ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

Пояснювальна записка

до бакалаврської роботи
на ступінь вищої освіти бакалавр

на тему: «**РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ НА ANDROID ДЛЯ САМООСВІТИ МОВОЮ JAVA.**»

Виконав: студент 4 курсу, групи ПД–41
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності/спеціалізації)

Мальований А.М.
(прізвище та ініціали)

Керівник Гаманюк І.М.
(прізвище та ініціали)

Рецензент

(прізвище та ініціали)

Київ –2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти -«Бакалавр»
Спеціальність підготовки – 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного
забезпечення
_____ Негоденко
О.В.
"____" _____2021 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТА

МАЛЬОВАНОГО АРТЕМА МИКОЛАЙОВИЧА
(прізвище, ім'я, по батькові)

1. Тема роботи: «Розробка мобільного додатку на Android для самоосвіти мовою_Java»

Керівник роботи: Гаманюк І.М., старший викладач кафедри ІПЗ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом вищого навчального закладу від «12» березня 2021 року №65.
Строк подання студентом роботи 01.06.2021.
2. Вхідні дані до роботи
Методи розробки додатку;
Науково-технічна література з питань, пов'язаних з програмним забезпеченням щодо освіти та онлайн курсів; Зміст розрахунково-пояснювальної записки(перелік питань, які потрібно розробити).
4.1 Системи онлайн курсів та відео уроків
4.2 Вимоги та оцінка якості системи.

4.3 Опис проектування системи.
4.4 Опис використаних технологій.

3. Перелік демонстраційного матеріалу (назва основних слайдів)

1. Актуальність додатку
2. Існуюче програмне забезпечення
3. Принцип роботи додатку
4. Архітектура додатку
5. Логічна діаграма компонентів архітектури програмного забезпечення

4. Дата видачі завдання 19.04.2021

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	19.04-21.04	Виконано
2	Вимоги до системи	22.04-23.04	Виконано
3	Створення та навчання моделі для вилучення полів	24.04-25.04	Виконано
4	Створення та навчання моделі для вилучення таблиць	26.04-28.04	Виконано
5	Концепція та архітектура програмного забезпечення	29.04-30.05	Виконано
6	Вступ, висновки, реферат	01.05-08.05	Виконано
7	Розробка обов‘язкових демонстраційних матеріалів	09.05	Виконано
8	Попередній захист роботи	11.05	Виконано
9	Здача роботи	01.06	Виконано

Студент _____ Мальований А.М.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Гаманюк І.М.
(підпис) (прізвище та ініціали)

ЗМІСТ

РЕФЕРАТ

Текстова частина бакалаврської роботи 45 с., 18 рис., 6 джерел.

«РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ НА ANDROID ДЛЯ САМООСВІТИ МОВОЮ JAVA.»

Об'єкт дослідження – навчання студентів в додатках для самоосвіти.

Предмет дослідження – мобільний додаток для самонавчання на Android .

Мета роботи – покращення процесу навчання студентів за рахунок мобільного додатку.

Методи дослідження – методи організації інформаційних систем, аналіз середовища обміну інформацією, аналіз результатів роботи інформаційних систем.

Щоб забезпечити високу якість самонавчання необхідно вирішити ряд завдань:

1. Підготовка викладачів та студентів до користування інформаційними системами;
2. Адаптація та вдосконалення існуючих інформаційних систем для проведення самонавчання;
3. Адміністрування інформаційних систем для забезпечення стабільної роботи додатку;
4. Розширення інформаційних систем для підвищенн якості самонавчання;

У роботі наведено аналіз існуючих додатків, таких як Prometheus, Stepik та ін.

Загальною проблемою цих додатків є необхідність у постійному з'єднанні з мережею інтернет та обов'язкова авторизація.

Даний додаток може бути використаний для підготовки учнями та студентами до тестів та екзаменів з різних дисциплін та для простого вивчення предмету який їх цікавить.

Галузь використання – навчання учнів та студентів.

АННОТАЦІЯ

Дипломну роботу виконано на 45 аркушах, вона містить 0 додатків та перелік посилань на використані джерела з 6 найменувань. У роботі наведено 18 рисунків. Було виконано реалізацію системи дистанційного навчання. Тема роботи є досить актуальною у зв’язку з потребою в структурованій системі дистанційного навчання. Програмний продукт являє собою програму для мобільної платформи Android. Мобільний додаток запрограмований на мові Java. Система може використовуватись для отримання освіти поза вищими навчальними закладами

Ключові слова: Android, мобільний додаток, дистанційне навчання, Java, Android Studio, Gradle.

ABSTRACT

The thesis is completed on 45 sheets, it contains 0 applications and a list of references to used sources of 6 titles. The paper presents 18 figures. The implementationof the distance learning system was implemented. The theme of the work is very relevant in connection with the need for a structured system of distance learning. The software is a program for the Android mobile platform. The mobile application is programmed in Java. The system can be used for education outside of higher education institutions

Keywords: Android, mobile app, distance learning, Java, Android Studio, Gradle.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

USB (universal serial bus) –інтерфейс для послідовного підключення периферійних пристроїв до обчислювальної техніки.

ADB (android debug bridge) – засіб виявлення помилок в додатках, передачі даних між програмою та девайсом.

URI (uniform resourse identifier) – це текстовий рядок, по якому можна знайти дані, наприклад файл.

JDK (java development kit) – комплект розробки додатків на мові Java, що включає в себе стандартні бібліотеки, компілятор, приклади, документацію, виконуючу систему.

AVD (android virtual device) –пристрій спеціально створений для емулятора Android, що має свої властивості для перевірки роботи розробленого продукту та його відладки.

XML (extensible markup language) – стандарт побудови мов для розмітки ієрархічно структурованих даних для обміну через мережу інтернет.

IDE (integrated development environment) – це комплекс програмних засобів, який включає в себе компілятор, текстовий редактор та інтерпритатор.

DPI (dots per inch) – означає роздільну здатність екрану девайса.

API (Application Programming Interface) – набір класів, функцій, процедур, структур, що надаються додатком або операційною системою для застосування у зовнішніх програмних продуктах.

APK (android package) –це архівні файли, додатки, які включають в себе весь код додатку, активи, ресурси та нативні бібліотеки.

JIT (Just-in-time compilation) – це технологія збільшення продуктивності систем, що використовують байт-код, застосовуючи компіляції байт-коду в машинний код безпосередньо під час роботи програми.

SDK (software development kit) – це набір засобів для розробки, що довзволяє спеціалістам створювати програми для певного пакета додатків, платформи, комп’ютерної системи, ігрових консолей, операційних систем.

ПК – персональний комп’ютер.

ПЗ – програмне забезпечення.

ОС – операційна система.

GUI (graphical user interface) – це різновид інтерфейсу користувача, в якому елементи інтерфейсу, представлені користувачеві на дисплеї або виконані у вигляді графічних зображень.

JVM (java virtual machine) – це віртуальна машина, в якій виконуються компіляції на всіх мовах для платформи Java.

ВСТУП

У сучасному суспільстві стрімкими темпами розвиваються інформаційні та мобільні технології, і все більший вплив на наше життя має мобільна техніка та інтернет. Тепер безграмотною людиною вважається не той хто не вміє читати або писати, а той хто не вміє користуватись мобільним телефоном, комп'ютером або інтернетом, але таких людей з кожним роком стає все менше і менше.

За допомогою мобільних технологій ми можемо тепер заплатити за парковку прямо з телефону. Вбудовані карти дозволяють не заблукати в будь-якому районі міста. Нам доступний розклад транспорту за допомогою натискання всього лише однієї кнопки. Зрештою, тепер ми можемо просто дізнаватися і досліджувати цікаві місця, знаходячись поблизу них.

Зараз у нас в кишенях і сумках лежить більше всілякої інформації про навколишній світ ніж будь-коли в історії. Фактично для сотень мільйонів людей стало абсолютно природніми і звичним відразу шукати в смартфонах і планшетах інформацію про будь яку діяльність. Наші гаджети вже можуть передбачити, яка інформація нас може зацікавити і подають нам її в зручній формі.

Подібна на перший погляд. Поверхневе і легковажне повсякденне використання мобільних технологій змінило людське співтовариство. Воно зробило наші зв'язки з іншеми людьми набагато ширше і тісніше. У мережі ми знаходимо друзів яких ніколи в житті не зустрічали наживо. Ми повинні пізнавати і обговорювати будь які актуальні питання і події, що відбуваються по всьому світу. Це стимулює виникнення спільнот і обмін думками які раніше були просто неможливі

Крім змін в нашій соціальній активності та розширенні доступної інформації, мобільні технології незабаром можуть почати активно впливати на наше здоров'я.

В наше життя стійко увійшли мобільні технології, які кардинальним способом покращують і процеси виробництва, і процеси споживання інформації.

Використання мобільних технологій дозволяє бути в курсі всіх подій у світі, докладаючи до цього мінімум зусиль.

Крім того, мобільні технології дозволяють знизити вартість продукції для кінцевих споживачів за рахунок оптимізації процесів, скорочення виробничих витрат і невиробничих витрат.

Метою даної бакалаврської роботи є створення мобільного додатка для розвиваючого освітнього центру, для більш ефективного самонавчання людей та підвищення інформованості користувачів з предметами які вони хочуть вивчати та закріплення практичних навичок в вивченні мов, програмування та ін.

1.ЗАДАЧА ПОБУДОВИ СИСТЕМИ САМООСВІТИ ДЛЯ ВИКОРИСТАННЯ В ВИЩИХ НАВЧАЛЬНИХ ЗАКЛАДАХ

Протягом останніх декількох років розвиток смартфонів набуває все більших обертів. З кожним днем все більше і більше операційних системи стають досконалішими в багатьох аспектах, мають все більше і більше функцій, розширюють своє коло можливостей щоб реалізувати додатки з використанням найновіших інструментів які може запропонувати розробник. Майже всі перші мобільні додатки так чи інакше почали з'являтися одночасно з появою операційних систем для мобільних пристроїв. Можливості цих перших мобільних додатків були на дуже низькому рівні. В кожній новій версії операційної системи кількість доступних інструментів для розробника зростала. В наслідок чого це дало розробникам змогу захоплювати все більше і більше областей для застосування своїх продуктів, а саме й інструментарій цих додатків ставав більш просторим. Це посприяло розробці додатків для використання у майже всіх областях нашого життя. На сьогоднішній день існує безліч додатків, які можуть об'єднатися у певні категорії, наприклад: відео програвачі, редактори, навігатори, онлайн магазини, додатки для соціальних мереж. В наш час вони використовуються майже повсюди. Зараз кожен популярний банк має власний клієнтський додаток з базою даних, що цим самим значно спрощує взаємодію банків з клієнтами. Більше того сьогодні навіть існують банки, які повністю перейшли в онлайн обслуговування клієнтів. В таких банках обов'язково повинен бути додаток для клієнта, а основним способом взаємодії банка з клієнтами є програмне забезпечення. Також однією з областей, яка майже повністю перейшла в мобільні додатки є пасажирські перевезення. Майже всі служби таксі мають додатки через які клієнт може замовити автомобіль, спостерігати за напрямком руху водія, а також оцінювати сервіс. Такождодатки мають всі мережі

супермаркетів. З допомогою цих додатків можна переглянути асортимент товарів, актуальні знижки а також здійснити замовлення та оплату товарів онлайн. Ці замовлення доставляють не менш популярні кур'єрські служби, також мають власні додатки. Область застосування додатківдуже широка та збільшується з кожним днем.

