачі на іншу і в той же час різні програми можуть працювати одночасно.

9. Віджети з можливістю зміни розміру: віджети можна змінювати, щоб користувачі могли розширити їх, щоб показати більше вмісту, або зменшити їх, щоб заощадити місце.

10. Багатомовний: підтримує односторонній та двонаправлений текст.

11. GCM: Google Cloud Messaging (GCM) – це послуга, яка дозволяє розробникам надсилати короткі дані повідомлення своїм користувачам на пристрої Android, не потребуючи власного рішення синхронізації.

12. Wi-Fi Direct: технологія, яка дозволяє програмам безпосередньо виявляти та створювати пару через високошнурову мережу однорангового зв'язку.

13. Android Beam: популярна технологія на базі NFC, яка дозволяє користувачам миттєво ділитися, просто торкнувшись двох телефонів з підтримкою NFC разом.

iOS – це операційна система, розроблена компанією apple.inc. iOS працює на всіх мобільних пристроях Apple. IOS – друга за величиною в світі операційна система після Android [3].

IOS використовує мультитач-інтерфейс, в якому керування пристроєм виконується за допомогою простої текстури. Прості жести, такі як розміщення пальцями вгорі екрана пристрою, працюють досить плавно. І натискання

пальцем, щоб збільшити екран. Ця робота вільно виконується всю цю роботу вільно на iOS.

IOS робить датчики свого пристрою дуже сильними. Що полегшує користувацьку роботу, виявляючи кінчики пальців негайно. Xeos контролює всі аспекти апаратури пристрою Apple. Крім того, він обробляє всі функції програмного забезпечення. Це означає, що iOS надає повний досвід своїм користувачам. Де програмне забезпечення вміло поєднується з апаратним забезпеченням. Що дає користувачеві кращі показники роботи від апаратних засобів пристрою. Більше 2 мільйонів додатків для iOS доступні для завантаження в App Store Apple. Це відомий магазин додатків усіх пристроїв Apple.

IOS абсолютно відрізняється від інших операційних систем мобільного телефону. Тому що він зберігає всі програми в своєму пристрої всередині своєї захисної оболонки. Так що додатки тримаються подалі одне від одного і не втручаються в роботу один одного. IOS розроблений таким чином, що якщо пристрій випадково потрапить до вірусу через додатки, він може не завдати шкоди іншим програмам. Така функція не спостерігається в інших операційних системах. iOS забезпечує надійну безпеку в своєму апаратному та програмному забезпеченні:

1. Перед повним завантаженням в iOS є код низького рівня, який працює від завантажувального ПЗУ. Його функція полягає в тому, щоб перевірити, що Apple root CA підписаний відкритим ключем перед запуском завантажувача низького рівня.

2. Безпечний Enclave може бути процесором обрізання, знайденим на пристрої iOS, який містить Touch ID або Face ID. це власний захищений процес завантаження, щоб переконатися, що він повністю захищений. До складу цього співпроцесора додатково входить апаратний генератор випадкових чисел. У безпечному анклаві кожного пристрою є унікальний ідентифікатор, який надається під час його створення, і його неможливо змінити

3. На пристроях iOS може бути пароль, який не має змоги розблокувати пристрій, змінити системні налаштування та зашифрувати вміст пристрою.

4. Touch ID може бути сканером відбитків пальців, вбудованим в домашню кнопку і, можливо, бути використаним для розблокування пристрою, здійснення покупки та входження в пристрій для виконання інших завдань.

5. iOS використовує функцію Execute Never (EN) архітектури ARM. Це дозволяє ASLR також як деякі частини пам'яті зупиняти атаки переповнення буфера, включаючи атаки повернення до libc.

6. Як було сказано вище, одне використання шифрування в iOS знаходиться в пам'яті Secure Enclave. Коли на пристрої iOS використовується код доступу, його вміст шифрується.

7. Двофакторна автентифікація – це опція в iOS, щоб переконатися, що якщо несанкціонована особа знає комбінацію ідентифікаторів Apple і пароля, вона не зможе отримати доступ до облікового запису.

8. iOS підтримує TLS як з API низького, так і з високого рівня для розробників. Коли Wi-Fi увімкнено, iOS використовує випадкову MAC-адресу, щоб ніхто не міг відслідковувати пристрій, нюхаючи бездротовий трафік.

Програмний додаток буде виконаний для ОС Android, адже вона є найбільш використовуваною. Проте за бажанням його можна адаптувати під платформу iOS. Для цього, перш за все, на машині Mac OS X повинні бути встановлені всі необхідні інструменти для розробки під Xamarin, і також повинен бути встановлений XCode. Бажано, щоб була встановлена остання версія XCode.

Для підключення до Mac OS Visual Studio застосовує SSH і компоненти Xamarion.iOS.

2.2 Вибір мови програмування

Перш ніж почати виконувати розробку додатка, треба визначитися з мовою програмування. Існує чимала кількість мов програмування, які

дозволяють виконувати мобільні додатки: Objective-C, Java, C#, інші. Проте треба враховувати що цільовою платформою буде Android. Саме тому треба обрати між Java та C#.

Java – об'єктно-орієнтована мова програмування. Була розроблена Sun Microsystems на початку 1990-х. Хоча в основному використовується для Інтернет-додатків, Java - це проста, ефективна мова загального призначення. Java спочатку була розроблена для вбудованих мережевих додатків, що працюють на кількох платформах. Це портативна, об'єктно-орієнтована, інтерпретована мова [4].

