**НПУ**

**Кафедра ІПЗ**

**Дипломна робота бакалавра**

**Ст. гр. 42ІПЗ**

**Логвиненка Володимира Вікторовича**

**„Розробка додатку для вивчення української мови”**

**42ІПЗ Логвиненко Володимир Вікторович**

**КОМПАКТ – ДИСК з електронними матеріалами**

**(WriteCompetently - додаток для вивчення української мови)**

**Міністерство освіти і науки України**

**Національний педагогічний університет імені М.П. Драгоманова**

Факультет інформатики

Кафедра Програмної інженерії

**ДИПЛОМНА РОБОТА БАКАЛАВРА**  
пояснювальна записка

050103. 36ІПЗ17 ІПЗ

Розробка додатку для вивчення української мови

Студент гр. 42ІПЗ Логвиненко В.В

Керівник роботи Мучник М.М

Зав. кафедри, кандидат тех. наук, доцент

Допускається до захисту **Мучник М.М.**  
**Зав. кафедри, кандидат тех. наук, доцент**

2019 р.

*Національний педагогічний університет імені М.П.Драгоманова*

|  |  |
| --- | --- |
| Факультет *інформатики.* | Кафедра *програмної інженерії* |
| Спеціальність *Програмна інженерія* | |
|  | ЗАТВЕРДЖУЮ:  “\_\_\_\_”\_\_\_\_\_\_\_\_20\_\_р \_\_\_\_\_\_\_\_\_\_  Зав. кафедри, доцент М.М.Мучник |

ЗАВДАННЯ

НА ДИПЛОМНУ РОБОТУ БАКАЛАВРА СТУДЕНТОВІ

***Логвиненку Володимиру Вікторовичу***

1. Тема проекту (роботи) “*Розробка програмного забещпечення для вивчення української мови”* затверджена наказом університету №\_\_\_\_\_\_\_\_ від „\_\_\_\_”\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ р.
2. Термін здачі студентом закінченої роботи „\_\_\_\_”\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ р.
3. Вихідні дані до проекту (роботи): *Використовувати ОС Windows, СУБД SQL Server, середовище об'єктно-орієнтованого проектування Visual Studio Code.*
4. Зміст пояснювальної записки *(перелік питань, що їх належить розробити) мета роботи, аналіз проблемної області і постановка задачі, опис об'єктних моделей, використовувані методи та алгоритми, структура бази даних, опис розробленої програмної системи, захист інформації (за необхідністю), аналіз можливих застосувань.*
5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень) *Мета завдання, обґрунтування доцільності розроблення, постановка задачі, об'єктна модель системи, базові моделі, методи й алгоритми, структура бази даних, структурно-логічна схема взаємодії даних, план захисту інформації (за необхідністю), інтерфейс програмної системи, результати тестування програмної системи.*
6. Консультанти з проекту (роботи) із зазначенням розділів проекту, що їх стосуються

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Консультант | Підпис, дата | |
| Завдання видав | Завдання прийняв |
| Спецчастина | Мучник М.М. |  |  |

**Календарний план**

|  |  |  |  |
| --- | --- | --- | --- |
| Но-мер | Назва етапів дипломної роботи | Термін виконання етапів проекту (роботи) | Примітка: інформація про виконання |
| 1 | Об'єктний аналіз поставленої задачі |  |  |
| 2 | Розробка моделі взаємодії даних |  |  |
| 3 | Розробка структури зберігання даних |  |  |
| 4 | Створення коду програми |  |  |
| 5 | Тестування і налагодження програми |  |  |
| 6 | Підготовка пояснювальної записки. |  |  |
|  | Спецчастина |  |  |
| 7 | Підготовка презентації та доповіді |  |  |
| 8 | Попередній захист |  |  |
| 9 | Нормоконтроль, рецензування |  |  |
| 10 | Занесення диплома в електронний архів |  |  |
| 11 | Допуск до захисту у зав. кафедри |  |  |

Дата видачі завдання “\_\_\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ р.

Керівник зав. кафедри, кандидат тех. наук, доцент Мучник М.М.

Завдання прийняв до виконання Логвиненко В.В.

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до дипломної роботи: 45 с., 33 рис., ? додатки, 18 джерел.

Метою роботи є розробка додатку для вивчення англійської мови.

Методи розробки базуються на технології Java та платформи Android, сервер бази даних SQLite.

У результаті роботи здійснена програмна реалізація системи для вивчення англійської мови шляхом виконання тестових завдань

ВИВЧЕННЯ АНГЛІЙСЬКОЇ, МОБІЛЬНИЙ ДОДАТОК, АНГЛІЙСЬКА МОВА, ТЕРМІНИ, ТЕСТИ, SQLite, JAVA, ANDROID, IT

The purpose of the work is to develop an application for learning English.

Development methods are based on Java technology and the Android platform, the SQLite database server.

As a result of the work the program implementation of the system for studying English language through the implementation of test tasks

ENGLISH STUDYING, MOBILE APPLICATION, ENGLISH LANGUAGE, TERMS, TESTS, SQLITES, JAVA, ANDROID

ЗМІСТ

Вступ

1. Відомості про об`єкт розробки
2. Опис методів та засобів вирішення задачі
   1. Запропоновані методи та засоби вирішення задачі
   2. Архітектуру програмного забезпечення
   3. Обрані програмні засоби
      1. Середовище розробки
      2. Мови програмування
      3. База даних
   4. Структура даних
   5. Особливості реалізації
   6. Особливості застосування
3. Аналіз результатів

Висновки

Список літеретури

Вступ

Кожна людина, незалежно від національності та країни проживання, має свою рідну мову, яка стає не лише важливим способом спілкування між людьми, але й аспектом для пізнання оточуючого світу. Це та мова, з якою людина пов’язана з самого народження, вона на ньому говорить свої перші слова, декламує вірші, читає книги, пише та думає. Для україномовного населення рідною є і буде українська мова, найспівучіша із мов.

Цей предмет вивчають в школах, починаючи з першого класу, та продовжують її вдосконалювати протягом всього свого життя, так як будь-які знання передаються, в першу чергу, мовним шляхом.

Вільне володіння українською літературною мовою в усній чи письмовій формах і є основною задачею навчання школярів.

Діти повинні вчити свою рідну мову, адже вона слугує ключем для розвитку особистості, його освіченості та пізнанню, без мовних знань вони не зможуть повноцінно приймати участь в суспільному житті та розвитку вітчизняного мистецтва та культури.

Українська мова нам необхідна для спілкування в людському суспільстві і щоб це суспільство нас розуміло, ми повинні розмовляти грамотно, правильно висловлювати свої думки, а для цього необхідно вчитися. Наше спілкування не обмежується лише розмовами, ми навчаємося читанню, правопису, переказуємо та пишемо твори. Тому українську мову необхідно вдосконалювати та продовжувати вчити все життя, і весь час ми будемо дізнаватися про неї щось нове.

* Кожна людина повинна знати свою рідну мову.
* Знання цієї дисципліни робить наш народ культурним та освіченим.
* Дитина, що володіє виразною та грамотною мовою, може багато чого достягти в своєму житті.

Українська мова продовжує розвиватися та змінюватися с плином часу. Вона запозичує слова з іноземних мов, поповнюється різноманітним сленгом, але головне, нам необхідно зберегти справжню українську літературну мову. Ось для чого необхідно мову вивчати та вдосконалювати.

*Метою розробки* є створення і реалізацію додатку для вивчення української мови.

Завдання розробки є :

1) аналіз відомих аналогів

2) проектування програмного засобу

3) розробку програмного засобу та супровідних документацій

4) тестування програмного продукту

Для створення інтерактивного курсу для вивчення української мови використовуються такі програмні засоби: Visual Studio Code – середовище розробки frontend частини курсу;

Visual Studio – середовище розробки backend частини курсу; C# – об’єктно-орієнтована мова програмування для написання додатків в середовищі Visual Studio; HTML – мова розмітки WEB – сторінок; CSS – спеціальна мова, що використовується для опису зовнішнього вигляду сторінок, написаних мовами розмітки даних; JavaScript - динамічна, об'єктно-орієнтована прототипна мова програмування; SQL Server – база даних.

Даний веб-ресурс буде розроблений для того щоб кожен бажаючий міг здобути потрібні знання з української мови, дізнатися визначення незнайомих йому термінів та закріпити свої знання з української мови.

Програмне забезпечення повинне навчити користувача правилам української мови шляхом його постійного тестування.

Система для вивчення української мови створюється з метою використання її людьми які хочуть вивчити українську мову, або уже володіють нею і хочуть поліпшити свої знання. Веб-ресурс в подальшому буде розвиватися і розширюватися новими модулями.