Однією з категорій додатків є освіта та навчання. Останнім часом вони стають дедалі більш популярними, у зв'язку з популяризацією освіти та саморозвитку. Ці додатки дозволяють отримувати нові знання в інтерактивному форматі, використовуючи для цього мобільний пристрій. Вони містять певні навчальні матеріали у вигляді відеозаписів, аудіо записів, текстових документів. Всі ці матеріли зазвичай мають структурований формат у вигляді курсів. Вони в свою чергу також об'єднуються в певні категорії за темами та областями знань. Курс являє собою сформовану за змістом інформацію яка розрахована на певний період вивчення матеріалу. В ході навчання користувач отримує певний пласт знань та навичок, які він зможе реалізовувати в своєму житті.

Андроїд додатки такого типу розраховані на декілька категорій користувачів. Перша категорія користувачів використовують їх для розширення свого кругозору або для поповнення своїх знань в тих чи інших питаннях. Друга категорія користувачів, яка не має можливості здобувати освіту в вищих навчальних закладах, використовує схожі додатки як своє основне джерело знань та як інструмент освоєння бажаної професії самостійно. Саме для таких людей які хочуть отримувати ці знання за програмою, яка створена для тієї чи іншої спеціальності. Це у свою чергу підвищує рівень та якість знань які користувач хоче отримати.

Отримуємо висновок, що необхідно дослідити можливість створення додатку для самостійного навчання та його впровадження в загальну навчальну систему вищих навчальних закладів.

Отже, основною метою дослідження є створення програмного продукту, який

повинен реалізувати наступні функції:

- створення курсів наповнених потрібними навчальними матеріалами за заданою структурою.
- забезпечення користувача лише тою інформацією , якою наповнений додаток.
- сповіщення користувача про додавання або зміну інформації.
- поділ курсів за спеціальністю.
- зручний для використання графічний інтерфейс користувача.

Отже, для виконання основної задачі додатку необхідно:

- аналіз інструментів для реалізації додатку для дистанційного навчання ввищих навчальних закладах;
- аналіз існуючого програмного забезпечення для дистанційного навчання ввищих навчальних закладах;
- розробка структури курсу та його наповнення.

У ході розробки пріоритетом є собівартість комплексу при збереженні необхідної функціональності. При розробці необхідно звернути увагу на надійність системи, вартість, простоту експлуатації та налагодження системи.

2.АНАЛІЗ ІСНУЮЧИХ ВИМОГ СИСТЕМИ

Розглянемо суть самоосвіти , особливості самостійного навчання, проаналізуємо приклади реалізації систем для самостійного навчання та їх використання у вищих навчальних закладах.

2.1.Поняття самоосвіти

На сьогоднішній день всім користувачам доступні різноманітні системи для навчання та вивчення тих чи інших матеріалів. Актуальність самоосвіти серед них важко переоцінити. – Самостійне навчання за допомогою різних додатків, взаємодія користувача та додатку на відстані, яка відображає всі компоненти навчального процесу, яка реалізована з використанням інтерактивних методів навчання. Такий тип навчання зародився в 90-х роках минулого століття. З того часу самоосвіта значно збільшила свою область застосування та суттєво змінило методи навчання. Сучасна самоосвіта заснована на використанні наступних основних компонентів: середовище передачі інформації (пошта, телебачення, радіо, інформаційні комунікаційні системи) та методів навчання, які залежать від технічного середовища обміну інформацією. Можна виділити наступні основні форми самонавчання : в режимі офлайн та в режимі онлайн.

Самоосвіта має ряд суттєвих переваг :

- далекосяжність – студенти мають можливість навчатись не залежно від місця перебування.
- гнучкість – студенти мають можливість отримувати знання в той час коли їм це потрібно та отримувати лише ту інформацію яка їх цікавить.

В основному в наші дні самонавчання реалізоване у якості додатків на мобільні пристрої та веб платформ. При цьому вони завжди доступні, так як смартфон

протягом основного періоду часу залишається разом з користувачем. Ці додатки мають різну мету, а тому і різні структуру та тип контенту.

Додатки для вивчення певної мови зазвичай реалізовані у вигляді словників. Вивчення мови відбувається безпосередньо в середовищі додатку. Користувачеві надаються вправи в ході яких він поповнює свій словниковий запас і вивчає структуру даної мови. Прикладами таких додатків є: “Duolingo”, “Поліглот”, “Lingualeo”.

Також існують додатки для розвитку певних фізіологічних характеристик людини, таких як: слух, пам`ять, вокальні здібності, швидкість роботи мозку. Вони містять в собі спеціально розроблені вправи, в результаті яких і розвиваються ці фізіологічні характеристики людини. До таких додатків можна віднести наступні: “Speed Reading”, “Vocaberry”, “NeuroNation”.

Ще одним типом навчальних додатків направлені на продукти для поглибленого вивчення тієї чи іншої теми. Вони цілком складаються з контенту та навчальних матеріалів, що відповідають цій темі. Вони користуються значно меншим попитом у користувачів, адже цільова аудиторія кожного з таких додатків є дуже малою у порівнянні з загальною кількістю користувачів. Прикладами таких продуктів можуть слугувати: “Орігамі”, “Star walk 2 Free”, “Екзамен ПДР 2019 Україна”.

Проте основною частиною всіх навчальних додатків є саме ті, що розроблені увигляді середовища онлайн курсів. Ці додатки надають структурований доступ до великої кількості курсів, що охоплюють різні області знань. Вони є найбільш поширеними і їх використовує велика кількість користувачів, адже кожен має змогу знайти саме те, що йому потрібно. Користувач має змогу обирати потрібний йому курс із переліку доступних. Далі йому надаються навчальні матеріали та контент обраному курсу. Після завершення одного курсу, користувач може приступати до наступного. Курси формуються у вигляді списку і розбивають на різні види категорій. Зараз у вільному доступі користувач має доступ майже до всіх додатків такого типу і їх вибір доволі широкий. Аналіз систем самоосвіти у

вигляді онлайн курсів

Системи самоосвіти у вигляді онлайн курсів є найкращою реалізацією самонавчання для мобільних девайсів. Користувач отримує всі свої знання в інтерактивному форматі та у різному вигляді, що значно покращує розуміння та запам’ямтовування інформації. У процесі пошуку тої чи іншої інформації, та аналізу існуючих рішень систем самонавчання, було виявлено, що на даний момент існуючі рішення не виконують певні вимоги і не надають можливості проходження всіх курсів певної спеціальності.

2.2.Система самонавчання Prometheus

Система Prometheus є українським проектом доспутних для всіх онлайн курсів. Даний ресурс було створено в 2014 році. Саме в той час ця програма була відкрита для реєстрації на першу четвірку відритих онлайн курсів цього ресурсу, що були розроблені викладачами з трьох провідних українських вищих навчальних закладів. В перший час з початку запуску на сайті проекту вже було зареєстровано близько 70 тисяч користувачів, які мали доступ до 20 курсів. На сьогоднішній день користувачам доступно більш ніж 100 курсів, які викладають викладачі різноманітних університетів, компаній та організацій. Розміщені на ресурсі курси охоплюють велику кількість областей знань, таких як історія, мова,інженерія та багато інших напрямків, Також в рамках проекту було впроваджено майже всі курси для абітурієнтів, які готуються до вступу в ВНЗ. Основною задачею цієї програми є надання безкоштовного доступу через Інтернет до всіх курсів всім охочим, а також створювати, поширювати та публікувати власні курси провідним вищим навчальним закладам та підприємствам.

Система Prometheus являє собою набір курсів в режимі онлайн доступу, які створені в системі за циклами. Це поєднує в собі декілька важливих аспектів. Перший аспект це заздалегідь записані відео конференції та лекції, які користувачам можна переглянути в будь-який зручний для нього час. Другий аспект це

різноманітні завдання, які надають можливість перевірити себе, як користувачі засвоїли весь матеріал лекції. Третій аспект це форум для всіх користувачів, на якому вони спілкуються між собою та ставлять питання один одному, а викладачі курсів або інші слухачі відповідати на ці запитання. Четвертим аспектом це є можливість всім користувачам що пройшли курс отримати електронний сертифікат за підписом викладача, успішно пройшовши той чи інший курс.І п’ятий аспект це відсутність обмежень для слухачів курсів тобто вони можуть обирати безліч курсів в тому напрямку який вони заздалегідь обрали.

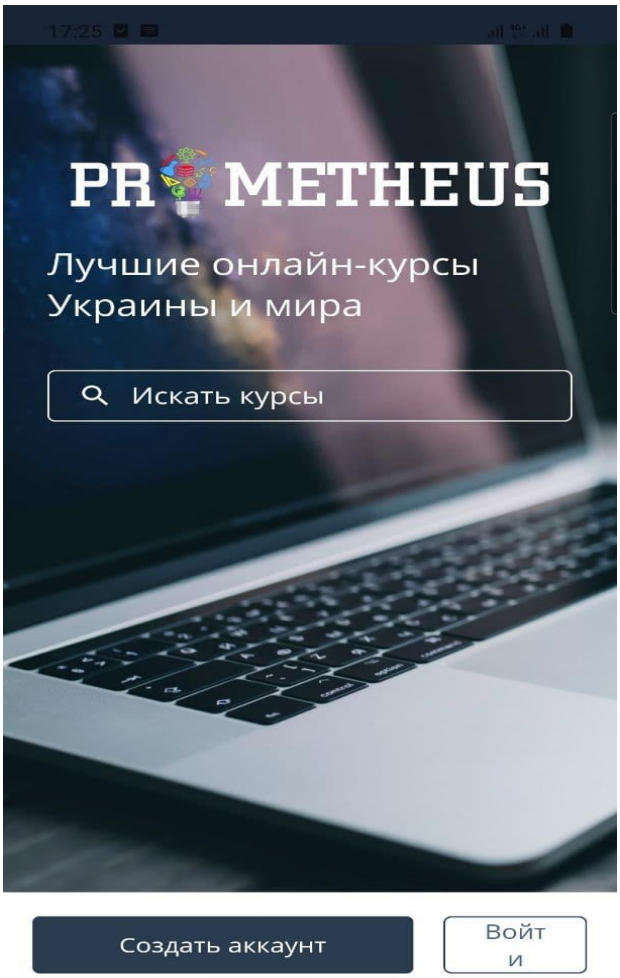

Рисунок 2.1 –Інтерфейс мобільного додатку Prometheus.