Java надзвичайно портативна. Розроблений додаток Java працюватиме однаково на будь-якому комп’ютері, незалежно від апаратних особливостей або операційної системи, якщо у ньому є інтерпретатор Java. Окрім мобільності, однією з ключових переваг Java є набір функцій безпеки, які захищають ПК, що працює на програмі Java, не лише від проблем, викликаних помилковим кодом, але й від шкідливих програм (наприклад, вірусів). Це можливо безпечно запустити аплет Java, завантажений з Інтернету, оскільки функції безпеки Java заважають цим типам аплетів отримати доступ до жорсткого диска ПК або мережевих з'єднань. Аплет типово невелика програма Java, яка вбудована в HTML-сторінку.

Java може вважатися як компільованою, так і інтерпретованою мовою, оскільки її вихідний код спочатку компілюється у двійковий байт-код. Цей байт-код працює на віртуальній машині Java (JVM), яка зазвичай є перекладачем на основі програмного забезпечення. Використання компільованого байт-коду дозволяє інтерпретатору (віртуальній машині) бути невеликим та ефективним (і майже таким же швидким, як процесор, що працює з власним, складеним кодом). Крім того, цей байт-код надає Java свою портативність: він буде працювати на будь-якому JVM, який правильно реалізований, незалежно від апаратного забезпечення чи конфігурації програмного забезпечення. Більшість веб-браузерів (таких як Microsoft Internet Explorer або Netscape Communicator) містять JVM для запуску аплетів Java [5].

Порівняно з C ++ (іншою об'єктно-орієнтованою мовою), код Java працює трохи повільніше (через JVM), але він більш портативний і має набагато кращі функції безпеки. Віртуальна машина забезпечує ізоляцію між ненадійною програмою Java та ПК, на якому працює програмне забезпечення. Синтаксис Java схожий на C ++, але мови зовсім різні. Наприклад, Java не дозволяє програмістам здійснювати перевантаження оператора в той час, як це робить C ++. Крім того, Java - це динамічна мова, де можна безпечно змінювати програму під час її запуску, тоді як C ++ не дозволяє. Це особливо важливо для мережевих додатків, які не можуть дозволити собі простоїв. Крім того, всі базові типи даних Java визначені заздалегідь і не залежать від платформи, тоді як деякі типи даних можуть змінюватися за допомогою платформи, що використовується в C або C ++ (наприклад, тип int).

Програми Java більш структуровані, ніж еквіваленти C ++. Усі функції (або методи Java) та виконувані оператори на Java повинні знаходитись у класі, тоді як C ++ дозволяє визначенням функцій та рядків коду існувати поза класами (як у програмах у стилі C). Глобальні дані та методи не можуть перебувати поза класом на Java, тоді як C ++ дозволяє це. Ці обмеження, хоч і громіздкі часом, допомагають підтримувати цілісність та безпеку програм Java і змушують їх бути повністю орієнтованими на об'єкти.

Ще однією ключовою особливістю Java є те, що це відкритий стандарт із загальнодоступним вихідним кодом. Sun Microsystems контролює мову Java та пов'язані з нею продукти, але ліберальна ліцензійна політика Sun сприяла тому, що Інтернет-спільнота сприйняла Java як стандарт. Можна вільно завантажувати всі інструменти, необхідні для розробки та запуску Java-апплетів та програм із Sun's Java.

Основні особливості мови Java перераховані нижче:

1. Незалежність платформи.
2. Повністю об'єктно-орієнтований, але простіший за C ++.
3. Динамічне інкрементальне навантаження та зв'язування.
4. Автоматичний GC.

5. Багатопотокові.

6. Систематичне іменування файлів класів, пакетів та вихідних файлів.

7. GUI та графічне програмування.

8. Підтримка веб-та мережевих додатків.

9. Інтернаціоналізація (програми в Unicode).

C# – це сучасна, об'єктно-орієнтована мова програмування, розроблена Microsoft та затверджена Європейською асоціацією виробників комп'ютерів (ECMA) та Міжнародною організацією зі стандартів (ISO) [6]. C# розроблений для загальної мовної інфраструктури (CLI), яка складається з виконуваного коду та середовища виконання, що дозволяє використовувати різні мови високого рівня на різних комп'ютерних платформах та архітектурах.

Наступні причини роблять C# широко використовуваною професійною мовою:

1. Це сучасна мова програмування загального призначення.

2. Орієнтовано на об’єкти.

3. Вона орієнтована на компоненти.

4. Навчитися легко.

5. Це структурована мова.

6. Він виробляє ефективні програми.

7. Його можна скласти на різних комп'ютерних платформах.

8. Це частина .Net Framework.

Хоча конструкції C# чітко дотримуються традиційних мов високого рівня, C і C ++ і є об'єктно-орієнтованою мовою програмування, він має велику схожість з Java, має численні сильні функції програмування, які роблять його приємним для ряду програмістів у всьому світі [7].

Кросплатформеність. .NET є, її переносять між платформами (з деякими обмеженнями). Наприклад, остання версія платформи на даний момент .NET Core підтримується на більшості сучасних ОС Windows, MacOS, Linux. Використовуючи різні технології на платформі .NET, можна розробляти

програми на мові C# для самих різних платформ: Windows, MacOS, Linux, Android, iOS, Tizen.

Потужна бібліотека класів. .NET представляє єдину для всіх підтримуваних мов бібліотеку класів. І який додаток не розроблюється на C# - текстовий редактор, чат або складний веб-сайт – так чи інакше буде використана бібліотека класів .NET.