1. Відомості про об`єкт розробки

На сьогоднішній день існує досить багато способів вивчення української мови: деякі досить добре вивчають мову в учбових закладах, і цього їм вистачає на все життя, деякі наймають репетиторів для того щоб вивчити або покращити знання мови, а деякі просто встановлюють на свій мобільний телефон додаток, або заходять на веб-ресурс яким вини можуть скористатися при будь-якій нагоді. Мобільний телефон ми завжди тримаємо при собі, хоч дома, хоч на прогулянці, хоч на роботі чи навчанні. Досить велику частину часу ми проводимо в дорозі і для того щоб використати цей час з користю ми можемо просто відкрити веб-ресурс на телефоні і вивчати українську. Тому і мій вибір впав на веб-ресурс адже на даний момент це найбільш розповсюджена платформа.

Досить великою проблемою є те що на ринку веб-ресурсів водночас досить багато додатків для вивчення української мови проте вони не бездоганні, саме тому я і вирішив зробити свій веб-ресурс по вивченню української мови.

На веб просторі є наступні веб-ресурси:

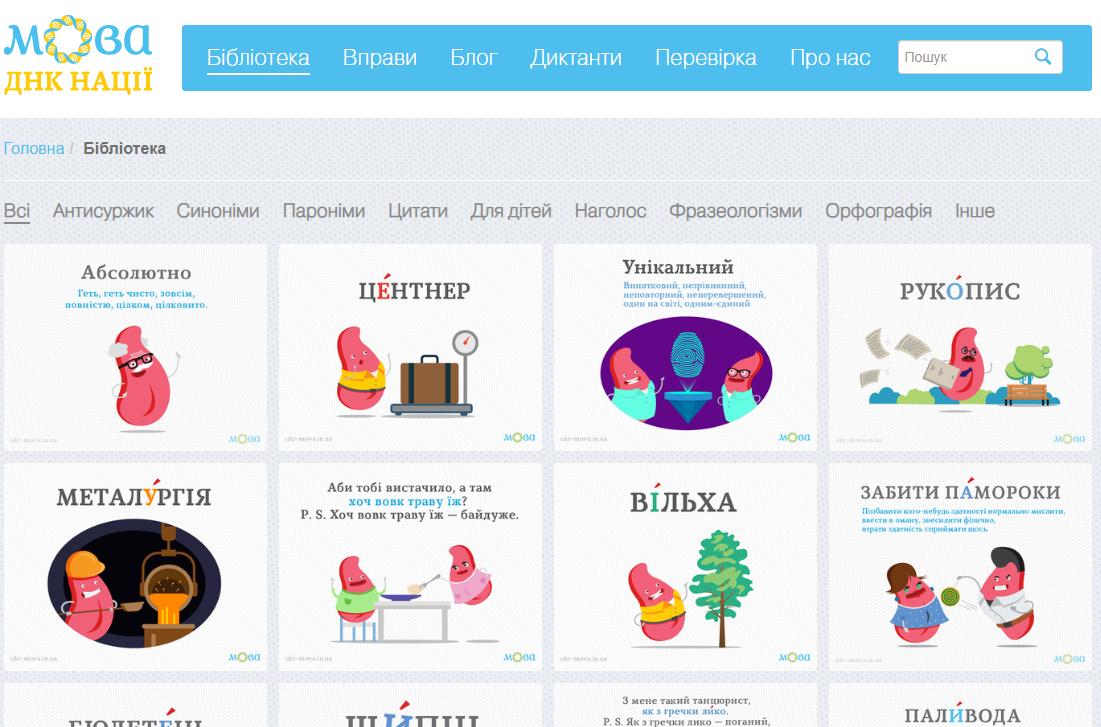


Рисунок 1.1 — вивчення української мови з ukr-mova.in.ua



Рисунок 1.2 — Тренажер з правопису української мови

І це лиш найпопулярніші додатки. Всі ці додатки є досить зручними для вивчення української, вони підійдуть майже для всіх, проте є і певні недоліки, які заважають роботі з даними ресурсами.

Було проведено дослідження ринку додатків які призначені для вивчення української мови, і як було з’ясовано на даний момент немає підходящого додатку який би задовольнив всі потреби, так як додатки надають лише теорію або лише практику а додатку який поетапно дає теорію та закріплює все це практикою, немає.

Підсумовуючи ми можемо зробити висновок, що додатків по «вивченню української мови» багато проте немає жодного додатку в якому досить непогано поєднано теорію з практикою. Тому і порівнянь з іншими додатками в плані різних специфікацій ми не можемо зробити.

2. Опис методів та засобів вирішення задачі

2.1. Запропоновані методи та засоби вирішення задачі

Проаналізувавши ринок було виявлено що на даний момент відсутні додатки які б задовольняли всі потреби, а саме:

* Навчити людину термінології на українській мові;
* Провести тестування користувача по цим термінам;
* Навчити людину правил української мови
* Провести тестування по даним правилам
* Ведення статистики по проходженню;
* Створення декількох профілів для проходження декільком користувачам на одному пристрої.
* Україномовний інтерфейс

На основі вище перерахованих потреб було вирішено створити даний програмний засіб.

Для вирішення поставленої задачі було запропоновано такі варіанти форматів додатків:

* Створення прикладного додатку для персонального комп’ютера;
* Створення веб-додатку;
* Створення мобільного додатку.

Суттю даного програмного продукту є те що користувач міг користуватися додатком в вільний час. Дана програма призначена для самоосвіти, а зазвичай у людини яка працює чи навчається не досить багато вільного часу і тому було вирішено зробити веб-додаток. Головною перевагою такого додатку є те що він «веб-додаток» тобто користувач може використати в будь-який момент на будь якому пристрої який підключений до мережі, а це саме те що було потрібно для продуктивного опрацювання матеріалу з додатку.

Враховуючи все вище сказане можна зробити висновок що хоч додаток для персонального комп’ютера працює найшвидше, додаток для мобільного пристрою зручніший у використанні на телефоні, проте найкращим вибором є веб-додаток так як доступ до нього можна здійснити з будь-якого пристрою (телефон, комп’ютер, планшет, телевізор…).

2.2 Архітектуру програмного забезпечення

Для даного програмного засобу було вирішено обрати архітектуру модені MVC.

**Model-View-Controller** (MVC, «Модель-Представлення-Контролер», «Модель-Вид-Контролер») - схема поділу даних програми, призначеного для користувача інтерфейсу і керуючої логіки на три окремих компоненти: модель, уявлення і контролер - таким чином, що модифікація кожного компонента може здійснюватися незалежно.

* Модель (Model) надає дані і реагує на команди контролера, змінюючи свій стан.
* Представлення (View) відповідає за відображення даних моделі користувачеві, реагуючи на зміни моделі.
* Контролер (Controller) інтерпретує дії користувача, сповіщаючи модель про необхідність змін.

Основна мета застосування цієї концепції полягає в відділенні бізнес-логіки (моделі) від її візуалізації (уявлення, виду). За рахунок такого поділу підвищується можливість повторного використання коду. Найбільш корисне застосування даної концепції в тих випадках, коли користувач повинен бачити ті ж самі дані одночасно в різних контекстах і / або з різних точок зору. Зокрема, виконуються наступні завдання:

До однієї моделі можна приєднати кілька видів, при цьому не зачіпаючи реалізацію моделі. Наприклад, деякі дані можуть бути одночасно представлені у вигляді електронної таблиці, гістограми і кругової діаграми;

Не торкаючись реалізацію видів, можна змінити реакції на дії користувача (натискання мишею на кнопці, введення даних) - для цього досить використовувати інший контролер;

Ряд розробників спеціалізується тільки в одній з областей: або розробляють графічний інтерфейс, або розробляють бізнес-логіку. Тому можливо добитися того, що програмісти, які займаються розробкою бізнес-логіки (моделі), взагалі не будуть обізнані про те, яке уявлення буде використовуватися.

**Модель**

Модель надає дані і методи роботи з ними: запити до бази даних, перевірка на коректність. Модель не залежить від уявлення (не знає як дані візуалізувати) і контролера (не має точок взаємодії з користувачем) просто надаючи доступ до даних і управління ними.

Модель будується таким чином, щоб відповідати на запити, змінюючи свій стан, при цьому може бути вбудовано повідомлення «спостерігачів».

Модель, за рахунок незалежності від візуального представлення, може мати кілька різних уявлень для однієї «моделі».

**Представлення**

Представлення відповідає за отримання необхідних даних з моделі і відправляє їх користувачеві. Подання і не виконує жодних введені дані користувача.

Представлення може впливати на стан моделі, повідомляючи моделі про це.