Додаток має в собі наступні функції:

- Проходження курсів користувачем онлайн або за допомогою бездротової мережі або мобільного зв’язку;
- завантаження всіх відео уроків для навчання в будь-який час;
- перевірка знань за допомогою тестів уході проходження того чи іншого курсу;
- перегляд свіжих новин та оновлень роздаткових матеріалів курсів;
- проходження курсів в будь-якому місці та у зручний для користувача час;

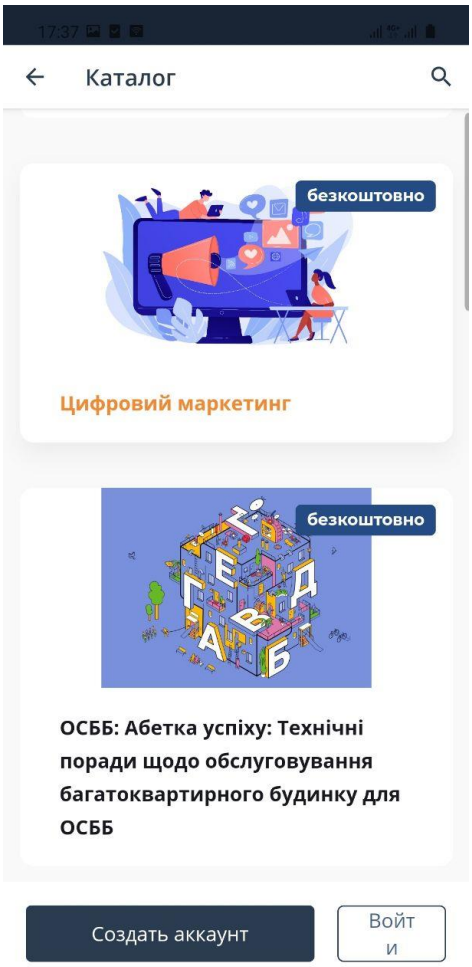

Рисунок 2.2 – Загальний вигляд обраного курсу Prometheus.

. 2.3.Система самонавчання Stepik

Онлайн платформа Stepik є освітньою програмою, а також конструктором безкоштовних курсів та відеоуроків в режимі онлайн. Система надає можливість всім зареєстрованим користувачам створювати онлайн курси та уроки, використовуючи для цього різні типи контенту, такі як відео, аудіо та текстові файли, а також різні тести та задачі з автоматичною перевіркою та зворотнім зв'язком від розробника. В ході свого навчання користувач має можливість спілкуватися з іншими користувачами та викладачами на форумі та в чаті ресурсу. На платформі є велика кількість вже готових курсів з різних тем, наприклад вищої математики, біології, розробки та навіть біоінформатики. На на сьогодняшній день ця освітня платформа має більше 1 мільйона користувачів.

Платформу Stepik було розроблено у 2013 році. Історія платформипочалась з того, що компанія JetBrains та лабораторія з алгоритмічної біології створила безкоштовні курси з біоінформатики, які пізніше і лягли в основу молодої платформи.

Структура всіх курсів що знаходяться на платформі представлена у вигляді відеоуроків, які згруповані в тематичні модулі з темами, проте різні уроки можуть бути розміщені окремо від курсів, вони зберігаються на серверах платформи. В свою чергу відеоуроки складаються з декількох кроків, які можуть бути відео лекціями, текстом або завданнями для застосування практичних навичок. Всього дана платформа дозволяє використовувати не більше 20різних типів завдань, таких як тести, завдання з математичними формулами та рівняннями, задачі на розробку програм та інші.

Автор цього курсу, що був ним складений і розроблений та розміщений на платформі має право зберігати за собою авторське право на розроблений самостійно курс, також він має можливість без обмежень створювати всі свої навчальні матеріали у вигляді відеоуроків та курсів це дає змогу використовувати їх у власному освітньому процесі поза межами даного ресурсу, експортувати та будь-які

інші ресурси чи платформи, мати доступ до статистики проходження його курсів. Всі матеріали, що розміщені на даній платформі та безпосередньо використовуються користувачами знаходяться під дією ліцензії.

Платформа також впровадила онлайн курси, схожі до тих що викладаються у вищих навчальних закладах. Це однорічні та дворічні програми навчання на ресурсі. Також на платформі є можливість різним фахівцям пройти перепідготовку, в результатічого їм буде видано відповідний сертифікат.

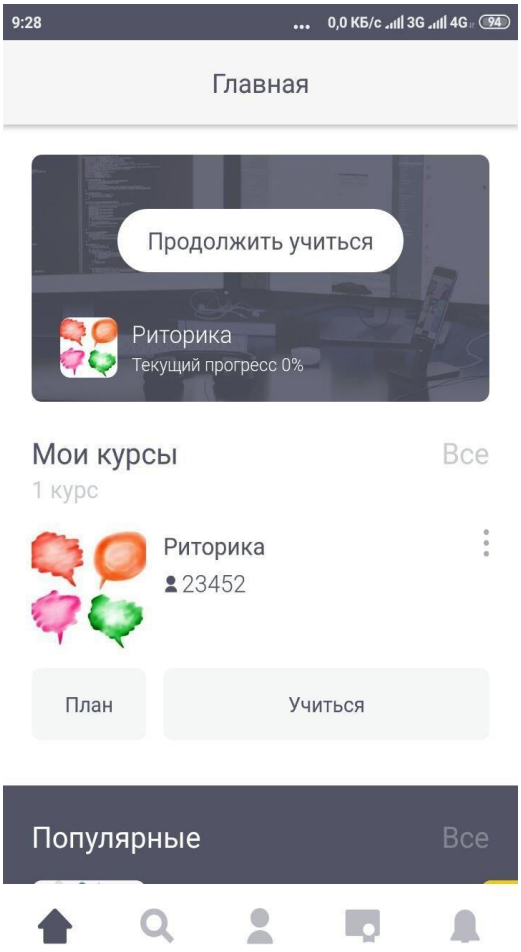

Рисунок 2.2 – Інтерфейс мобільного додатку Stepic.

Додаток має наступні функції:

- Проходження різних курсів онлайн за допомогою мережі Інтернет або мобільного зв’язку;
- обговорення завдань та спілкування з іншими студентами;
- можливість імпортувати терміни проходження своїх курсів в календар, що має бути встановлений на мобільному пристрої;
- завантаження відеолекцій або аудіозаписів на пристрій для відтворення в режимі офлайн та онлайн;
- можливість додати свої матеріали в закладки для подальшого їх відтворення;
- встановлення швидкості відео та лекцій для зручного навчання;
- встановлення нагадування, щоб покращувати свої показники;

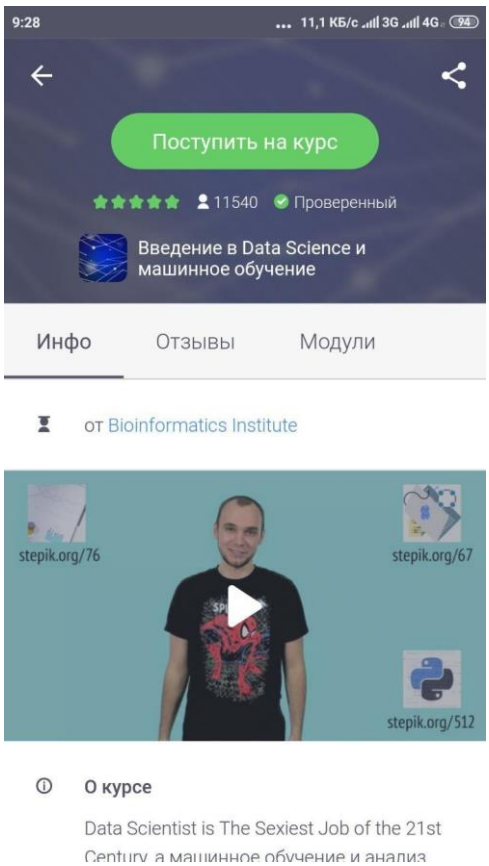

Рисунок 2.3 – Загальний вигляд обраного курсу Stepic.

2.4.Висновки до розділу

В даному розділі мною було розглянуто основні доступні на ринку системи самоосвіти. Оскільки всіляких можливостей для розробки додатків для різних мобільних пристроїв стає дедалі більше і більше, створення нової системи самоосвіти на сьогодняшній день є актуальною задачею. Ця система має бути зручна, проста та зрозуміла для всіх користувачів.

Найкращим рішенням є проектування та розробка нової системи самоосвіти для більш зручного використання на всіх мобільних пристроях, яка буде задовольнити всі потреби користувача та відповідати всім його вимогам для зручного та ефективного отримання знань.

3.МЕТОДИ РЕАЛІЗАЦІЇ ПРОГРАМНОЇ СИСТЕМИ

Для того щоб нам чітко усвідомлювати в якому напрямку рухатися в подальшому, нам необхідно скласти список вимог яким повинен відповідати наш додаток.

Технічні вимоги:

1. Відповідати розробленому дизайну.
2. Працювати без серйозних помилок.
3. Працювати з усіма новими версіями Android.
4. Працювати без візуальних затримок.

Функціональні вимоги:

Користувач заходить в додаток, без будь-якої авторизації, як тільки додаток завантажитися повинен відобразитися розділ Новини.

Для переходу до іншого розділу повинні бути кнопка відповідна дизайну угорі ліворуч і можливість відкриття меню жестом по екрану зліва на право. У відкритому меню повинні бути логотип в шапці меню, список розділів нижче, з інтуїтивно зрозумілими картинками і назвами розділів.

У розділі новин повинна бути можливість прокрутки, як тільки завантажені новини закінчуються в списку, повинні завантажитися ще обмежена кількість новин, якщо новин більше немає, то більше завантаження відбуватися неповинно.

У розділі напрямки при його відкритті повинен з'являється список напрямків, при натисканні на елемент списку має відкритися нове вікно або список з підкатегоріями з

таким же функціоналом. У новому вікні має відображатися вміст, який прийшов з сервера.

Згідно вище перерахованим вимогам, які ми визначаємо для нашого мобільного застосування можна провести аналіз який допоможе нам ефективно, швидко і без зайвих фінансових і людських витрат здійснити комплексну, самодостатню розробку програми. Додаток можна буде легко розширювати, підтримувати, утримувати легко читається програмний код, буде зрозумілий людям, які будуть його розширювати і підтримувати в подальшому.

Для того щоб більш глибше зрозуміти принципи розробки, потрібно зрозуміти для якої програмної платформи ми будемо розробляти додаток для цього нам потрібно познайомитися ближче з операційною системою Android.