Загальномовне середовище виконання CLR і базова бібліотека класів є основою для цілого стека технологій, які розробники можуть задіяти при побудові тих чи інших додатків. Наприклад, для роботи з базами даних в цьому стеку технологій призначена технологія ADO.NET і Entity Framework Core. Для побудови графічних додатків з багатим насиченим інтерфейсом - технологія WPF і UWP, для створення більш простих графічних додатків - Windows Forms. Для розробки мобільних додатків - Xamarin. Для створення веб-сайтів - ASP.NET і т.д.

Також ще слід відзначити таку особливість мови C# і фреймворка .NET, як автоматичне прибирання сміття. А це означає, що в більшості випадків не доводиться, на відміну від C ++, піклуватися про звільнення пам'яті. Вищезазначена загальномовного середовища CLR сама викличе збирач сміття і очистить пам'ять.

Варто відзначити, що .NET довгий час розвивався головним чином як платформа для Windows під назвою .NET Framework. В 2019 вищла остання версія цієї платформи - .NET Framework 4.8. Вона більше не розвивається.

З 2014 Microsoft став розвивати альтернативну платформу - .NET Core, яка вже призначалася для різних платформ і повинна була увібрати в себе всі можливості застарілого .NET Framework і додати нову функціональність. Тому слід розрізняти .NET Framework, який призначений переважно для Windows, і кросплатформенних .NET Core.

Також варто згадати про платформу Mono, яка була створена ще в 2004 році і представляла опенсорс-версію платформи .NET Framework для Linux і

MacOS. Використовуючи Mono, можна було створювати кросплатформенні додатки на C#.

Також наявна можливість для створення мобільних додатків – Xamarin.

Переваги використання Xamarin.Forms:

1. В процесі розробки створюється єдиний код для всіх платформ.

2. Xamarin надає прямий доступ до нативним API кожної платформи.

При створенні додатків можна використовувати платформу .NET і мову програмування C # (а також F #), яка є досить продуктивною, і в той же час ясною і простою для освоєння і застосування. Xamarin Forms підтримує кілька платформ. Основні платформи: Android, iOS, UWP, Tizen. Додаткові платформи в стані превью: MacOS, WPF, GTK # [8].

Для виконання додатку було обрано мову C#, а саме платформу Xamarin, адже вона доволі зручна при розробці мобільних додатків для операційної системи Android а також дає можливість виконувати один додаток для платформ iOS та Android одночасно.

2.3 Огляд середовища розробки

Офіційним середовищем розробки мобільних додатків мовою C# на платформі Xamarin є Visual Studio. Він містить емулятор, засоби відлагодження, профілювання пам'яті та швидкодії. Також доступні плагіни для редагування коду (наприклад ReSharper).

Перевагами Visual Studio є:

1. Наявність рекомендацій IntelliSense для швидкого і точного введення потрібних змінних при виникненні труднощів. Зберігається високий темп роботи незалежно від складності за рахунок швидкого переходу до будь-якого файлу, типу, елементу або символу. Можна використовувати значки лампочок, які рекомендують дії щодо поліпшення коду, наприклад пропонують перейменувати функцію або додати параметр.

2. CodeLens допомагає легко отримувати важливі аналітичні відомості, наприклад про внесені в код зміни, їх наслідки та модульному тестуванні

методів. Можна миттєво переглядати посилання, авторів, тести, журнал фіксацій і інші ключові дані.

3. Коли необхідно проаналізувати якусь помилку, Visual Studio дозволяє припиняти виконання коду за допомогою потрібних методів і точок зупину. Якщо розробник зіткнувся з непередбаченою зміною, є можливість повернутися до будь-якому рядку коду без необхідності перезапускати сеанс або відтворювати необхідний стан.

Visual Studio включає один або декілька з наступних компонентів:

1. Visual Basic .NET, а до його появи - Visual Basic.

2. Visual C++.

3. Visual C#.

4. Visual F# (входить до складу Visual Studio 2010).

5. Visual Studio Debugger.

Багато варіантів постачання також включають:

1. Microsoft SQL Server.

2. MSDE Visual Source Safe - файл-серверна система управління версіями.

У минулому до складу Visual Studio також входили продукти:

1. Visual InterDev.

2. Visual J++.

3. Visual J#.

4. Visual FoxPro.

5. Visual Source Safe – файл-серверна система управління версіями.

Нерідко розробники використовують плагін ReSharper для редагування коду.

Функції ReSharper

1. Аналіз якості коду.

Аналіз якості коду на льоту працює для C #, VB.NET, XAML, ASP.NET, ASP.NET MVC, JavaScript, TypeScript, CSS, HTML і XML. Відразу стане зрозуміло, якщо код потребує доопрацювання.

2. Усунення помилок і проблем в структурі коду

ReSharper не тільки попереджає про проблеми в коді, а й надає сотні швидких виправлень для автоматичного усунення виявлених проблем. У переважній більшості випадків можна вибрати найкраще швидке виправлення з великого списку варіантів.

3. Безпечна зміна кодової бази

Автоматичний рефакторинг коду в масштабі всього рішення дозволяє безпечно вносити зміни в кодову базу. ReSharper допоможе повернути успадкований код або навести порядок в структурі проекту.

4. Миттєвий пошук і навігація по всьому рішенню

Навігація і пошук по всьому рішенню працюють миттєво. Можна перейти до будь-якого файлу, типу або члену типу, а від будь-якого символу в коді - до його використання, базових і похідних символів і реалізацій.

5. Допомога при редагуванні коду

Безліч можливостей допомоги при редагуванні коду, включаючи розширені можливості IntelliSense, сотні миттєвих перетворень коду, автоматичний імпорт просторів імен, реорганізацію коду і зручний перегляд документації.