**Контролер**

Контролер забезпечує «зв'язок» між користувачем і системою. Контролює і направляє дані від користувача до системи і навпаки. Використовує модель і уявлення для реалізації необхідного дії.

2.3 Обрані програмні засоби

2.3.1 Середовище розробки

Інтегроване середовище розробки, ІСP (англ. Integrated development environment - IDE), також єдине середовище розробки, ЕСР - комплекс програмних засобів, який використовується програмістами для розробки програмного забезпечення (ПО).

Середовище розробки включає в себе:

* текстовий редактор,
* компілятор і / або інтерпретатор,
* засоби автоматизації збирання,
* відладчик.

Іноді містить також засоби для інтеграції з системами управління версіями і різноманітні інструменти для спрощення конструювання графічного інтерфейсу користувача. Багато сучасні середовища розробки також включають браузер класів, інспектор об'єктів і діаграму ієрархії класів - для використання при об'єктно-орієнтованої розробки ПЗ. ІСР зазвичай призначені для декількох мов програмування - такі як IntelliJ IDEA, NetBeans, Eclipse, Qt Creator, Geany, Embarcadero RAD Studio, Code :: Blocks, Xcode або Microsoft Visual Studio, Visual Studio Code але є і IDE для однієї певної мови програмування - як, наприклад , Visual Basic, Delphi, Dev-C ++.

Для розробки веб-додатків в наш час актуальними є такі середовища розробки:

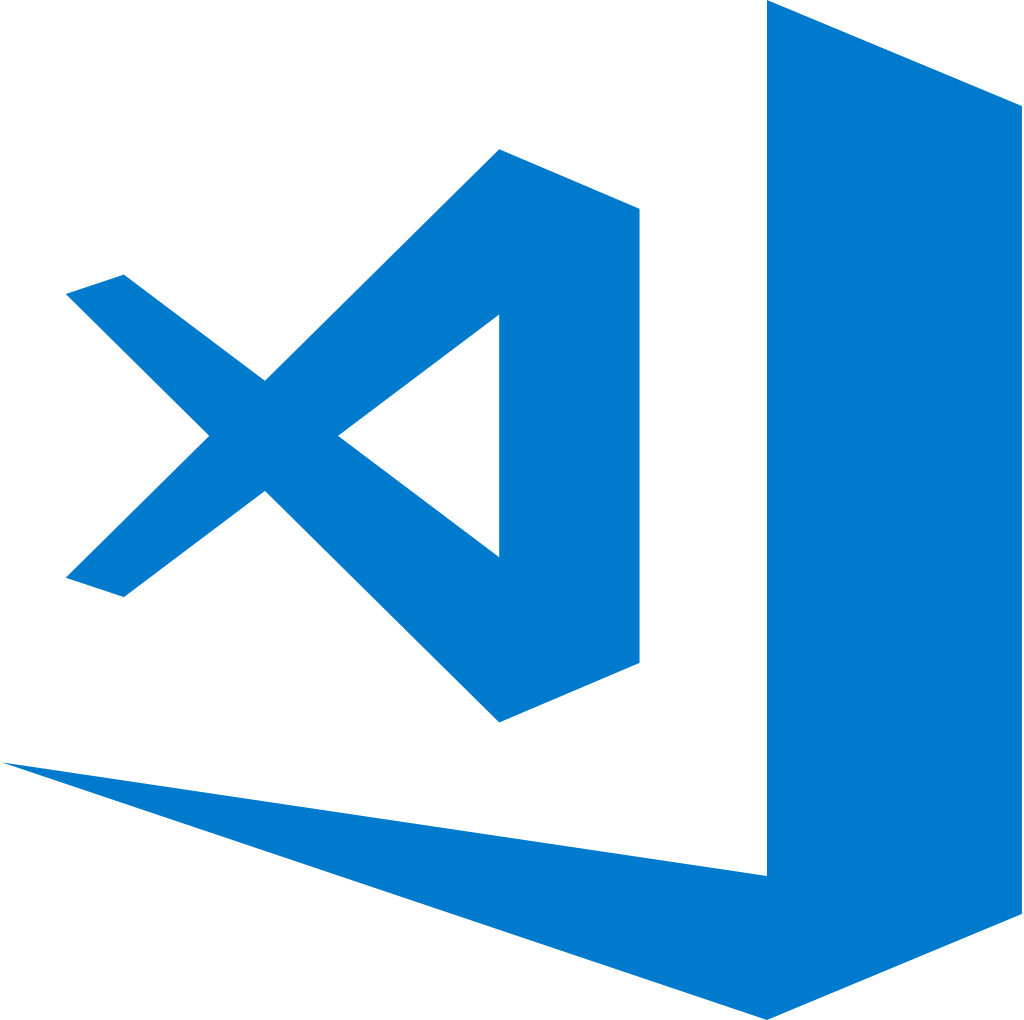


Рисунок 2.3.1.1 – логотип Visual Studio Code

1. **Visual Studio Code** — засіб для створення, редагування та завантаження сучасних веб-застосунків і програм для хмарних систем. Visual Studio Code розповсюджується безкоштовно і доступний у версіях для платформ Windows, Linux і OS X. Компанія Microsoft представила Visual Studio Code у квітні 2015 на конференції Build 2015. Це середовище розробки стало першим крос-платформовим продуктом у лінійці Visual

Studio.

За основу для Visual Studio Code використовуються напрацювання вільного проекту Atom, що розвивається компанією GitHub. Зокрема, Visual Studio Code є надбудовою над Atom Shell, що використовують браузерний рушій Chromium і Node.js.

Примітно, що про використання напрацювань вільного проекту Atom на сайті Visual Studio Code і в прес-релізі і в офіційному блозі не згадується. Редактор містить вбудований зневаджувач, інструменти для роботи з Git і засоби рефакторингу, навігації по коду, автодоповнення типових конструкцій і контекстної підказки. Продукт підтримує розробку для платформ ASP.NET і Node.js, і позиціонується як легковаге рішення, що дозволяє обійтися без повного інтегрованого середовища розробки. Серед підтримуваних мов і технологій: JavaScript, C++, C#, TypeScript, jade, PHP, Python, XML, Batch, F#, DockerFile, Coffee Script, Java, HandleBars, R, Objective-C, PowerShell, Luna, Visual Basic, Markdown, JSON, HTML, CSS, LESS і SASS, Haxe.

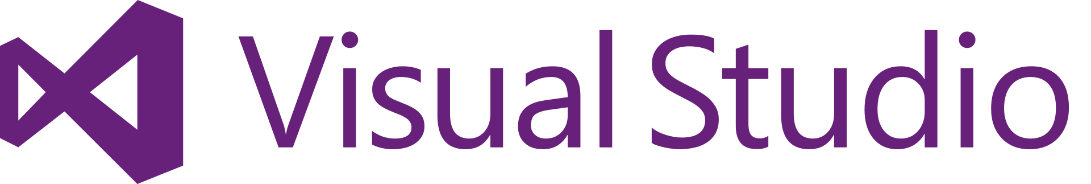


Рисунок 2.3.1.2 – Логотип Visual Studio

1. **Microsoft** **Visual Studio** - серія продуктів фірми Майкрософт, які включають інтегроване середовище розробки програмного забезпечення та ряд інших інструментальних засобів. Ці продукти дозволяють розробляти як консольні програми, так і програми з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також веб-сайти, веб-застосунки, веб-служби як в рідному, так і в керованому кодах для всіх платформ, що підтримуються Microsoft Windows, Windows Mobile, Windows Phone, Windows CE, .NET Framework, .NET Compact Framework та Microsoft Silverlight.

Visual Studio 2017 - Перший нестабільний випуск наступної версії програми під умовною назвою "15" побачив світ 30 березня 2016 року. Основними змінами стали інтерфейс інсталятора та численні незначні покращення у різних компонентах середовища розробки. Очікується підтримка мови програмування Solidity.

Версія остаточно побачила світ під назвою Visual Studio 2017 7-го березня 2017-го року.



Рисунок 2.3.1.3 – Логотип MonoDevelop

1. **MonoDevelop** — відкрите інтегроване середовище розробки для платформ Linux, Mac OS X[4] та Microsoft Windows[5], передусім націлене на розробку програм, які використовують і Mono, і Microsoft .NET framework. На даний момент підтримуються мови C#, Java, Boo, Visual Basic.NET, CIL, Python, Vala, C та C++. Також MonoDevelop підтримує такі технології, як Gtk#, ASP.NET MVC, Silverlight, MonoMac и MonoTouch.