Продовжуючи зусилля по залученню до платформи Android нових розробників, Google почала проводити різні змагання. Перше з них, назване Android Developer Challenge (ADC), було проведено в 2008 році. Переможцем проектам були обіцяні щедрі грошові призи. ADC проводилося і в наступному році, в ньому також взяла участь велика кількість претендентів. У 2010 році ADC не проводилося, що може бути пояснено тим, що Android набрала необхідну базу розробників і тому необхідність в залученні нових учасників відпала. Крім того, Google на початку 2010 року почав програму поширення пристроїв. Кожен розробник, додаток (або додатки) якого було завантажено більше 5000 разів при середньому призначеному для користувача рейтингу не менше 3,5 зірки, отримував новий телефон Motorola Droid, Motorola Milestone або NexusOne.

Ця акція викликала бурхливу реакцію спільноти розробників, хоча спочатку зіткнулася з недовірою. Багато хто вважав отримані по електронній пошті повідомлення продуманої містифікацією. На щастя, все виявилося правдою, і тисячі

розробників по всьому світу отримали в подарунок пристрої - вдалий крок з боку Google по залученню нових розробників і підтримки впевненості вже залучених. Крім цього Google пропонує розробникам спеціальну версію апаратів - Android Dev Phone (ADP) Першим ADP був варіант T-Mobile G1 для розробників (також відомий як HTC Dream). Наступне покоління, ADP 2, було варіантом HTC Magic. До того ж Google продавала власний телефон (Nexus One) кінцевим користувачам. Хоча спочатку він не позиціонувався як девелоперський, багато хто розглядав його як нащадка ADP 2.

В результаті Google припинила продажу Nexus One споживачам, і тепер він доступний тільки для її партнерів і розробників. Цей апарат компанії Samsung, що працює на Android 2.3 (Gingerbread), називається Nexus S. ADP-пристрої можуть бути придбані на Android Market (для цього потрібно мати обліковий запис розробника). Щорічна конференція Google I / O - захід, на якому демонструються новітні та кращі технології і проекти Google, і Android в останні роки займає серед них особливе місце. На Google I / O зазвичай проводиться кілька сесій, пов'язаних з Android Android - операційна система для смартфонів, інтернет-планшетів, електронних книг, цифрових програвачів, наручних годинників, ігрових приставок, нетбуків, смартбуків, окулярів Google, телевізорів та інших пристроїв.

В майбутньому планується підтримка автомобілів і побутових роботів. Заснована на ядрі Linux і власної реалізації віртуальної машини Java від Google. Спочатку розроблялася компанією Android, Inc., яку потім купила Google. Згодом Google ініціювала створення альянсу Open Handset Alliance (OHA), який зараз займається підтримкою і подальшим розвитком платформи. Android дозволяє створювати Java-додатка, що керують пристроєм через розроблені

Google бібліотеки. Android Native Development Kit дозволяє перенести бібліотеки і компоненти додатків, написані на Сі та інших мовах.

Це не просто ще один дистрибутив Linux для мобільних пристроїв. При розробці для Android вам, швидше за все, не доведеться мати справу з самі ядром Linux. С точки зору програміста, Android - платформа, абстрагуюча розробника від ядра і дозволяє йому створювати код на Java. Android володіє декількома корисними можливостями. По-перше, це фреймворк, що пропонує великий набір API для створення різних типів додатків і, крім того, що забезпечує можливості повторного використання і заміни компонентів, які пропонуються платформою і сторонніми додатками. По-друге, наявність віртуальної машини Dalvik, що відповідає за запуск додатків на Android. Крім того, до послуг розробника набір графічних бібліотек для 2D- і 3D-додатків, підтримка мультимедіа-форматів (Ogg Vorbis, MP3, MPEG-4, H.264, PNG), API для доступу до камери, GPS, компасом, акселерометру, сенсорного екрану, джойстика і клавіатурі. Є навіть спеціальний API для відтворення фонових звукових ефектів, яке стане в нагоді нам при розробці ігор. Не всі Android-пристрої володіють всіма цими можливостями - у наявності апаратне поділ. Звичайно, список можливостей Android не вичерпується згаданими мною. Архітектура Android формується з набору компонентів. Кожен компонент побудований на основі елементів нижчого рівня. На рис. 3.1 представлений короткий огляд головних компонентів Android.

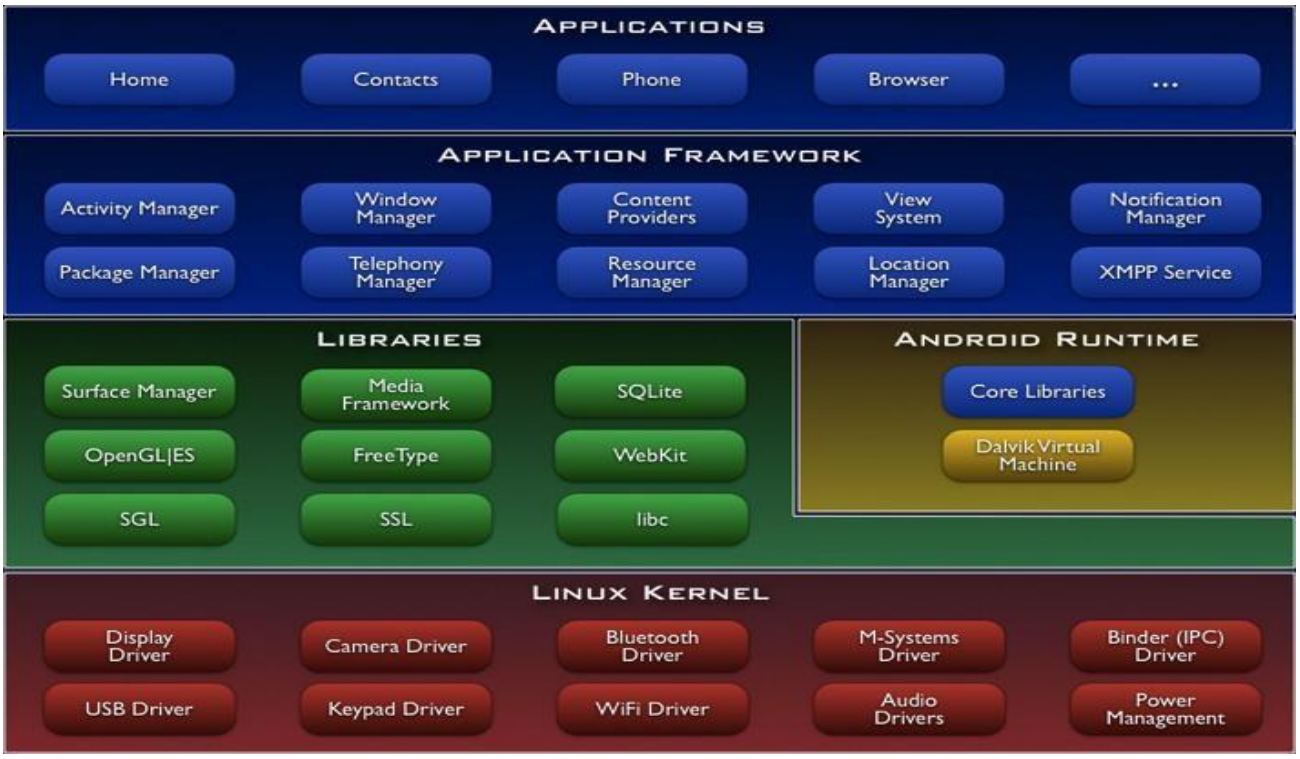

Рисунок 3.1 — Схема архітектури системи.

Фреймворк додатки пов'язує разом системні бібліотеки і середу виконання, створюючи, таким чином, призначену для користувача бік Android. Фреймворк управляє додатками і пропонує продуману середу, в якій вони працюють. Розробники створюють додатки для цього фреймворка за допомогою набору програмних інтерфейсів на Java, що охоплюють такі області, як розробка користувальницького інтерфейсу, фонові служби, оповіщення, управління ресурсами, доступ до периферії і т. Д. Все ключові програми, що поставляються разом з ОС Android (наприклад , поштовий клієнт), написані за допомогою цих API. Додатки, будь вони з інтерфейсом або з фоновими службами, можуть зв'язуватися з іншими додатками. Цей зв'язок дозволяє одному з додатком використовувати компоненти інших. Простий приклад - програма, яка робить фотознімок і потім обробна його. Додаток запитує у системи компонент іншої програми, що забезпечує цю дію. Далі перший додаток може повторно використовувати цей компонент (наприклад, від вбудованого додатка

камери або від фотогалереї). Подібний алгоритм знімає значну частину ноші з програміста, а також дозволяє налаштувати різноманіття аспектів поведінки Android.

У програмуванні для Android використовуються об'єктно орієнтовані технології, тому ми наведемо огляд об'єктних технологій. В умовах постійно зростаючого попиту на нові потужні програмні продукти досить важко поєднувати такі вимоги, як швидкість розробки, правильність роботи і економічність. Об'єкти, по суті, являють собою повторно використовувані програмні компоненти. Як об'єкти можуть використовуватися дата, час, відео, людина, автомобіль та інші предмети матеріального світу. Практично кожне іменник може бути адекватно представлено програмним об'єктом в поняттях атрибутів (наприклад, ім'я, колір і розмір) і поводжень (наприклад, обчислення, переміщення і передача даних). Розробники програм бачать, що використання модульної структури і об'єктно орієнтованого проектування при розробці додатків підвищує продуктивність роботи. Цей підхід прийшов на зміну застосовувався раніше структурному програмуванню - об'єктно-орієнтована код простіше зрозуміти і змінити.

Щоб краще зрозуміти суть об'єктів і їх вмісту, скористаємося простою аналогією. Уявіть собі, що ми знаходимося за кермом автомобіля, і натискаєте педаль газу, щоб набрати швидкість. Що має статися до того, як ми отримуємо таку можливість? Перш ніж ми поведемо автомобіль, хтось повинен його спроектувати. Виготовлення будь-якого автомобіля починається з інженерних креслень, які детально описують пристрій автомобіля. Зокрема, на цих кресленнях показано пристрій педалі акселератора.

За цією педаллю ховаються складні механізми, які безпосередньо прискорюють автомобіль, подібно до того, як педаль гальма приховує механізми, які гальмують автомобіль, а кермо приховує механізми повороту. Завдяки цьому люди, які не мають поняття про внутрішній устрій автомобіля, можуть легко їм управляти.

Подібно тому, як неможливо готувати їжу на кухні, яка лише зображена на аркуші паперу, не можна водити автомобіль, існуючий лише в кресленнях.