6. Рефакторинг.

Набір функцій ReSharper для рефакторингу значно перевищує за кількістю та зручністю використання вбудовані засоби середовища Visual Studio. У програмі забезпечено можливість відповідного покращення коду в мові C #, переважна більшість рефакторингів також доступні в VB.NET, деякі в JavaScript, XAML та інших підтримуваних мовах. Для здійснення рефакторингу необхідно лише навести курсор на відповідний блок коду. Деякі з підтримуваних рефакторингів наведено далі:

7. Зміна сигнатури методу.

Цей рефакторинг дозволяє змінити сигнатуру методу у такі способи:

1. Додавання, видалення, перейменування або зміна порядку параметрів.

2. Зміна типу результату, що повертається.

3. Зміна типів параметрів.

4. Перейменування методу.

Після зміни сигнатури методу, ReSharper оновлює всі його виклики, реалізації та перевизначення, підтримуючи працездатність коду.

8. Перетворення непорожнього методу до властивості і навпаки.

Існує можливість перетворення властивості до пари методів за допомогою програми ReSharper.

9. Відповідність стандартам оформлення коду

Можливість налаштування стилю коду і форматування разом з тонким налаштуванням параметрів для окремих мов допомагають позбавлятися від невикористаного коду і створювати спільні стандарти написання коду для всієї команди.

10. Автоматична генерація програмного коду.

В ReSharper існує можливість використання методу, властивості, поля змінної або навіть класу, перш ніж відповідні елементи будуть оголошені в програмному коді. ReSharper спритно запропонує виправити та доповнити створені автоматично блоки для таких структурних елементів. Наприклад, при автоматичному створенні методу з використанням ReSharper, необхідно додатково визначити тип результату що повертається, а також типи його параметрів.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

Першим кроком у розробці будь-якого додатку є створення різних допоміжних діаграм, які спрощують весь процес розробки. Для створення Android-додатку буде розроблено UML-діаграму використання, основне призначення якої полягає у відображенні процесу користування додатком.

3.1 Проектування додатку

Перш ніж розпочати розробку додатку, треба спроектувати архітектуру та продумати дизайн. Для відображення архітектури зручно використовувати UML – уніфіковану мова моделювання, яка використовується у парадигмі об'єктно-орієнтованого програмування та є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML може бути застосовано на всіх етапах життєвого циклу аналізу бізнес-систем і розробки прикладних програм [9].

У загальному випадку діаграма варіантів використання буде мати наступний вигляд:

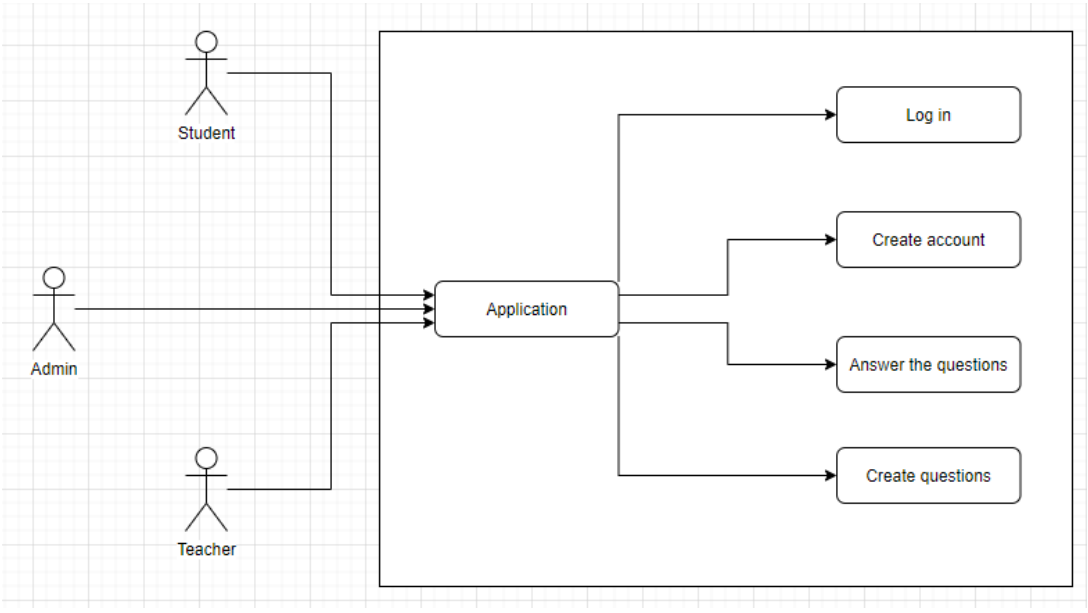

Рисунок 3.1 – UML-діаграма варіантів використання

Спочатку користувач повинен буде завантажити додаток з PlayMarket.

Мобільний додаток, який буде виконаний в рамках даного випускної кваліфікаційної роботи, міститиме наступний функціонал:

1. Авторизація (рис. 3.2а) і реєстрація (рис. 3.2б) користувача через форму.

а б

Рисунок 3.2 – Форма авторизації (а) і реєстрація нового користувача(б)

2. Можливість створення нових областей для тестування.

Рисунок 3.3 – Форма створення нової області тестування

3. Можливість створення нових запитань (тестів) для існуючих областей.