MonoDevelop включає можливості подібні до NetBeans та Microsoft Visual Studio, такі як автоматичне доповнення, інтеграція контролю коду, графічний користувацький інтерфейс і веб-дизайнер. В MonoDevelop інтегрований Gtk# GUI дизайнер під назвою Stetic.

2.3.2 Мови програмування

Мови програмування які використовуються підчас створення веб-додатку в середовищах розробки, які наведені вище:

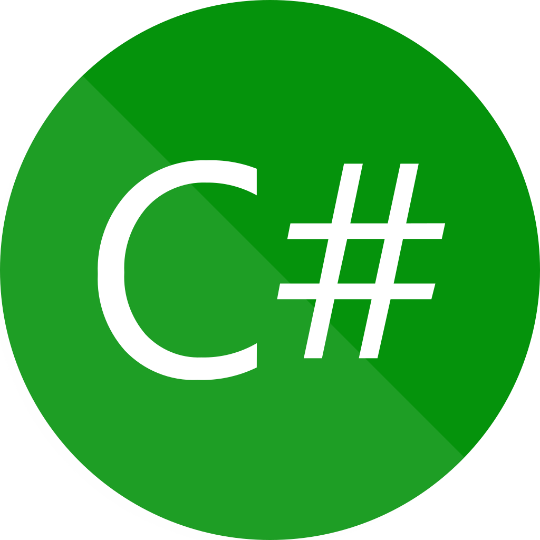


Рисунок 2.3.2.1 – Логотип C#

1. **C# (вимовляється Сі-шарп)** — об'єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET. Розроблена Андерсом Гейлсбергом, Скотом Вілтамутом та Пітером Гольде під егідою Microsoft Research (при фірмі Microsoft).

Синтаксис C# близький до С++ і Java. Мова має строгу статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML. Перейнявши багато що від своїх попередників — мов С++, Delphi, Модула і Smalltalk — С#, спираючись на практику їхнього використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад множинне спадкування класів (на відміну від C++).

C# стандартизований в ECMA[5] та ISO.

У серпні 2000 Microsoft Corporation, Hewlett-Packard та Intel Corporation виступили спонсорами стандартизації специфікації мови C#, а також Common Language Infrastructure (CLI) в організації зі стандартизації ECMA International. У грудні 2001 ECMA випустила ECMA-334 Специфікація мови C#. C# стала стандартом ISO у 2003 (ISO/IEC 23270:2006 — Information technology—Programming languages—C#). До того ECMA ще встигла адоптувати еквівалентну специфікацію як другу редакцію C# у грудні 2002.

У червні 2005 ECMA схвалила редакцію 3 специфікації C#, і відредагувала ECMA-334. Доповнення включали часткові класи, анонімні методи, тип null, і генерики (аналоги шаблонів C++).

У липні 2005 ECMA подала стандарти і відповідні технічні умови на ISO/IEC JTC 1 через пришвидшену процедуру (Fast-Track). Цей процес звичайно займає 6-9 місяців.



Рисунок 2.3.2.2 – Логотип JavaScript

1. **JavaScript (JS)** — динамічна, об'єктно-орієнтована прототипна мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується для створення сценаріїв веб-сторінок, що надає можливість на стороні клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд веб-сторінки.

JavaScript класифікують як прототипну (підмножина об'єктно-орієнтованої), скриптову мову програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми програмування (імперативну та частково функціональну) і деякі відповідні архітектурні властивості, зокрема: динамічна та слабка типізація, автоматичне керування пам'яттю, прототипне наслідування, функції як об'єкти першого класу.

Мова JavaScript використовується для: написання сценаріїв веб-сторінок для надання їм інтерактивності; створення односторінкових веб-застосунків (React, AngularJS, Vue.js); програмування на стороні сервера (Node.js); стаціонарних застосунків (Electron, NW.js); мобільних застосунків (React Native, Cordova);

сценаріїв в прикладному ПЗ (наприклад, в програмах зі складу Adobe Creative Suite чи Apache JMeter);

всередині PDF-документів тощо.

Незважаючи на схожість назв, мови Java та JavaScript є двома різними мовами, що мають відмінну семантику, хоча й мають схожі риси в стандартних бібліотеках та правилах іменування. Синтаксис обох мов отриманий «у спадок» від мови С, але семантика та дизайн JavaScript є результатом впливу мов Self та Scheme.



Рисунок 2.3.2.3 – Логотип HTML

1. **HTML** — стандартна мова розмітки для створення веб-сторінок і веб- додатків .

З Cascading Style Sheets (CSS) і JavaScript, вона утворює тріаду основних технологій для World Wide Web.

Веб-браузери отримують HTML-документи з веб-сервера або з локальної пам'яті і передають документи в мультимедійні веб-сторінки. HTML описує структуру веб-сторінки семантично і спочатку включені сигнали для зовнішнього вигляду документа.

Елементи HTML є будівельними блоками сторінок HTML. За допомогою конструкцій HTML, зображення та інші об'єкти, такі як інтерактивні форми, можуть бути вбудовані у візуалізовану сторінку. HTML надає засоби для створення структурованих документів, позначаючи структурну семантику тексту, наприклад заголовки, абзаци, списки, посилання, цитати та інші елементи. Елементи HTML окреслені тегами , написаними з використанням кутових дужок . Теги, такі як і безпосередньо вводять вміст на сторінку. Інші теги, такі як <img /> <input /> <p>оточують і надають інформацію про текст документа і можуть включати інші теги як під-елементи. Браузери не показують теги HTML, але використовують їх для інтерпретації вмісту сторінки.

HTML може вбудовувати програми, написані на мові сценаріїв, наприклад JavaScript, що впливає на поведінку та вміст веб-сторінок. Включення CSS визначає вигляд і компонування вмісту. World Wide Web Consortium (W3C), які супроводжують як HTML і CSS стандартів, заохочує використання CSS над явним презентаційним HTML з 1997 року.

HTML впроваджує засоби для: створення структурованого документа шляхом позначення структурного складу тексту: заголовки, абзаци, списки, таблиці, цитати та інше; отримання інформації із Всесвітньої мережі через гіперпосилання;

створення інтерактивних форм;

включення зображень, звуку, відео, та інших об'єктів до тексту.

2.3.3 База даних

База даних (БД) - упорядкований набір логічно взаємопов'язаних даних, що використовується спільно, та призначений для задоволення інформаційних потреб користувачів. У технічному розумінні включно й система управління БД.

Головним завданням БД є гарантоване збереження значних обсягів інформації та надання доступу до неї користувачеві або ж прикладній програмі. Таким чином БД складається з двох частин: збереженої інформації та системи управління нею. З метою забезпечення ефективності доступу записи даних організовують як множину фактів (елемент даних).



Рисунок 2.3.3.1 – Логотип Microsoft SQL Server

**1. Microsoft SQL Server** - комерційна система керування базами даних, що розповсюджується корпорацією Microsoft. Мова, що використовується для запитів — Transact-SQL, створена спільно Microsoft та Sybase. Transact-SQL є реалізацією стандарту ANSI / ISO щодо структурованої мови запитів SQL із 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. Microsoft SQL Server як мову запитів використовує версію SQL, що отримала назву Transact-SQL (скорочено T-SQL), яка є реалізацією SQL-92 (стандарт ISO для SQL) з багатьма розширеннями. T-SQL дозволяє використовувати додатковий синтаксис процедур, що зберігаються і забезпечує підтримку транзакцій (взаємодія бази даних з керуючим застосунком). Microsoft SQL Server та Sybase ASE для взаємодії з мережею використовують протокол рівня застосунка під назвою Tabular Data Stream (TDS, протокол передачі табличних даних).

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

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

SQL Server підтримує надлишкове дублювання даних за трьома сценаріями:

Знімок: Виконується «знімок» бази даних, який сервер відправляє одержувачам.

Історія змін: Всі зміни бази даних безперервно передаються користувачам.

Синхронізація з іншими серверами: Бази даних декількох серверів синхронізуються між собою. Зміни усіх баз даних відбуваються незалежно на кожному сервері, а під час синхронізації відбувається звірка даних. Дублювання такого типу передбачає можливість вирішення протиріч між базами даних.

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

**2. SQL Server Management Studio (SSMS)** — утиліта із Microsoft SQL Server 2005 и більш пізніших версій для конфігурування, управління і адміністрування всіх компонентів Microsoft SQL Server. Утиліта включає скриптовий редактор і графічну программу, яка працює з об’єктами и налаштуваннями сервера.

Головним інструментом SQL Server Management Studio являється Object Explorer, який дозволяє користувачу переглядати, вилучати об’єкти сервера, а також повністю ними керувати.