Перш ніж ми сядемо за кермо машини, її потрібно побудувати на основі інженерних креслень. Втілений в металі автомобіль має реальну педаль газу, за допомогою якої він може прискорюватися, але і це не все - він не може це робити самостійно, а тільки після того, як водій натисне на педаль. Для виконання операції в програмі потрібно метод, в якому «ховаються» інструкції програми, які безпосередньо виконують операцію. Метод приховує ці інструкції від користувача подібно до того, як педаль газу автомобіля приховує від водія механізми, що викликають прискорення автомобіля. Програмна одиниця, іменована класом, включає методи, які виконують завдання класу. Наприклад, клас, який представляє банківський рахунок, може включати три методи, один з яких поповнює рахунок, другий знімає кошти з рахунку, а третій запитує поточний баланс.

Водій не сяде за кермо автомобіля, поки його не спорудять за кресленнями, так і побудова об'єкта класу необхідно для виконання завдань, визначених методами класу. Цей процес називається створенням екземпляра. Отриманий при цьому об'єкт називається екземпляром класу.

На основі одних і тих же креслень можна створити багато автомобілів, а на основі одного класу можна створити багато об'єктів. Використання існуючих класів для створення нових класів економить час і сили розробника. Повторне використання також полегшує створення більш надійних і ефективних систем, оскільки раніше створені класи і компоненти зазвичай проходять ретельне тестування, налагодження і оптимізацію. Подібно до того, як концепція використання взаємозамінних частин лягла в основу промислової революції, повторно використовувані класи грають ключову роль в програмної революції, в основу якої покладено впровадженням об'єктних технологій.

Коли ми ведемо машину, натиснення педалі газу відправляє автомобілю повідомлення із запитом на виконання певного завдання (прискорення автомобіля). Подібним же чином відправляються повідомлення об'єкту. Кожне повідомлення представляється викликом методу, який «повідомляє» методу об'єкта про необхідність виконання деякої задачі. Наприклад, програма може викликати метод deposit об'єкта банківського рахунку, щоб поповнити банківський рахунок.

Будь-який автомобіль крім можливості виконувати певні операції також володіє атрибутами, такими як колір, кількість дверей, запас палива в баку, показання спідометра і одометра. За аналогією з операціями атрибути автомобіля представляються на інженерних діаграмах (в якості атрибутів автомобіля можуть виступати одометр і покажчик рівня бензину). При водінні автомобіля його атрибути переміщаються разом з ним. Кожен автомобіль містить власний набір атрибутів. Наприклад, кожен автомобіль «знає» про те, скільки бензину залишилося в його баку, але йому нічого не відомо про запаси пального в баках інших автомобілів. Об'єкт, як і автомобіль, має власний набір атрибутів, які він «переносить» з собою при використанні цього об'єкта в програмах. Ці атрибути визначаються як частина об'єкта класу. Наприклад, об'єкт bankaccount має атрибут балансу, який представляє кількість коштів на банківському рахунку. Кожен об'єкт bankaccount «знає» про кількість коштів на власному рахунку, але нічого не «знає» про розміри інших банківських рахунків. Атрибути визначаються за допомогою інших змінних екземпляра класу.

Класи інкапсулюють атрибути і методи в об'єкти (атрибути і методи об'єкта між собою тісно пов'язані). Об'єкти можуть обмінюватися інформацією між собою, але зазвичай вони не «знають» про деталі реалізації інших об'єктів, які приховані всередині самих об'єктів. Подібне приховування інформації життєво важливо в практиці хороших програмних архітектур.

За допомогою успадкування можна швидко і просто створити новий клас об'єктів. При цьому новий клас успадковує характеристики існуючого класу, які при

цьому можуть частково змінюватися. Також в новий клас додаються унікальні характеристики, властиві тільки цьому класу. Якщо згадати аналогію з автомобілем, «трансформер» є об'єктом більш узагальненого класу «автомобіль», у якого може підніматися або опускатися дах.

А тепер дайте відповідь на питання, яким чином ми збираємося програмувати? Швидше за все, таким же чином, як і більшість інших програмістів, - включимо комп'ютер, і почнете вводити вихідний код програми. Подібний підхід годиться при створенні маленьких програм, але що робити в тому випадку, коли доводиться створювати великий програмний комплекс, який, наприклад, керує тисячами банкоматів великого банку? Або якщо вам доводиться очолювати команду з 1000 програмістів, зайнятих розробкою системи управління повітряним рухом наступного покоління? У таких великих і складних проектах не можна просто сісти за комп'ютер і почати набирати код. Щоб виробити оптимальне рішення, слід провести детальний аналіз вимог до програмного проекту (тобто визначити, що повинна робити система) і розробити архітектуру, яка буде відповідати цим вимогам (тобто визначити, як система буде виконувати свої завдання). В ідеалі перед початком створення коду слід виконати цю процедуру і ретельно проаналізувати проект (або доручити виконання цього завдання іншим професіоналам). і проектування системи з застосуванням об'єктно-орієнтованого підходу, значить, ми маємо справу з процесом об'єктно-орієнтованого аналізу і проектування (ООАП). Мови програмування, подібні Java, називаються об'єктно-орієнтованими. Програмування на таких мовах, зване об'єктно-орієнтованим програмуванням (ООП), реалізує об'єктно-орієнтовані проекти в вигляді працездатних систем.

3.1.Аналіз інтерфейсу користувача

Функціональність є фактором, на який розробники додатків часто звертають основну увагу. Вони намагаються створювати програми так, щоб користувачі могли виконувати свої завдання, і їм було зручно це робити. Функціональність важлива, але, тим не менш, це не єдиний показник, який повинен враховуватися в ході розробки.

Естетичний зовнішній вигляд самого додатка і способу його уявлення дозволяє сформувати у споживача позитивну думку про програму. Однак естетичні характеристики дуже суб'єктивні і описати їх кількісно набагато важче, ніж функціональні вимоги або показники продуктивності. Вся естетика додатки часто зводиться до простого вибору: чи співвідносяться між собою використовуються кольори, передають чи елементи інтерфейсу їх призначення і сенс подаються операцій, що відчуває людина при використанні тих чи інших елементів управління і наскільки успішно він їх використовує.

Продуктивність, а так само і надійність, також впливають на перспективу застосування програми. Якщо додаток добре виглядає, має просте і зручне управління, але, наприклад, повільно промальовує екрани, регулярно «підвисає» на десяток-другий секунд або, того гірше, падає з критичною помилкою при некоректних діях користувача, у нього, ймовірно, буде мало шансів на тривалу експлуатацію. У свою чергу, швидка і стабільна робота програми можуть частково компенсувати його не самий стильний дизайн або відсутність якихось вторинних функцій.

Для забезпечення успішної роботи користувача від дизайнера інтерфейсу потрібно дотримуватися балансу між перерахованими вище факторами протягом всього життєвого циклу розробки програми. Це досягається послідовною і ретельним опрацюванням деталей інтерактивної взаємодії на кожному з етапів розробки призначеного для користувача інтерфейсу включають:

1. Проектування

• Функціональні вимоги: визначення мети розробки та вихідних вимог

• Аналіз користувачів: визначення потреб користувачів, розробка сценаріїв, оцінка відповідності сценаріїв очікуванням користувачів

• Концептуальне проектування: моделювання процесу, для якого розробляється програма

• Логічне проектування: визначення інформаційних потоків в додатку

• Фізичне проектування: вибір платформи, на якій буде реалізований проект і засобів розробки.

2. Реалізація

• Прототипування: розробка паперових та / або інтерактивних макетів екранних форм

• Конструювання: створення програми з урахуванням можливості зміни його дизайну.

3.2.Прототипування інтерфейсу користувача

Прототипи стали незамінним етапом професійної розробки програмного забезпечення. Прототипи полегшують життя всім людям, залученим до процесу створення того чи іншого проекту: клієнту, менеджерам, дизайнерам і розробникам. Прототип візуалізує кінцевий продукт і допомагає виявити серйозні проблеми на самих ранніх етапах, допомагаючи уникнути проблем на пізніх стадіях розробки.

Звичайно, початкові ескізи, виконані, наприклад, на папері і втілюють перші ідеї дизайнера ще ніхто не відміняв. Але всю подальшу роботу, в яку будуть залучені всі члени робочої групи, найкраще проводити в спеціальному програмному забезпеченні, яке підтримує редагування, управління версіями і спільну роботу.

Прототип додатки буде створюватися в досить популярною графічній програмі Adobe Photoshop. Ця програма є досить зручною у використанні і швидкому освоєнні для побудови прототипу.

Починати прототипирование ми почнемо з розробки лівого меню, так як даний функціонал є головним в додатку. Перше, що буде додано в макет це так звані StatusBar і Toolbar (Рис 3.2).

Рис 3.2. Зверху Statusbar, знизу Toolbar.

Statusbar - це візуальний графічний елемент системи Android для взаємодії користувача з системою.

Toolbar - також відомий як панелі дій, є одним з найбільш важливих елементів дизайну в діяльності нашого застосування, так як він забезпечує візуальну структуру і інтерактивні елементи, які знайомі користувачам.

У рекомендаціях матеріального дизайну Google йдеться, що об'єкти інтерфейсу користувача повинні відкидати тіні, як і об'єкти реального світу. Коли для подання задається властивість elevation, Android автоматично генерує тінь від цього уявлення. Чим більше значення elevation, тим чіткіше виражена тінь. У рекомендаціях матеріального дизайну наведені бажані значення elevation для різних екранних компонентів - наприклад, для діалогових вікон рекомендоване значення elevation одно 24dp, а для меню - 8dp.

Розробники додатків часто призводять кольору теми у відповідність з фірмовим стилем компанії. Якщо вам буде потрібно змінити кольори теми, рекомендації матеріального дизайну Google по використанню кольору радять вибрати колірну палітру, що складається з основного кольору (не більше ніж з трьома відтінками) і акцентного кольору. Основні кольори зазвичай використовуються для фарбування рядки стану та панелі програми у верхній частині екрану; крім того, вони можуть використовуватися в графічному інтерфейсі.

Згідно з рекомендаціями дизайнерів Google, меню повинно містити елементи вигляді списку з іконками і написами, так само мати заголовок з необхідною

функціональністю. Ми вибрали так званий Navigation Drawer для реалізації навігації в додатку.