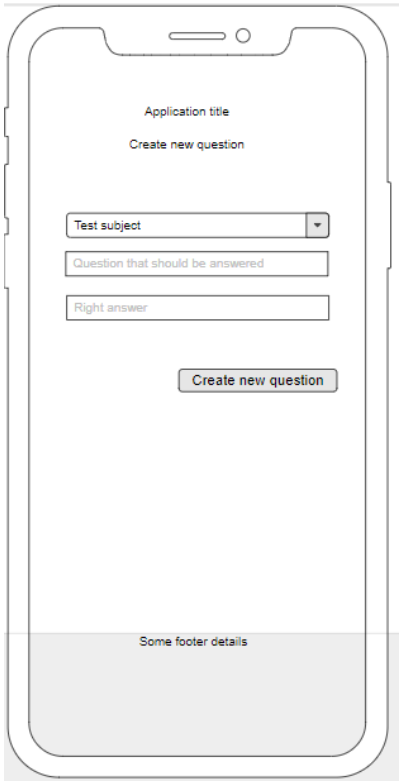

Рисунок 3.4 – Форма створення нових запитань

4. Можливість проходження тестів.

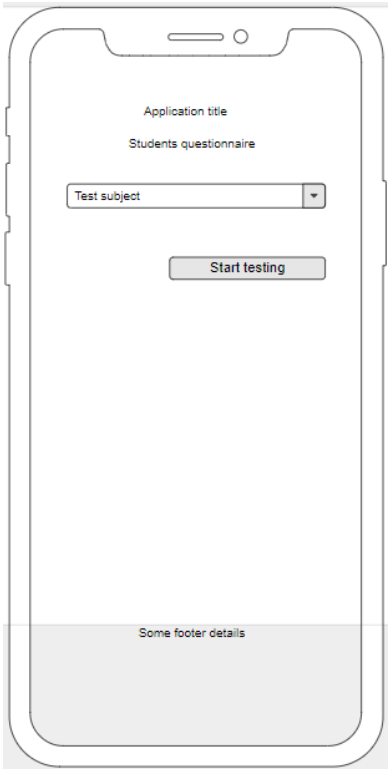

Рисунок 3.5 – Вибір області тестування

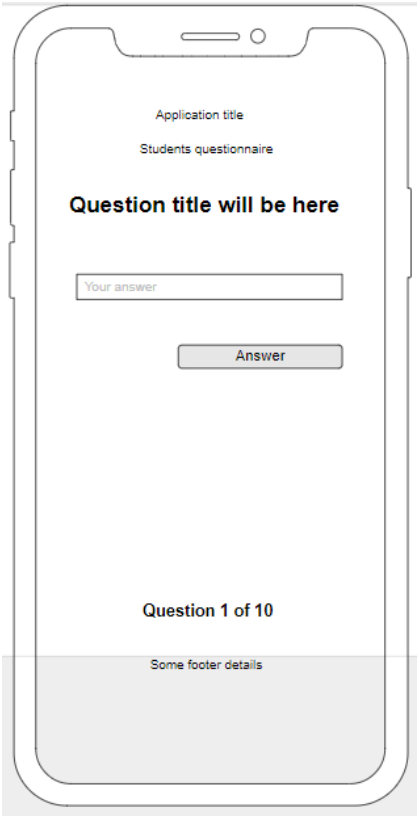

Рисунок 3.6 – Проходження тестування

3.2 Проектування бази даних

Для зберігання даних користувача буде розроблена база даних. Вона є невід'ємною частиною розробки мобільного додатку.

У загальному випадку модель бази даних матиме наступний вигляд:

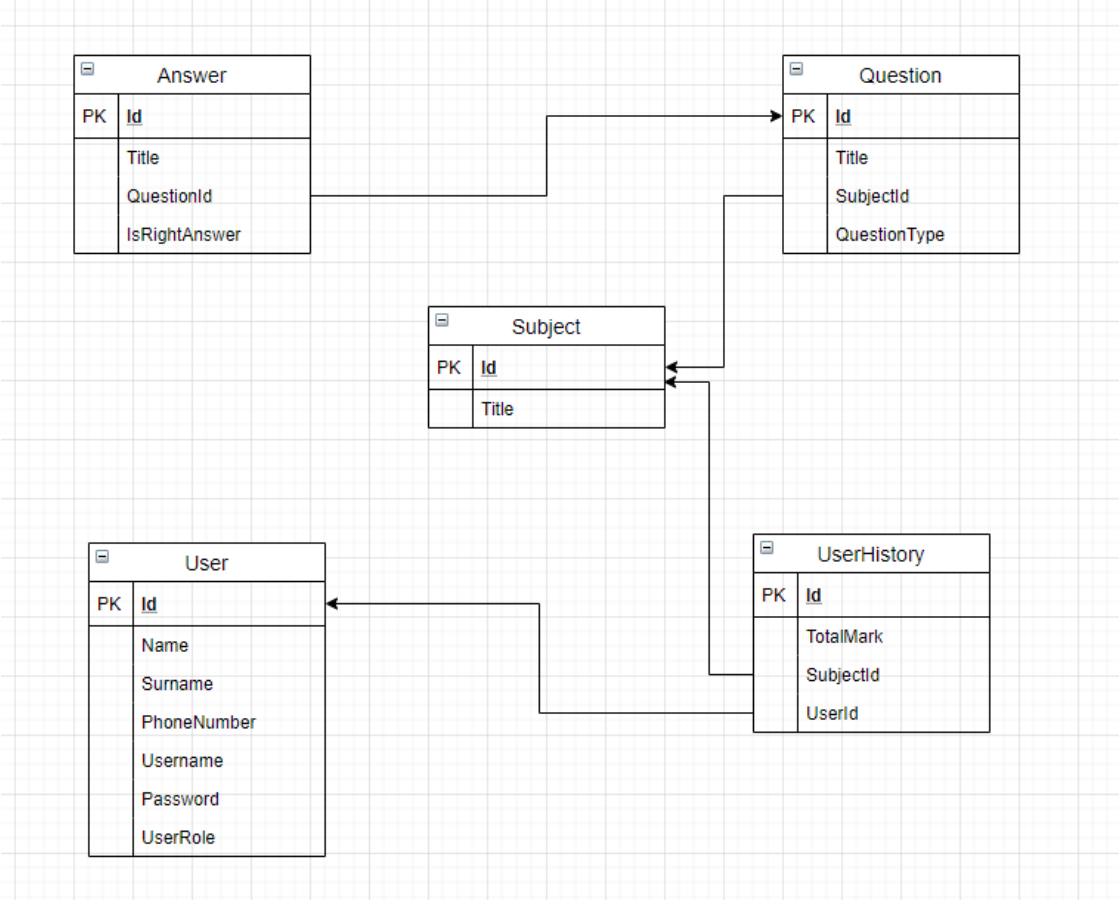

Рисунок 3.7 – Модель бази даних

3.3 Структура проекту

Основна структура проекту виглядає наступним чином (рис. 3.8):

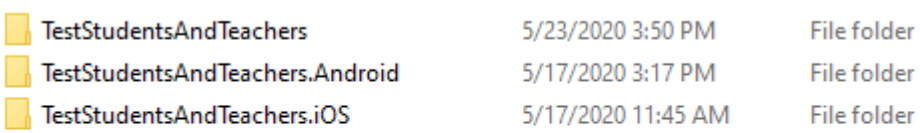

Рисунок 3.8 – Основна структура проекту

Основні файли, які присутні в структурі базового проекту:

1. Папка TestStudentsAndTeachers: містить основну логіку та програмний код.

2. Папка TestStudentsAndTeachers.Android: містить код, властивий лише для Android-додатку.

3. Папка TestStudentsAndTeachers.iOS: містить код, властивий лише для iOS-додатку.

Оскільки основна логіка міститься в файлах в папці TestStudentsAndTeachers, то далі буде розглянуто її структуру (рис.3.9).

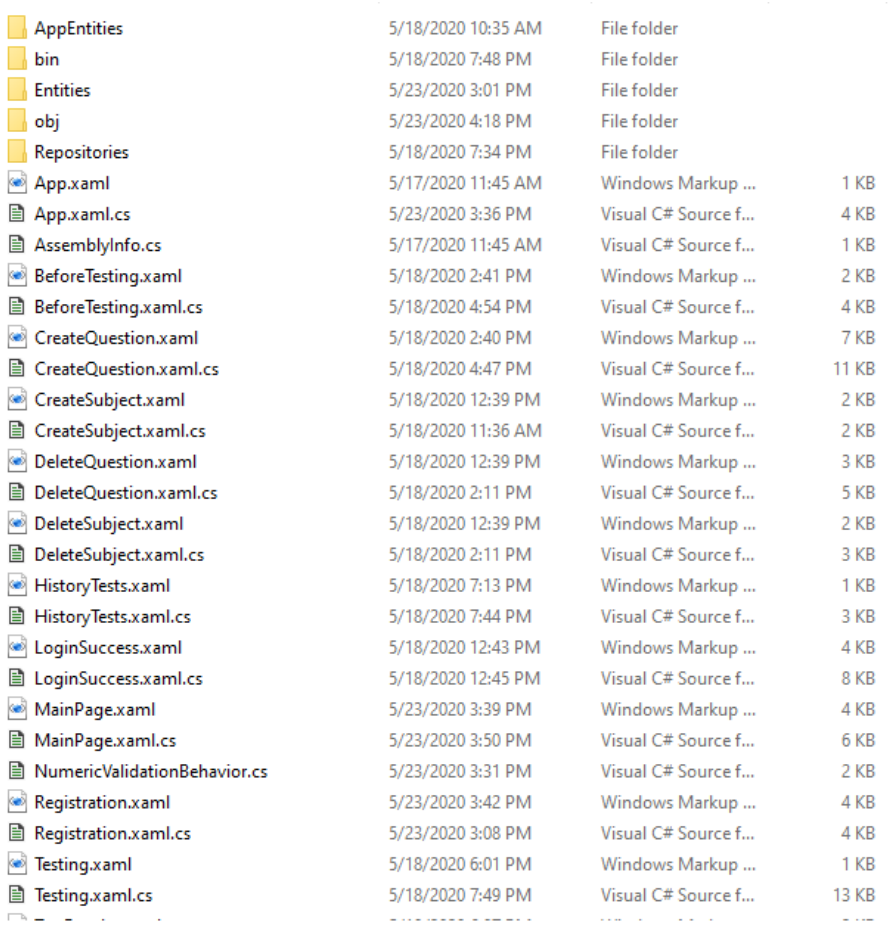

Рисунок 3.9 – Вміст папки TestStudentsAndTeachers

У цій папці наявна велика кількість файлів. Кожна сторінка додатку представлена двома файлами з розширенням .xaml (інтерфейс сторінки: поля вводу, кнопки, т.д.) та .xaml.cs (класс C#, який представляє код сторінка).

Також є папки bin та obj, які є стандартними для всіх проектів мови C# (в них міститься скомпільований код, специфічні ресурси додатку).

Також для виконання завдання треба було налаштувати роботу з базою даних. Для роботи з базою даних в C# необхідна наявність класів-моделей (в даному випадку вони були створені в папці Entities) та класів-репозиторіїв для за допомогою яких і відбувається обмін даними з базою (в даному випадку вони були створені в папці Repositories).