Ще є SQL Server Management Studio Express для Express версії сервера, яка являється безкоштовною. Однак в ні немає підтримки ряду компонентів (Analysis Services, Integration Services, Notification Services, Reporting Services) и SQL Server 2005 Mobile Edition.

Починаючи з версії 16.5.3 пакет SSMS виділений в окремий оновлюваний продукт. Версія доступна на даний момент SSMS 18.0.18118.0 (general availability) підтримує MS SQL server починаючи з версії 2008 по 2019 (preview)

Підсумовуючи все вище сказане можливо зробити висновки що найкращим вибором для мене було середовище розробки як Visual Studio Code в ньому самий широкий вибір інструментарію на відміну від MonoDevelop. Тому для мене вибір Visual Studio Code був єдиновірним. На даний момент для розробки на WEB платформі застосовується різні мови програмування. Проте я обрав такі мови як C#, JavaScript, HTML5, CSS3 так як це одні з найпопулярніших мов, до того ж я звик розробляти додатки саме на них. SQL Server на даний момент займає лідируючу позицію з баз даних і виразних конкурентів в неї нема тому і вибір впав на неї.

2.4 Структура даних

NotFinish

В процесі розробки даного програмного засобу я використав базу даних SQLite в якій було створено такі таблиці:

1. Answer — дана таблиця містить відомості про вірну відповідь на запитання;
2. Category — таблиця в якій знаходяться категорії які є в програмі;
3. Challenge — таблиця в якій містяться запитання;
4. Completion — таблиця для зберігання вірних відповідей;
5. Settings — таблиця для зберігання налаштувань;
6. Statistics — статистика по курсу;
7. User — дані користувача.

Таблиця Answer містить такі поля як:

* \_id — порядковий номер;
* TEXT — додатковий текст до питання ;
* ANSWER\_CORRECT — номер вірної відповіді на питання;
* CHALLENGE\_ID — номер питання.

Таблиця Category містить такі поля як:

* \_id — порядковий номер;
* TITLE — назва категорії;
* DESCRIPTION — опис категорії;
* IMAGE — зображення категорії.

Таблиця Challenge містить такі поля як:

* \_id — порядковий номер;
* CHALLENGE\_TYPE — тип запитання;
* QUESTION — текст запитання;
* CATEGORY\_ID — номер категорії.

Таблиця Completion містить такі поля як:

* \_id — порядковий номер;
* STAGE — стадія проходження тесту;
* LAST\_COMPLETED — останнє питання на яке дано відповідь;
* USER\_ID — порядковий номер користувача;
* CHALLENGE\_ID — номер питання

Таблиця Settings містить такі поля як:

* \_id — порядковий номер;
* TIME\_BOX\_STAGE1 — час на стадії 1;
* TIME\_BOX\_STAGE2 — час на стадії 2;
* TIME\_BOX\_STAGE3 — час на стадії 3;
* TIME\_BOX\_STAGE4 — час на стадії 4;
* TIME\_BOX\_STAGE5 — час на стадії 5;
* TIME\_BOX\_STAGE6 — час на стадії 6

Таблиця Statistics містить такі поля як:

* \_id — порядковий номер;
* SUCCEEDED — кількість виконаних запитань;
* TIME — час на виконання стадії;
* USER\_ID — порядковий номер користувача;
* CHALLENGE\_ID — номер запитання

Таблиця Statistics містить такі поля як:

* \_id — порядковий номер;
* NAME — ім’я користувача;
* AVATAR — аватар користувача;
* SETTINGS\_ID — порядковий номер користувача

2.5 Особливості реалізації

Процес реалізації програмного продукту включає в себе такі етапи як

* Створення інтерфейсу додатку
* Написання коду який взаємодіє з візуальною частиною а також коду який опрацьовує логіку додатку
* Створення та реалізація локальної бази даних
* Реалізація можливості імпортувати свої категорії

Створення інтерфейсу додатку складається з того що на пусту форму ми можемо як переміщати елементи управління програмою, наприклад, кнопки, тестові поля, так і ми можемо просто писати код в форматі ХМL. Головне що спочатку створюється це головне вікно яке буде завантажуватися спочатку. Приклад такого вікна представлений на рисунку 2.4.1

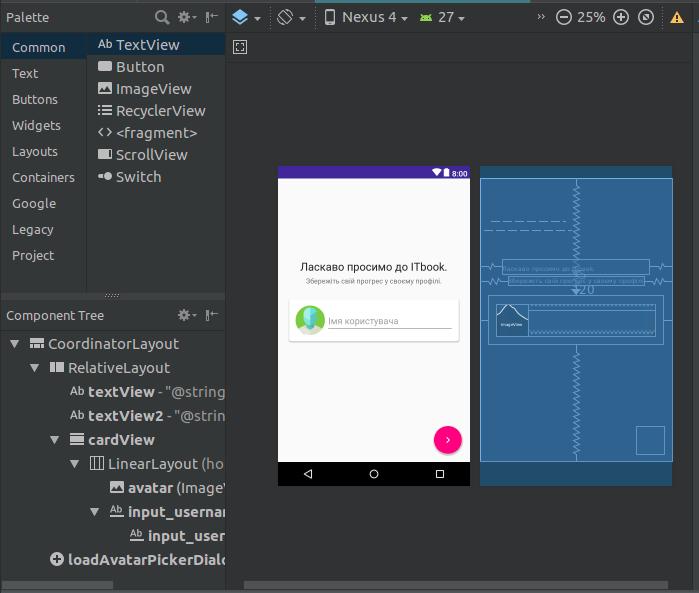


Рисунок 2.4.1 – візуальне відображення головного вікна форми написаної на XML коді

Представлення у вигляді XML коду

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?> <android.support.design.widget.CoordinatorLayout xmlns:android="http://schemas.android.com/apk/res/android" xmlns:app="http://schemas.android.com/apk/res-auto" xmlns:tools="http://schemas.android.com/tools" android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:fitsSystemWindows="true" tools:context=".activities.createuser.CreateUserActivity">  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:paddingBottom="@dimen/activity\_vertical\_margin"  android:paddingLeft="@dimen/activity\_horizontal\_margin"  android:paddingRight="@dimen/activity\_horizontal\_margin"  android:paddingTop="@dimen/activity\_vertical\_margin"  app:layout\_behavior="@string/appbar\_scrolling\_view\_behavior"  tools:context=".activities.createuser.CreateUserActivity"  tools:showIn="@layout/activity\_create\_user">  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textAppearance="?android:attr/textAppearanceLarge"  android:text="@string/welcome"  android:singleLine="true"  android:id="@+id/textView"  android:padding="0dp"  android:layout\_centerHorizontal="true"  android:layout\_above="@+id/textView2"  android:layout\_marginBottom="5dp"/>  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textAppearance="?android:attr/textAppearanceSmall"  android:text="@string/create\_profile"  android:id="@+id/textView2"  android:layout\_above="@+id/cardView"  android:layout\_centerHorizontal="true"  android:layout\_marginBottom="20dp"  />  <android.support.v7.widget.CardView  xmlns:card\_view="http://schemas.android.com/apk/res-auto"  android:id="@+id/cardView"  android:layout\_gravity="center"  android:layout\_width="350dp"  android:layout\_height="wrap\_content"  android:padding="20dp"  card\_view:cardUseCompatPadding="true"  card\_view:cardCornerRadius="4dp"  card\_view:cardElevation="4dp"  card\_view:contentPadding="10dp"  android:layout\_centerInParent="true">  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <ImageView  android:id="@+id/avatar"  android:src="@drawable/anonymous"  android:layout\_width="64dp"  android:layout\_height="64dp"  />  <android.support.design.widget.TextInputLayout  android:id="@+id/input\_username\_layout"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_weight="1"  >  <EditText  android:id="@+id/input\_username"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:hint="@string/username"  android:singleLine="true"  android:imeOptions="actionDone"  />  </android.support.design.widget.TextInputLayout>  </LinearLayout>  </android.support.v7.widget.CardView>  </RelativeLayout>  <android.support.design.widget.FloatingActionButton  android:id="@+id/loadAvatarPickerDialog"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="bottom|end"  android:layout\_margin="@dimen/fab\_margin"  android:src="@drawable/ic\_navigate\_next\_white\_24dp"  android:tint="#FFFFFF"/> </android.support.design.widget.CoordinatorLayout> |

Рисунок 2.4.2 – форма написана на XML коді

При використанні візуальних елементів код розмітки генерується автоматично відповідно до розміру, положенню, кольору і т.д.

В більшості випадків весь інтерфейс розробляється за допомогою візуальних елементів, але для більш точного налаштування всі інші елементи дописують в XML коді.