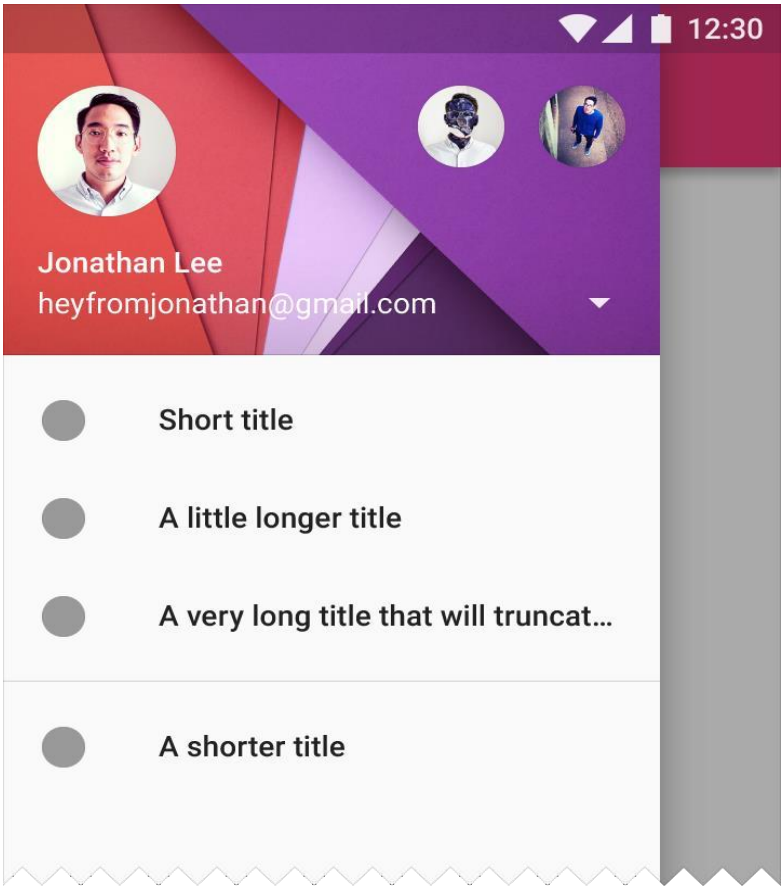

Рис 3.3 Типовий вид меню додатку.

Значок в документації називається "гамбургером" (Hamburger menu). Це офіційна позиція Google. При натисканні зліва вилізе навігаційна шторка. По висоті вона займає весь екран, включаючи системну область. Можете посувати шторку вперед-назад, щоб побачити, що верхня кромка шторки в системній області напівпрозора і не закриває системні значки. Подібна поведінка є на пристроях під Android 5 і вище. На старих пристроях шторка знаходиться під системної панеллю.

Сама шторка складається з двох основних частин - у верхній частині знаходиться картинка і текст, а в нижній - меню зі значками. Меню в свою чергу поділено на дві

групи. У верхній частині значки можна вибрати, і обраний пункт залишиться виділеним. У нижній частині меню пункти не виділяються. Приберіть шторку назад і викличте тепер її не натисканням на значок гамбургера, а рухом пальця від краю екран в центр. Ми побачимо, що під час руху значок трансформується. На жаль, шторка закриває значок і незрозуміло, на що перетворюються три смужки.

Платформа Android надає повний та добре організований асортимент високорівневих API інтерфейсів для створення додатків та використовування основних функцій платформи. Інтерфейси API забезпечують надзвичайно високий рівень абстракції, що робить їх відносно інтуїтивно зрозумілими і простими у використанні в процесі розробки.

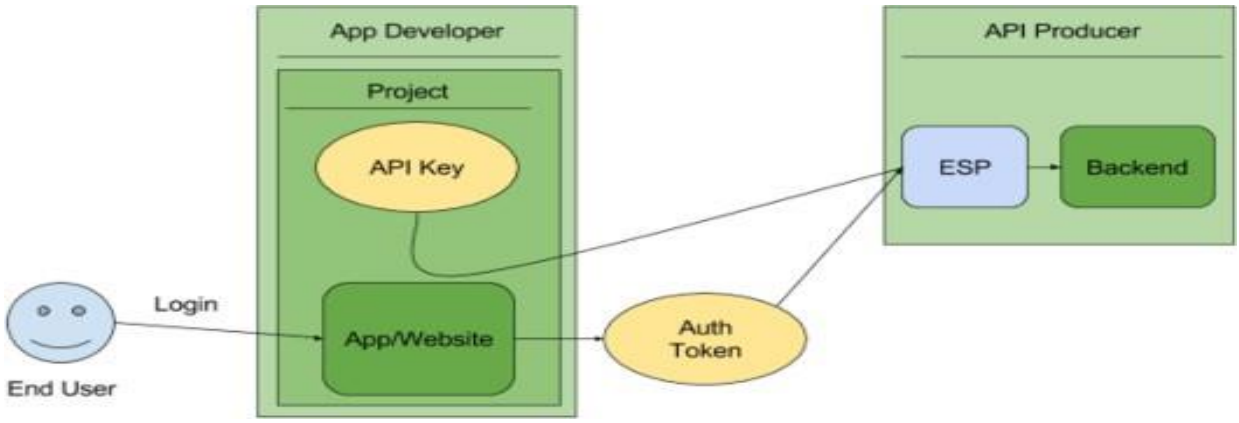

Рисунок 3.4. Приклад використання API.

Спрототіпірованний Navigation Drawer відповідає вимогам Material Design для Android і надає користувачеві додатки зручні можливості для навігації по розділах, також містить логотип організації, що поддёргівает приналежність додатки до цієї організації, яка займалася розробкою цього додатка для зручності користування користувача.

Розробляючи прототип розділу новин, ми застосували дизайн карток, який використовують багато додатки, в Android вони називаються CardView.

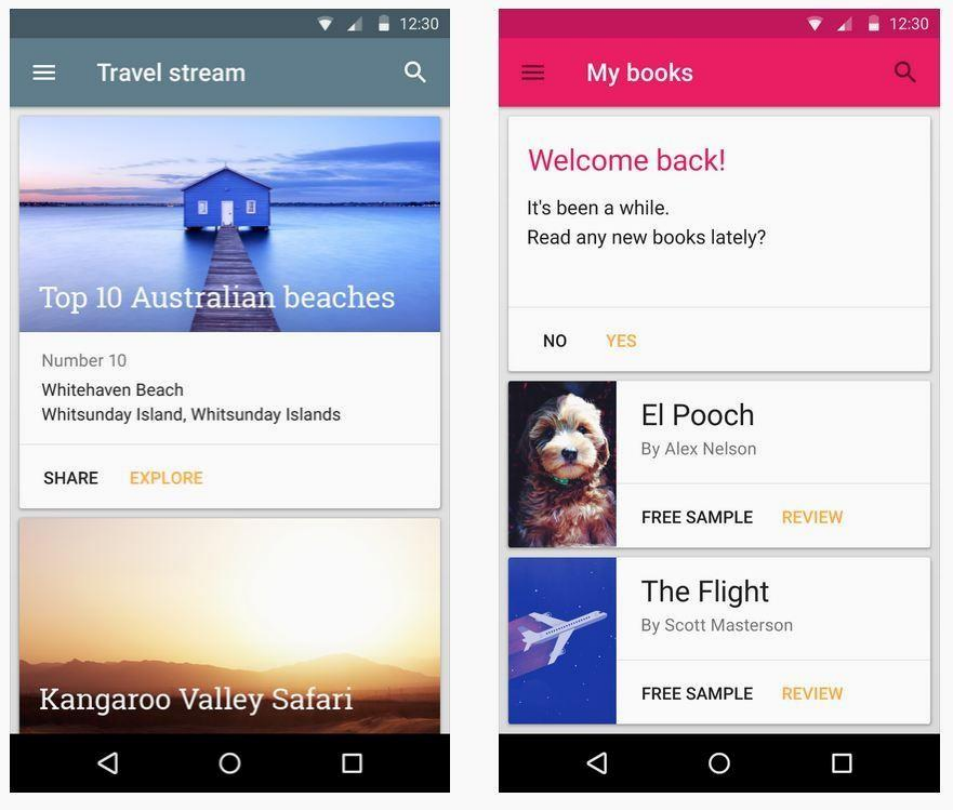

Рис 3.5. Приклад використання CardView.

Новий компонент CardView з'явився в Android Lollipop але завдяки бібліотеці сумісності доступний і для старих пристроїв. Дозволяє створювати красиву картку з тінню і закругленими кутами, який служить контейнером для інших компонентів. Звичайно ж, він можна застосувати не у всіх ситуаціях, але відмінно підходить при відображенні якогось пункту, що складається з картинки, заголовка і невеликий вступної інформації. Багато популярні додатки вже активно використовують CardView.

Рис 3.6. Прототип стрічки новин.

На дисплеї елементів в списках є дуже поширеною картини в мобільних додатках. Користувач бачить список елементів, і можна прокручувати їх. Якщо він вибирає один з елементів списку, це може оновити ActionBar або тригери детальну екран для вибору. Android надає ListView класу, який здатний відображати прокручувати список елементів. Ці елементи можуть бути будь-якого типу.

Розділ напрямки в зв'язку з його структурою буде володіти стандартним обліковим видом.

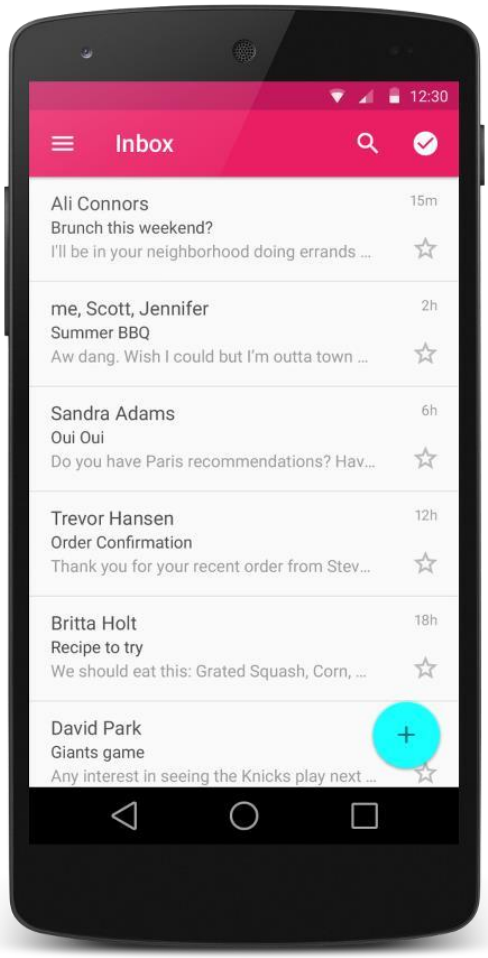

Рис 3.7. Приклад використання ListView.

Після вивчення структури розділу напрямки на сайті підприємства було створено прототип розділу в додатку.

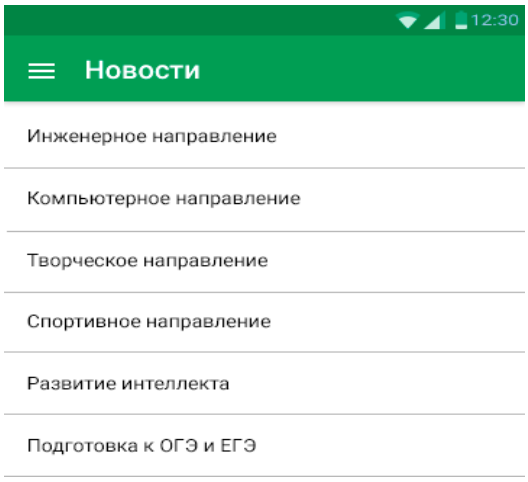

Рис 3.8 Розділ направлення.