І також наявна папка AppEntities, в якій розміщено додаткові типи даних, властиві даному додатку.

3.4 Опис основних функцій та процедур

Як було сказано, кожен модуль складається з двох частин: xaml файла розмітки та відповідного йому класу. Далі будуть описані саме класи додатку. Отже, основні частини коду, які були використані при написанні додатку:

1. Для того, щоб з одного модуля викликати інший, треба використати вбудований клас Navigation.

```
Navigation.PushAsync(new LoginSuccess(), true);
Navigation.RemovePage(this);
```

2. Для того, щоб отримати значення поля, треба використати його ім’я, задане в xaml-файлі

```
var login = Login.Text;
```

3. Для відображення повідомлення про помилку треба використовувати метод DisplayAlert.

```
if (string.IsNullOrEmpty(login))
{
await DisplayAlert("Помилка", "Поле вводу логіну обов'язкове для заповнення", "ОК");
Login.Focus();
return;
}
```

4. Для роботи з смс-повідомленнями було підключено бібліотеку Plugin.Messaging.

```
var smsMessenger = CrossMessaging.Current.SmsMessenger;
if (smsMessenger.CanSendSmsInBackground)
```

```
{
smsMessenger.SendSmsInBackground("+" + App.SessionUser.PhoneNumber,
"Ваш верифікаційний код: " + App.SentCode);
}
```

5. Для того, щоб зробити елемент інтерфейсу недоступним або невидимим, треба скористатися відповідними функціями isEnabled/isVisible.

```
RegisterUserButton.IsEnabled = true;
ConfirmNumber.IsVisible = false;
```

Інші частини коду представлені базовим синтаксисом мови C#: виконання арифметичних операцій, порівняння отриманих значень.

ВИСНОВКИ

Основними завданнями, які були визначені в ході проходження дипломного проектування, є наступні:

1. Огляд існуючих мов програмування та вибір необхідної для створення майбутнього додатка мови програмування, операційної системи та середовища розробки;

2. Розробка діаграми варіантів використання UML.

3. Розробка математичної моделі майбутнього Android-додатку.

4. Розробка додатку обраною мовою програмування в обраному середовищі розробки.

В ході проходження дипломного проектування було доведено необхідність розвитку технологій та автоматизації в області освіти, було сформовано вимоги до системи, серед них вимоги щодо функціоналу, інтерфейсу та програмної реалізації та визначено основні принципи розробки даного Android-додатку.

Для більшого розуміння суті програми було створено UML-діаграму варіантів використання.

По завершенню проходження дипломного проектування було розроблено програмний продукт «Інформаційна система тестування із ідентифікацією та авторизацією студентів». Цей додаток виконує весь функціонал, описаний в пункті «Постановка задачі». Також реалізовано перевірку полів на наявність значень, щоб додаток не видав системну помилку.

СПИСОК ЛІТЕРАТУРИ

1. Інформаційна система [Електронний ресурс] – Режим доступу до ресурсу: https://www.britannica.com/topic/information-system
2. Android [Електронний ресурс] – Режим доступу до ресурсу: https:// uk.wikipedia.org/wiki/Android - Назва з екрану.
3. iOS [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/IOS - Назва з екрану.
4. Кей С. Хорстманн, Гари Корнелл (2013). Java. Библиотека профессионала, том 1. Основы. 9-е издание. «Вільямс». ISBN 978-5-8459-1869-7.
5. Фрэд Лонг та ін. (2014). Руководство для программиста на Java: 75 рекомендаций по написанию надежных и защищенных программ. «Вільямс». ISBN 978-5-8459-1897-0.
6. C# [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/C_Sharp - Назва з екрану.
7. Вступ в мову C# [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com/sharp/tutorial - Назва з екрану.
8. Xamarin Forms [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com/sharp/xamarin/ - Назва з екрану.
9. Грейди Буч, Джеймс Рамбо, Айвар Джекобсон. Язык UML. Руководство пользователя = The Unified Modeling Language user guide. — 2-е изд. — М., СПб.: ДМК Пресс, Питер, 2004. — 432 с. — ISBN 5-94074-260-2.