Після створення такого вікна ми можемо нанизувати інші вікна наприклад після натиснення кнопки «Підтвердити» відкривається наступне вікно – вікно «Аватарів». А те в свою чергу відкриває наступне – вікно вибору категорій яке по суті своїй являться головним.

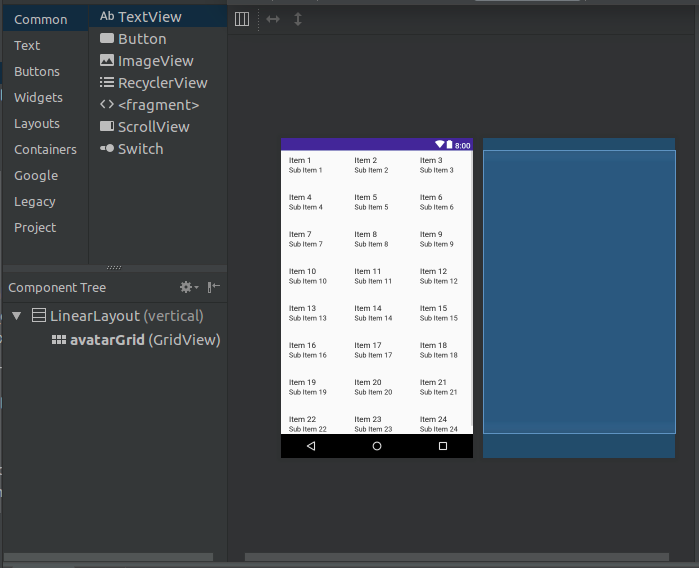


Рисунок 2.4.3 – вікно вибору «Аватарів»

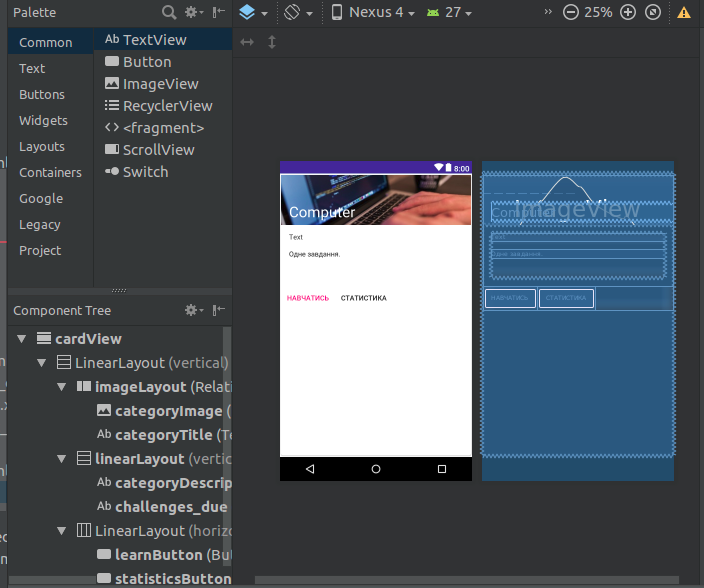


Рисунок 2.4.4 – вікно вибору категорій

Після створення всіх меню ми можемо перейти до розробки самої логіки програми її поведінку те що вона має виконувати.

Весь код програми пишеться на мові програмування Java також можливий варіант написання на мові Kotlin але в нього є свої недолі перераховані в пункті 2.3.2. Класів в середовищі Android Studio мають наступний вигляд представлений на Рисунку 2.4.2

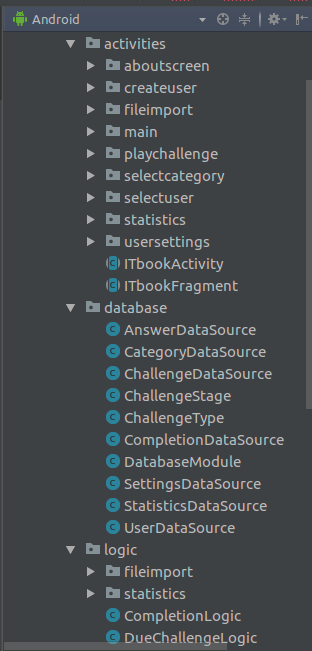


Рисунок 2.4.5 – відображення всіх класів додатку в Android Studio

В першу чергу пишеться оскільки програма починається з головного вікна ми пишемо дії буде виконувати програма при своєму відкритті чи при натисканні різних кнопок, заповненню текстових полів і т.д. Код такого класу буде виглядіти так

|  |
| --- |
| public class MainActivity extends ITbookActivity {  public static String EXTRA\_SHOW\_LOGGEDIN\_SNACKBAR = "SHOW\_SNACKBAR";  private final static int CODE\_FILEPICKER = 0;  @Inject UserManager mUserManager;  {@inheritDoc}  @Override  protected void injectComponent(ITbookComponent component) {  component.inject(this);  }  @Override  public void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.mainactivity);  Toolbar toolbar = (Toolbar)findViewById(R.id.toolbar);  setSupportActionBar(toolbar);  } |

Рисунок 2.4.6 – приблизний вигляд обробки подій головного меню

Далі також для кожного вікна розробляємо свої обробники подій

Для створення та підключення до бази даних ми також це пишемо в класі Java на мові SQL.

|  |
| --- |
| public static void createTable(SQLiteDatabase db, boolean ifNotExists) {  String constraint = ifNotExists? "IF NOT EXISTS ": "";  db.execSQL("CREATE TABLE " + constraint + "\"ANSWER\" (" + //  "\"\_id\" INTEGER PRIMARY KEY ," + // 0: id  "\"TEXT\" TEXT NOT NULL ," + // 1: text  "\"ANSWER\_CORRECT\" INTEGER," + // 2: answerCorrect  "\"CHALLENGE\_ID\" INTEGER NOT NULL );"); // 3: challengeId  } |

Рисунок 2.4.7 – створення таблиці з подальшим заповненням полів

Даний приклад наглядно показує як створюється база даних SQLite. В даному випадку створюється таблиця вірних відповідей на питання з номером питання якщо таблиця ще не створена то вона створюється. Таким чином і розробляються інші таблиці.

Також однією з моїх задач було розробити метод для імпортування своїх категорій і питань. Дану проблема була вирішена таким способом – ми створюємо спеціальний файл фомату .bpc який має мати наступний вигляд

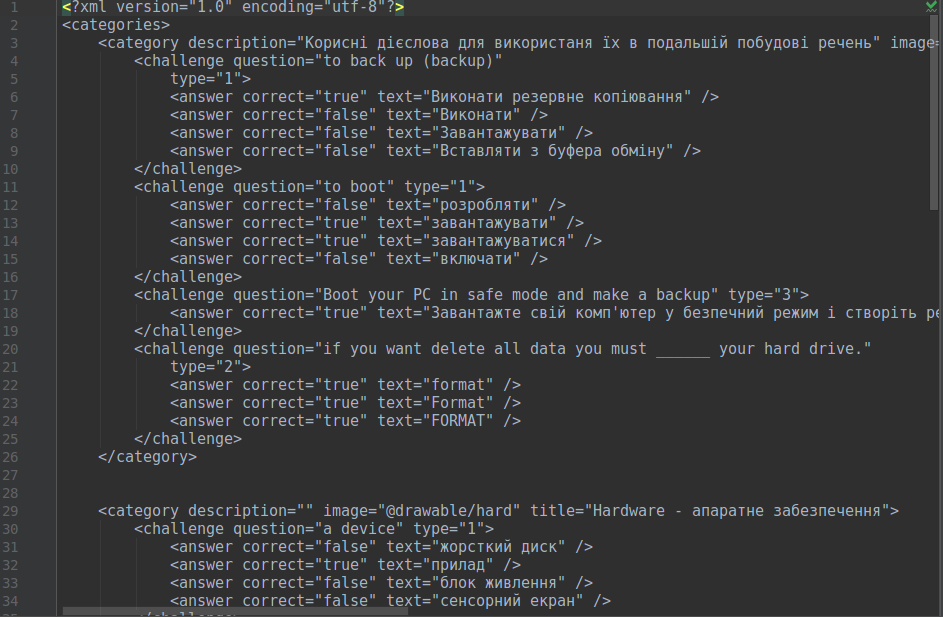


Рисунок 2.4.8 – формат імпортованих файлів в додаток

На даному малюнку ми можемо побачити що даний файл має досить специфічний формат але це зроблено для того щоб інтерпретатор зміг розпізнати по ключовим словам інформацію яку нам потрібно наприклад слово «category» дає інформацію що далі піде нова категорія «description» - опис категорії і т.д.