Подальше прототипування розділу напрямки було направлено на вікно з інформацією про напрямок, був обраний варіант з поява нового вікна. В даному вікні буде відображатися інформація доступна користувачу. Спочатку повинна відображатися картинка з зображенням заданим на віддаленому сервері, знизу буде знаходитися докладна інформація в певному, зручному форматуванні. Для зручної навігації зверху-ліворуч перебувати кнопка назад для закриття вікна з докладною інформацією. Користувач також повинен розуміти, що перебувати в новому вікні для цього змінюється заголовок вікна на назву відкритого напрямки.

Рис 3.9. Детальна інформація про направлення.

4.СТВОРЕННЯ МОБІЛЬНОГО ДОДАТКУ

Стрімке зростання продажів пристроїв на базі Android відкриває видатні можливості перед розробниками додатків Android.

На платформі Android зараз працюють смартфони, планшети, електронні книги, роботи, реактивні двигуни, супутники NASA, ігрові приставки, холодильники, телевізори, камери, медичні пристрої,

«Розумні годинник», автомобільні інформаційні системи (для управління радіо, GPS, телефонами, термостатами і т. Д.) І багато інших пристроїв.

За останніми прогнозами, доходи від мобільних додатків (за всіма мобільних платформ) до 2022 року досягнуть 99 мільярдів доларів.

Одне з головних переваг платформи Android - її відкритість. Операційна система Android побудована на основі відкритого вихідного коду і знаходиться у вільному розповсюдженні. Це дозволяє розробникам отримати доступ до вихідного коду Android і зрозуміти, яким чином реалізовані методи, властивості і функції додатків.

Будь-який користувач може взяти участь, де можна отримати вихідний код Android, дізнатися про ідеологію, закладену в основу операційної системи з відкритим кодом, і отримати ліцензійну інформацію.

Відкритість платформи сприяє швидкому оновленню. На відміну від закритої системи iOS компанії Apple, доступною тільки на пристроях Apple, система Android доступна на пристроях десятків виробників обладнання (OEM, Original Equipment Manufacturer) і телекомунікаційних компаній по всьому світу. Всі вони конкурують між собою, що йде на користь кінцевому споживачеві.

Для розробки додатків для ОС Android потрібно встановити Android Studio.

Установка Android SDK. Інструменти для розробки Android SDK можна завантажити на сайті для розробників. При установці можна вибрати потрібні для розробки платформи і елементи SDK.

При розробці додатків Android використовується Java - один з найбільш поширених мов програмування. Використання Java стало логічним вибором для платформи Android, тому що це потужний, вільний і відкритий мову, відомий мільйонам розробників. Досвідчені програмісти Java можуть швидко освоїти Android-програмування, використовуючи інтерфейси Google Android API (Application Programming Interface) та інші розробки незалежних фірм.

Мова Java є об'єктно-орієнтованим, надає розробникам доступ до потужних бібліотекам класів, прискорюють розробку додатків.

Програмування графічного інтерфейсу користувача управляється подіями, які реагують на ініційовані користувачами події, такі як торкання екрана. Крім безпосереднього написання коду додатків можна скористатися середовищами розробки Eclipse і Android Studio, що дозволяють збирати графічний інтерфейс з готових об'єктів, таких як кнопки і текстові поля, перетягуючи їх в певні місця екрану, додаючи підписи і змінюючи їх розміри.

Ці середовища розробки дозволяють швидко і зручно створювати, тестувати і налагоджувати додатка Android.

Компоненти графічного інтерфейсу в Android називаються уявленнями (views). Вертикальне уявлення LinearLayout використовується для розміщення тексту і графіки так, щоб кожну виставу займало половину вертикального простору LinearLayout. Компонент LinearLayout також дозволяє розміщувати уявлення по горизонталі. Для виведення тексту в додатку буде використовуватися компонент TextView, а графіка буде відображатися в компоненті ImageView. Графічний інтерфейс, створений для зі стандартними програмами, містить компонент TextView.

Різні параметри цього компонента - текст, розмір шрифту, колір тексту, розмір щодо компонента ImageView в LinearLayout і т. Д. - налаштовуються у вікні властивостей середовища розробки. Потім ми перетягнемо компонент ImageView з палітри в макет графічного інтерфейсу і налаштуємо його властивості, включаючи джерело графічних даних і позицію в LinearLayout.

Мова XML (eXtensible Markup Language, тобто розширювана мова розмітки) є природним способом опису графічних інтерфейсів. Розмітка XML добре читається як людиною, так і комп'ютером; в контексті Android вона використовується для опису макетів використовуваних компонентів і їх атрибутів: розміру, позиції, кольору, розміру тексту, полів і відступів. Android Studio розбирає розмітку XML, щоб відобразити макет в макетному редакторі і згенерувати код Java, яка формує графічний інтерфейс на стадії виконання. Також файли XML використовуються для зберігання ресурсів програми: рядків, чисел, кольорів і т. д.

У кожної програми існує тема, яка визначає оформлення стандартних компонентів, які ми використовуємо. Тема програми вказується у файлі AndroidManifest.xml додатки. Ми можемо налаштувати різні аспекти теми (наприклад, складові колірної схеми), визначаючи ресурси в файлі styles.xml, що знаходиться в папці res / values додатки. Ресурсний файл style.xml містить стиль з ім'ям "AppTheme", посилання на який включається в файл AndroidManifest.xml додатки для призначення теми. Цей стиль також визначає батьківську тему, яка може розглядатися як аналог суперкласу в Java - новий стиль наслідує атрибути батьківської теми і їх значення за замовчуванням. Як і субклассов Java, стиль може перевизначити атрибути батьківської теми значеннями, адаптованими для конкретних додатків (наприклад, для використання в додатку фірмової колірної гами компанії). Ми використовуємо цю концепцію для настройки трьох кольорів, використовуваних в темі програми. Як згадувалося раніше, шаблони додатків Android Studio тепер включають підтримку бібліотек AppCompat, що дозволяють використовувати нові можливості Android в

старих версіях платформи. За замовчуванням Android Studio вибирає батьківську тему Theme.AppCompat.Light.DarkActionBar, одну з декількох стандартних тим в бібліотеці AppCompat - додатки, що використовують цю тему, відображаються на світлому тлі, а у верхній частині додатка розташовується темна панель програми. Всі теми AppCompat використовують рекомендації матеріального дизайну Google для оформлення графічних інтерфейсів. Установка JDK і JRE. Для розробки потрібно середовище виконання Java Runtime Environment (JRE), комплект розробника Java Development Kit (JDK), які можна завантажити з офіційного сайту Oracle.

Створення віртуального пристрою Android. Android tools включає в себе емулятор «Android Virtual Device» (AVD). Емулятор AVD дозволяє тестувати програми на віртуальному мобільному пристрої з ОС Android. Емулятор дозволяє створювати кілька віртуальних пристроїв з різними конфігураціями. Проект - це група пов'язаних файлів (наприклад, файли коду, ресурси і графічні файли), що утворюють додаток. Робота над додатком починається зі створення проекту. Щоб створити додаток в середовищі Android Studio для операційної системи Android. Відкриємо Android Studio. Для створення нового проекту треба перейти до пункту меню File -> New-> New Project .... Після цього у нас з'явиться діалогове вікно створення нового проекту:

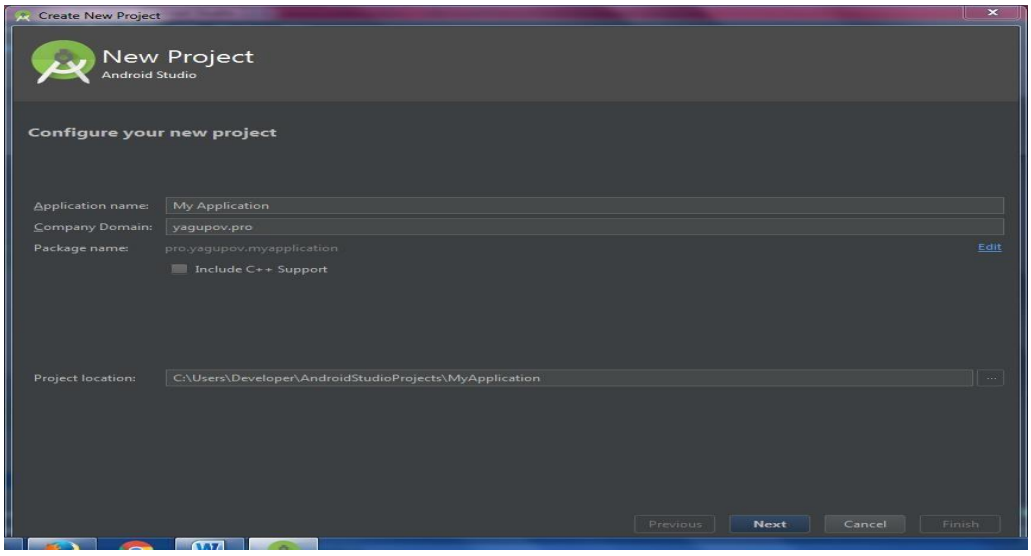

Рис 4.1. Вікно створення нового проекту.

На кроці Configure your new project майстра Create New Project введемо наступну інформацію.

1. Application Name: - назва програми.

2. Company Domain: - доменне ім'я веб-сайту компанії. створення можна використовувати ім'я example.com.

3. Package Name: - ім'я пакета Java для вихідного коду програми. Android і магазин Google Play використовують це ім'я в якості унікального ідентифікатора додатка, яке повинно залишатися постійним у всіх версіях програми, які ми будемо відправляти в магазин Google Store. Ім'я пакета зазвичай починається з доменного імені вашої компанії або установи, записаного в зворотному порядку. Наприклад, ми використовуємо доменне ім'я deitel.com, тому імена наших пакетів починаються із префікса example.com. Далі зазвичай слідує ім'я програми. За загальноприйнятим угодами в імені пакету використовуються тільки букви нижнього регістра без пробілів. За замовчуванням IDE вибирає ім'я пакета на підставі тексту, що вводиться в полях Application Name і Company Domain. Щоб змінити ім'я Package, клацніть на посиланні Edit праворуч від згенерованого імені пакета. Project Location: — шлях до папки на вашому комп’ютері, в якій буде зберігатись проект. За замовчуванням Android Studio розміщує папки нових проектів в вкладеній папці AndroidStudioProjects каталога облікового запису користувача. Ім’я папки проекту складається з ім’я проекту, із якого видаляються проблеми. Ми можемо змінити шлях до папки проекту; для цього введіть шлях або натисніть на кнопку (…) праворуч від поля и виберіть папку для зберігання проекта. Після того як папка буде вибрана, клацніть по кнопці OK, а потім перейдіть до наступного кроку кнопкою Next. Project Location: - шлях до папки на вашому комп'ютері, в якій буде зберігатися проект. За замовчуванням Android Studio розміщує папки нових проектів у вкладеній папці AndroidStudioProjects

каталогу облікового запису користувача. Ім'я папки проекту складається з імені проекту, з якого віддаляються прогалини. Ми можемо змінити шлях до папки проекту; для цього введіть шлях або клацніть на кнопці (...) праворуч від поля і виберіть папку для зберігання проекту. Після того як папка буде обрана, клацніть на кнопці OK, а потім перейдіть до наступного кроку кнопкою Next.