ДОДАТКИ

Зміст файлу MainPage.xaml.cs

```
using System;
using System.ComponentModel;
using System.Linq;
using Plugin.Messaging;
using TestStudentsAndTeachers.AppEntities;

namespace TestStudentsAndTeachers
{
[DesignTimeVisible(false)]
public partial class MainPage
{
public MainPage()
{
if (App.SessionUser != null)
{
Navigation.PushAsync(new LoginSuccess(), true);
Navigation.RemovePage(this);
}
else
{
InitializeComponent();
ConfirmNumber.IsVisible = false;
ConfirmNumberButton.IsVisible = false;
ReloginButton.IsVisible = false;
}
}

private async void LoginToApp(object sender, EventArgs e)
{
var login = Login.Text;
var password = Password.Text;

if (string.IsNullOrEmpty(login))
{
await DisplayAlert("Помилка", "Поле вводу логіну обов'язкове для
заповнення", "ОК");
Login.Focus();
return;
}

if (string.IsNullOrEmpty(password))
{
await DisplayAlert("Помилка", "Поле вводу паролю обов'язкове для
заповнення", "ОК");
Password.Focus();
return;
}

var foundUser = App.UserRepository.GetItems().FirstOrDefault(u =>
u.Username == login && u.Password == password);

if (foundUser == null)
```

```
{
await DisplayAlert("Помилка входу", "Не знайдено користувача за
введеними логіном та паролем", "ОК");
return;
}

ConfirmNumber.IsVisible = true;
ConfirmNumberButton.IsVisible = true;
ReloginButton.IsVisible = true;
Login.IsEnabled = false;
Password.IsEnabled = false;
LoginToAppButton.IsEnabled = false;
RegisterUserButton.IsEnabled = false;

App.SessionUser = new AppSessionUser
{
UserRole = foundUser.UserRole,
Password = foundUser.Password,
Username = foundUser.Username,
PhoneNumber = foundUser.PhoneNumber,
Name = foundUser.Name,
Id = foundUser.Id,
Surname = foundUser.Surname
};

App.SentCode = new Random().Next(1000, 9999);
var smsMessenger = CrossMessaging.Current.SmsMessenger;
if (smsMessenger.CanSendSmsInBackground)
{
smsMessenger.SendSmsInBackground("+" + App.SessionUser.PhoneNumber,
"Ваш верифікаційний код: " + App.SentCode);
}
else
{
await Navigation.PushAsync(new LoginSuccess(), true);
Navigation.RemovePage(this);

Login.IsEnabled = true;
Password.IsEnabled = true;
LoginToAppButton.IsEnabled = true;
RegisterUserButton.IsEnabled = true;

ConfirmNumber.IsVisible = false;
ConfirmNumberButton.IsVisible = false;
ReloginButton.IsVisible = false;

Login.Text = "";
Password.Text = "";
}
}

private async void ConfirmNumberEvent(object sender, EventArgs e)
{
if (int.Parse(ConfirmNumber.Text) == App.SentCode)
{
await Navigation.PushAsync(new LoginSuccess(), true);
Navigation.RemovePage(this);
```

```
Login.IsEnabled = true;
Password.IsEnabled = true;
LoginToAppButton.IsEnabled = true;
RegisterUserButton.IsEnabled = true;

ConfirmNumber.IsVisible = false;
ConfirmNumberButton.IsVisible = false;
ReloginButton.IsVisible = false;

Login.Text = "";
Password.Text = "";
}
else
{
await DisplayAlert("Помилка входу", "Ви ввели невірний
верифікаційний код", "ОК");
}
}

private void ReloginButtonEvent(object sender, EventArgs e)
{
ConfirmNumber.IsVisible = false;
ConfirmNumberButton.IsVisible = false;
ReloginButton.IsVisible = false;
Login.IsEnabled = true;
Password.IsEnabled = true;
LoginToAppButton.IsEnabled = true;
RegisterUserButton.IsEnabled = true;
}

private async void RegisterUser(object sender, EventArgs e)
{
LoginToAppButton.IsEnabled = false;
RegisterUserButton.IsEnabled = false;

await Navigation.PushAsync(new Registration(), true);
Login.Text = "";
Password.Text = "";

LoginToAppButton.IsEnabled = true;
RegisterUserButton.IsEnabled = true;
}
}
}
```

Зміст файлу MainPage.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
xmlns:d="http://xamarin.com/schemas/2014/forms/design"
xmlns:mc="http://schemas.openxmlformats.org/markup-
compatibility/2006"
mc:Ignorable="d"
x:Class="TestStudentsAndTeachers.MainPage"
Title="Тестування студентів та викладачів">

<StackLayout>
```

```
<RelativeLayout>
<Label x:Name="WelcomeMessage"
HorizontalTextAlignment="Center"
FontSize="Medium"
Margin="0, 10, 0, 0"
Text="Вітаємо. Цей додаток призначений для проходження студентами
та викладачами тестування. Для початку роботи виконайте вхід або
зареєструйтесь"
/>
<StackLayout RelativeLayout.YConstraint="{ConstraintExpression
Type=RelativeToView, ElementName=WelcomeMessage, Property=Y, Fac-
tor=1, Constant=130}" VerticalOptions="Center">
<Entry Placeholder="Введіть логін *"
HorizontalOptions="CenterAndExpand"
WidthRequest="500"
x:Name="Login"
Margin="0, 10"
/>
<Entry Placeholder="Введіть пароль *"
IsPassword="True"
HorizontalOptions="CenterAndExpand"
WidthRequest="500"
x:Name="Password"
Margin="0, 10"
/>
<RelativeLayout Margin="0, 10, 0, 0">
<AbsoluteLayout>
<Button Text="Зареєструвати користувача"
AbsoluteLayout.LayoutBounds="5, 5, 200, 70"
Clicked="RegisterUser"
x:Name="RegisterUserButton"
/>
<Button Text="Вхід"
AbsoluteLayout.LayoutBounds="215, 5, 130, 70"
Clicked="LoginToApp"
x:Name="LoginToAppButton"
/>
</AbsoluteLayout>
</RelativeLayout>
<Entry Placeholder="Введіть код з смс *"
IsPassword="True"
HorizontalOptions="CenterAndExpand"
WidthRequest="500"
x:Name="ConfirmNumber"
Keyboard="Numeric"
Margin="0, 10"
/>
<RelativeLayout Margin="0, 10, 0, 0">
<AbsoluteLayout>
<Button Text="Змінити логін та пароль"
AbsoluteLayout.LayoutBounds="5, 5, 200, 70"
Clicked="ReloginButtonEvent"
x:Name="ReloginButton"
/>
<Button Text="Підтвердити"
AbsoluteLayout.LayoutBounds="215, 5, 130, 70"
Clicked="ConfirmNumberEvent"
x:Name="ConfirmNumberButton"
```

```
/>
</AbsoluteLayout>
</RelativeLayout>
</StackLayout>
</RelativeLayout>
</StackLayout>
</ContentPage>
```