2.6 Особливості застосування

До особливостей даного програмного забезпечення можливо віднести:

* Використання декілька підходів для тестування користувача.
* Розробка призначена для фахівців IT.
* Розбиття всіх термінів на категорії
* Можливість пройти тестування відразу по всім категоріям
* Імпортування власних категорій користувачем
* Створення власного профілю для ведення статистики
* Статистика, яка відображає проходження курсу
* Налаштування профілю користувача

Дані особливості в цілому являють собою щось нове яке може заповнити пробіл на ринку мобільних додатків. Розроблена програма за своєю суттю призначена тільки для фахівців ІТ чи людей які хочуть ввійти в цю сферу. Тому її можна назвати унікальною в своєму роді.

3. Аналіз результатів

Тестування програмного забезпечення (англ. Software Testing) — це процес технічного дослідження, призначений для виявлення інформації про якість продукту відносно контексту, в якому він має використовуватись. Техніка тестування також включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою оцінки. Може оцінюватись:

* відповідність вимогам, якими керувалися проектувальники та розробники
* правильна відповідь для усіх можливих вхідних даних
* виконання функцій за прийнятний час
* практичність
* сумісність з програмним забезпеченням та операційними системами
* відповідність задачам замовника

Оскільки число можливих тестів навіть для нескладних програмних компонент практично нескінченне, тому стратегія тестування полягає в тому, щоб провести всі можливі тести з урахуванням наявного часу та ресурсів. Як результат програмне забезпечення (ПЗ) тестується стандартним виконанням програми з метою виявлення баґів (помилок або інших дефектів).

На даний момент існують два найрозповсюдженіших види тестування: тестування «білої скриньки» та тестування «чорної скриньки»

Тестування «білої скриньки». Об'єктом тестування тут є не зовнішня, а внутрішня поведінка програми. Перевіряється коректність побудови всіх елементів програми та правильність їхньої взаємодії один з одним. Зазвичай аналізуються керуючі зв'язки елементів, рідше — інформаційні зв'язки. Тестування за принципом «білого ящика» характеризується ступенем, в якому тести виконують або покривають логіку (вихідний текст) програми.

Тестування «чорної скриньки». Основне місце програми тестів «чорної скриньки» — інтерфейс ПЗ.

Ці тести демонструють:

* Як виконуються функції програми.
* Як приймаються вихідні дані.
* Як виробляються результати.
* Як зберігається цілісність зовнішньої інформації.

При тестуванні «чорної скриньки» розглядаються системні характеристики програм, ігнорується їхня внутрішня логічна структура. Вичерпне тестування, як правило, неможливе.

Щоб протестувати даний продукт було вирішено обрати метод експерименту тобто розроблений програмний засіб надавався декільком добровольцям, які згодились покористуватися засобом на протязі деякого часу. Нижче представлені очікувані результати тестування

При першому запуску програмного забезпечення нам запропоновують створити свій профіль для збереження даних користувача. Можливо обтати собі ім’я і «аватар» який буде відображатися ваш профіль.

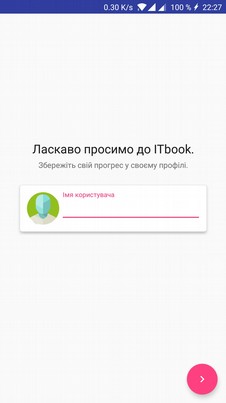
 

Рисунок 3.1 – вікно сторення профілю та вибору «Аватара»

Після того як ми створили свій профіль нам відкривається основне вікно додатку

В цьому вікні ми можемо побачити всі категорії, статистику по ним, а також можливо в список профілів, налаштування і інше.

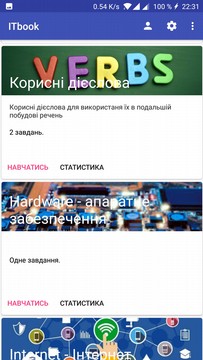
 

Рисунок 3.2 – вікно всіх категорій

Перед тим як почнемо виконувати завдання, відвідаемо владенку “Налаштування”. Вона позначена “шестернею” в правому верхньому куті.

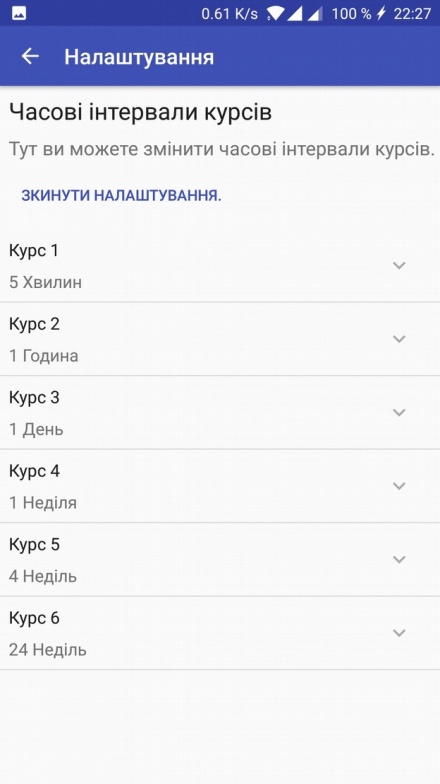
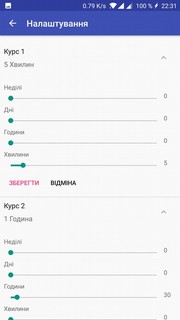
 

Рисунок 3.3 – налаштування часу інтервалів стадій (курсів)

Тут можливо налаштувати час між “стадіями”. Стадії — це кількість повторів питань наприклад, коли ми проходимо тест і відповідаємо на нього вірно то він переходить на наступну стадію, а якщо ні то питання залишається на поточній стадії. Коли ми налаштували натискаємо кнопку “зберегти“ і повертаємося до головного меню.

Якщо ви хочете змінити профіль або створити новий то ми можемо перейти в вкладку “Вибір користувача”. Дана вкладка розташовується також в правому верхньому куті в вигляді “чоловічка”.

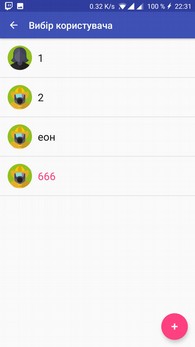


Рисунок 3.4 – вікно вибору користувача

Для того щоб створити новий профіль потрібно лиш натиснути на кнопку на якій зображений “+” (плюс) після того відкриється вікно яке було представлене на першому знімку екрана. Але якщо ви хочете просто відредагувати свій профіль наприклад змінити своє ім’я чи аватар потрібно лише натиснути і утримувати декілька секунд свій профіль після чого з’явиться ще одна кнопка “Редагування”. Натиснувши на неї відкриється вікно яке було представлене на першому знімку екрана. Відредагувавши профіль ми можемо побачити зміни на рис

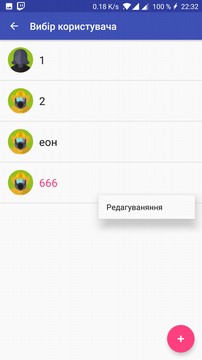
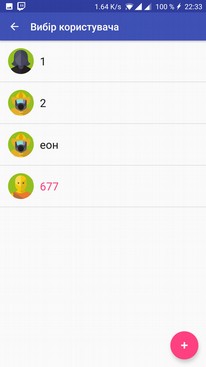
 

Рисунок 3.5 – вікно вибору користувача з відображенням можливості редагування

Повертаємося до головного меню і тепер обираєм курс. Можливо обрати один з курсів або пройти всі курси відразу обравши «Всі категорії». Наприклад натиснемо “Всі категорії” і почнемо проходження тестів.

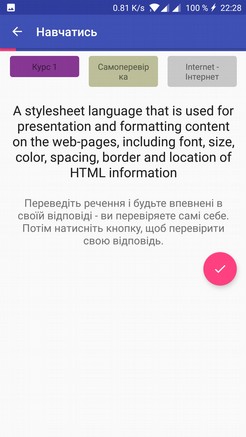
 

Рисунок 3.6 – вікна проходження тесту

Зверху ми можемо побачити на якій стадії знаходиться запитання, тип питання і курс з якого це питання взято. Трішки нижче знаходиться саме питання, ми можемо обрати варіант відповіді натиснути на нього і підтвердити свою відповідь натиснувши на кнопку з зображенням “галочки”. Далі ми переходимо до наступного питання, самі питання обираються випадково і ми не можемо передбачити яким буде наступне питання. Самі питання бувають чотирьох видів: вибір одного з декількох варіантів; вибір декількох варіантів; вписування тексту або слова посимвольно в пропущене місце; самоконтроль, наприклад переклад речення чи певного тексту.

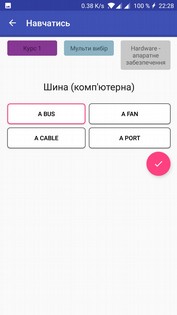
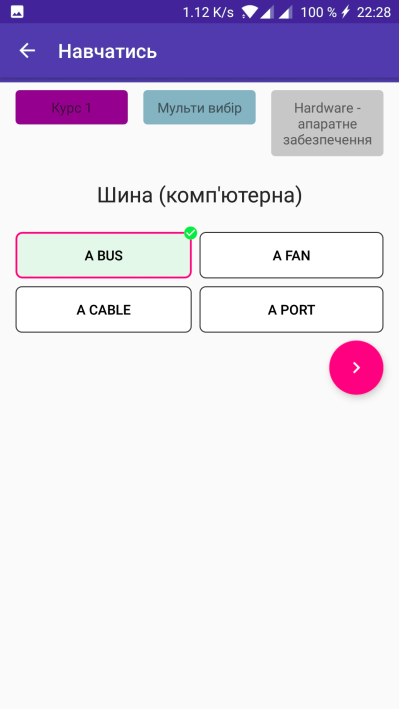
 

Рисунок 3.7 – тест з вибором однієї правильної відповіді



Рисунок 3.8 – тест в якому користувач перевіряє самого себе



Рисунок 3.9 – тест з введенням текстової відповіді

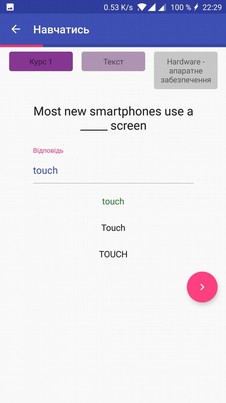


Рисунок 3.10 – повідомлення при невірній відповіді

Після проходження тестів можливо перевірити свою статистику на курсі натиснувши на відповідну клавішу.

Рисунок 3.11 – статистика по проходженню категорії

В даному вікні ми можемо побачити: скільки пройдено питань на курсі на даній стадії; які питання находяться на різних стадіях; найбільше відповідей на питання; питання на які дано найбільше неправильних відповідей; питання на які дано найбільше коректних відповідей.

Також в даному програмному засобі реалізована система імпорту свої питань але їх потрібно оформити в певному форматі. Для того щоб імпортувати додаткова категорії потрібно в головному меню натиснути на кнопку “інше” зображеною в вигляді “трьох крапок” і вибрати пункт “Import BPC” далі відкривається вікно вибору файлу в файловому менеджері

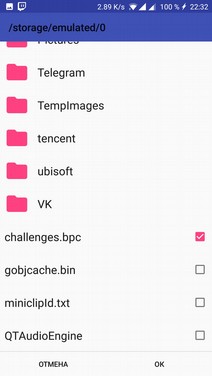


Рисунок 3.12 – вікно імпортування файлу

Обираємо файл для імпорту натискаємо “ОК” і дані курси імпортувались в додаток

Після проведення експерименту було виявлено що даний програмний засіб не має досить видимих недоліків чи невідповідностей. Таким чином можемо зробити висновок що додаток відповідає очікуваному результату.

Висновки

Даний програмний продукт створювався тому, що на даний момент фахівці ІТ потрібні знання технічної англійської мови і додаток надає їм зручний спосіб навчання термінів і інших слів на англійській мові для того щоб людина яка скористалась даним додатком могла знайти «спільну мову» з іншою людиною яка займається в сфері ІТ.

Проаналізувавши ринок додатків було виявлено що на даний момент відсутній додаток який би задовольняв всі потреби користувача, а саме: в першу чергу це саме направлення даного продукту – воно вузько спеціалізоване, призначене для користувачів в сфері ІТ; відсутність на ринку додатків з українською локалізацією; відсутність додавати свої категорії.

Платформою на яку впав вибір розробки даного додатку стала операційна система Android. Дана платформа на, даний момент, займає найбільшу частинку ринку смартфонів, а в Україні, на користувачів якої спрямований додаток, відсоток Android платформи ще більший.

Для створення даного продукту було обрано такі програмні засоби і мови програмування як: Android Studio – офіційне середовище розробки від компанії Google яка володіє платформою Android; Java – об’єктно-орієнтована мова програмування в Android Studio; XML – мова розмітки елементів; SQLite – база даних.

В результаті тестування було виявлено що даний додаток не має явних відхилень від технічного завдання і повністю відповідає їм.

В подальшому розвитку планується добавлення нових модулів таких як, наприклад: словник всіх слів, озвучку слів, виявлення в тексті слів, які наявні в словнику, з виділенням і т.д.

ПЕРЕЛІК ПОСИЛАНЬ

1. Герберт Шілдт. Java 8. Повне керівництво, 9-е видання = Java 8. The Complete Reference, 9th Edition. - М .: «Вільямс», 2015. - 1376 с. - ISBN 978-5-8459-1918-2.
2. Кей С. Хорстманн. Java SE 8. Вступний курс = Java SE 8 for the Really Impatient. - М .: «Вільямс», 2014. - 208 с. - ISBN 978-5-8459-1900-7.
3. Фред Лонг, Дхрув Мохіндра, Роберт С. Сікорд, Дін Ф. Сазерленд, Девід Свобода. Керівництво для програміста на Java: 75 рекомендацій з написання надійних і захищених програм = Java Coding Guidelines: 75 Recommendations for Reliable and Secure Programs. - М .: «Вільямс», 2014. - 256 с. - ISBN 978-5-8459-1897-0.
4. Кей С. Хорстманн, Гарі Корнелл. Java. Бібліотека професіонала, тому 1. Основи. 9-е видання = Core Java, Volume I: Fundamentals (9th Edition). - М .: «Вільямс», 2013. - 864 с. - ISBN 978-5-8459-1869-7.
5. Кей С. Хорстманн. Java. Бібліотека професіонала, тому 1. Основи. 10-е видання = Core Java. Volume I - Fundamentals (Tenth Edition). - М .: «Вільямс», 2017. - 864 с. - ISBN 978-5-8459-2084-3.
6. Кей С. Хорстманн. Java. Бібліотека професіонала, тому 2. Розширені засоби програмування. 10-е видання = Core Java. Volume II - Advanced Feature (Tenth Edition). - М .: «Вільямс», 2017. - 976 с. - ISBN 978-5-9909445-0-3.
7. Баррі Берд. Java 8 для чайників = Java For Dummies, 6th edition. - М .: «Діалектика», 2015. - 400 с. - ISBN 978-5-8459-1928-1.
8. Джеймс Гослінг, Білл Джой, Гай Стіл, Гілад Брача, Алекс Баклі. Мова програмування Java SE 8. Детальний опис, 5-е видання = The Java Language Specification, Java SE 8 Edition (5th Edition) (Java Series). - М .: «Вільямс», 2015. - 672 с. - ISBN 978-5-8459-1875-8.
9. Джошуа Блох. Java. Ефективне програмування = Effective Java. - М .: Лорі, 2002. - 224 с. - ISBN 5-85582-169-2.
10. Монахов Вадим. Мова програмування Java і середовище NetBeans. - 3-е изд. - СПб .: БХВ-Петербург, 2011. - 704 с. - ISBN 978-5-9775-0671-7.
11. Брюс Еккель. Філософія Java = Thinking in Java. - 3-е изд. - СПб .: Пітер, 2003. - 976 с. - ISBN 5-88782-105-1.
12. Офіційний збірник документації та засобів для розробки на платформі Android - [Электронный ресурс]: www/ URL: <https://developer.android.com/>.
13. Офіційний збірник документації для розробки на мові Java [Электронный ресурс]: www/ URL: <https://docs.oracle.com/javase/tutorial/>.
14. Голощапов А. Google Android: програмування для мобільних пристроїв. - СПб .: БХВ-Петербург, 2010. - 448 с. - ISBN 978-5-9775-0562-8.
15. Коматінені С., Маклін Д., Хешімі С. Google Android: програмування для мобільних пристроїв = Pro Android 2. - 1-е изд. - СПб .: Пітер, 2011. - 736 с. - ISBN 978-5-459-00530-1.
16. Сатія Коматінені, Дейв Маклін. Android 4 для професіоналів. Створення додатків для планшетних комп'ютерів і смартфонів = Pro Android 4. - М .: Вільямс. - 880 с. - ISBN 978-5-8459-1801-7.
17. Роджерс Р., Ломбардо Д. Android. Розробка додатків. - М .: ЕКОМ Паблишерз, 2010. - 400 с. - ISBN 978-5-9790-0113-5.
18. Донн Фелкер. Android: розробка додатків для чайників = Android Application Development For Dummies. - М .: Діалектика, 2011. - 336 с. - ISBN 978-5-8459-1748-5.