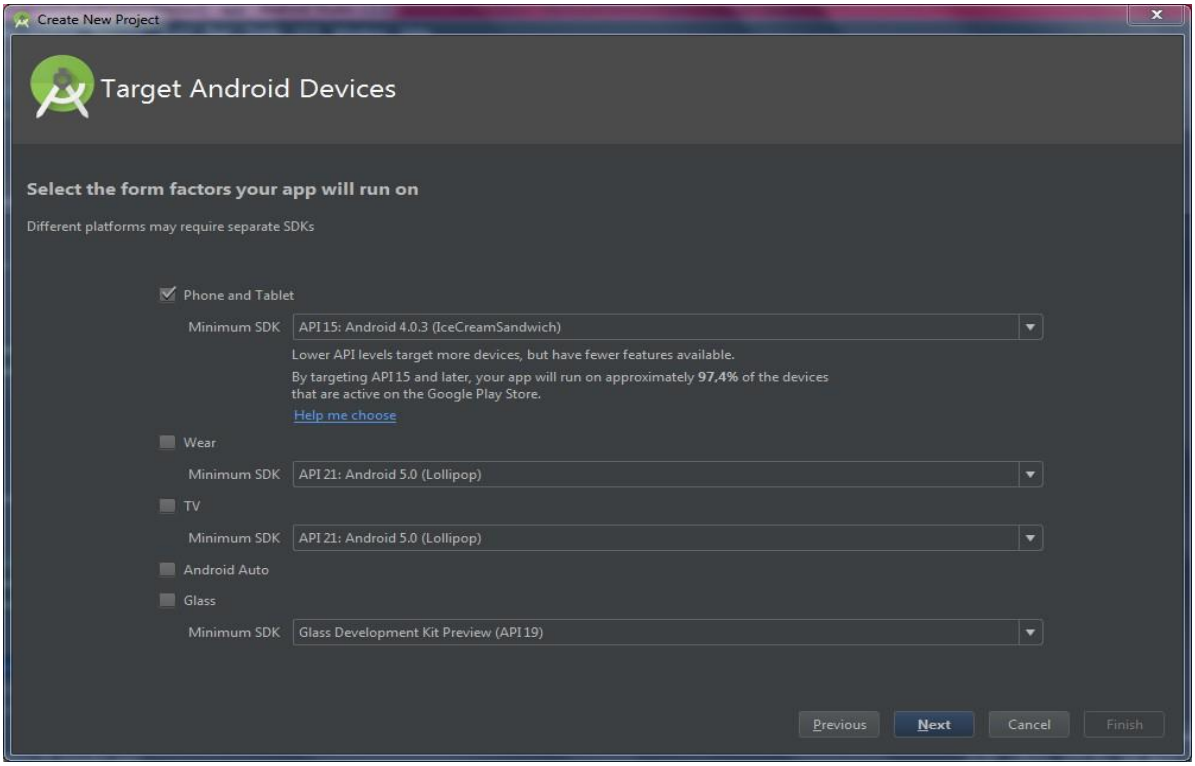

Рис 4.2. Вибір версії Android.

На цьому кроці буде запропоновано встановити мінімальну підтримувану версію проекту. За замовчуванням встановлюється версія Android 4.0.3, що покриває майже 97% пристроїв Android. Залишимо за замовчуванням і натиснемо на кнопку Next.

На наступному кроці треба вибрати Activity, яка буде використовуватися за замовчуванням для головного екрану програми:

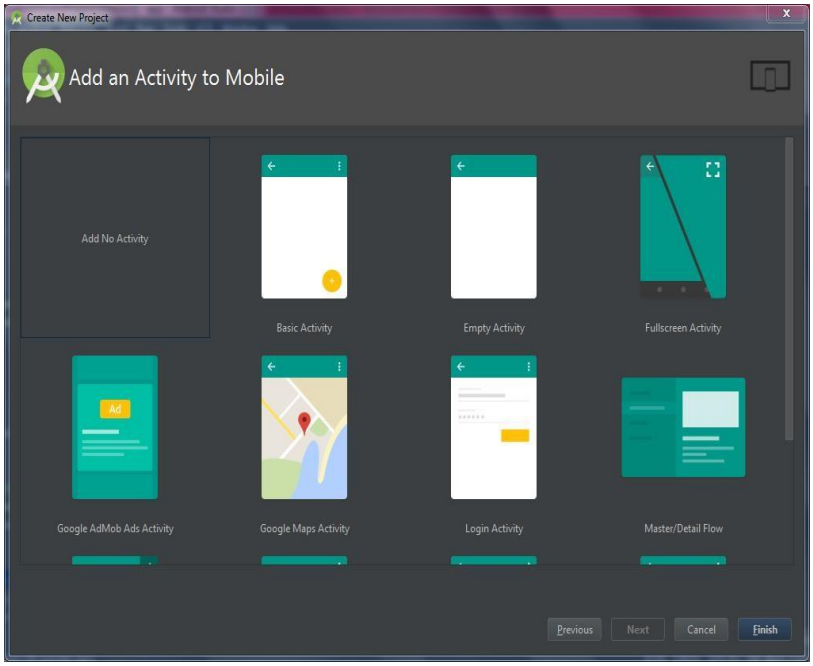

Рис 4.3 Вибір головної Activity.

У нашому додатку ми не будемо додавати Activity звичайним чином, тому що відбувається не ефективна генерація програмного коду. Пізніше буде створена базова реалізація батьківської Activity для інших вікон програми, для дотримання принципу DRY (не повторювати).

Розглянемо структуру проекту програми під ОС Android, яка створюється за замовчуванням.

Проект може включати різні модулі. І все модулі описуються файлом setting.gradle.

І якщо ми подивимося на структуру проекту, то весь значимий код - файли інтерфейсу, класи java і т.д. у нас за замовчуванням знаходяться в папці (модулі) app.

Файл build.gradle містить інформацію, яка використовується при побудові проекту. Кожен модуль має свій файл build.gradle, який визначає конфгурацію побудови проекту, специфічну для даного модуля. Так, якщо ми подивимося на вміст папки app,

то, як раз знайдемо в ній такий файл. На початковому етапі дані файли не настільки важливі, досить лише розуміти, для чого вони потрібні. За замовчуванням кожен проект включає один модуль - app. Власне весь код, з яким ми будемо працювати, розташовується всередині цього модуля.

У цьому модулі ми можемо побачити кілька папок і файлів, з яких для нас найважливішими є:

каталог libs - призначений для зберігання бібліотек, використовуваних додатком

каталог src - призначений для зберігання вихідного коду. Він містить ряд підкаталогів. Тексти програм розташовуються в папці main.

Папка main має складну структуру:

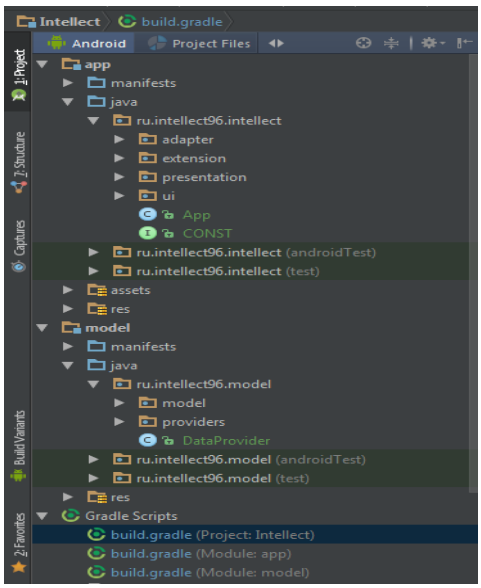

Рис 4.4 Структура Android проекта.

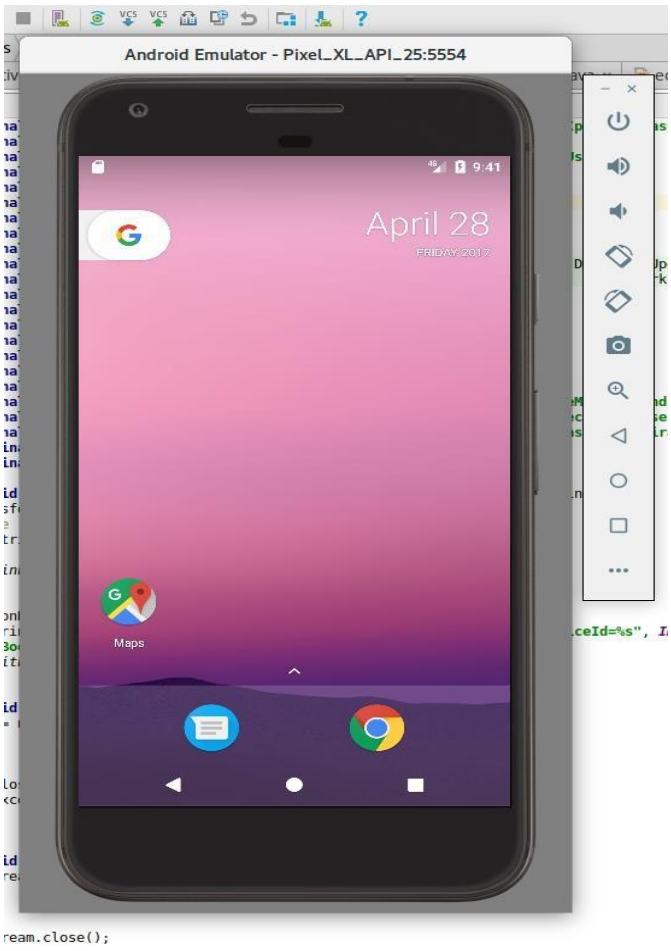

Рисунок 4.5. Віртуальний девайс в Android Studio.

5.ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Система самоосвіти буде складатися с декількох вікон керування. Головним буде базове вікно вибору курсів за певною спеціальністю. Також буде вікно самого курсу, яке наповнюється контентом в залежності від обраного курсу. Система матиме також декілька допоміжних вікон для взаємодії програмного продукту і користувача. Обрана структура буде простою та інтуїтивно зрозумілою для користувача за рахунок чіткої та зрозумілої організації модулів.

На рисунку наведена схема структури системи, на якій розташовані всі програмні модулі.

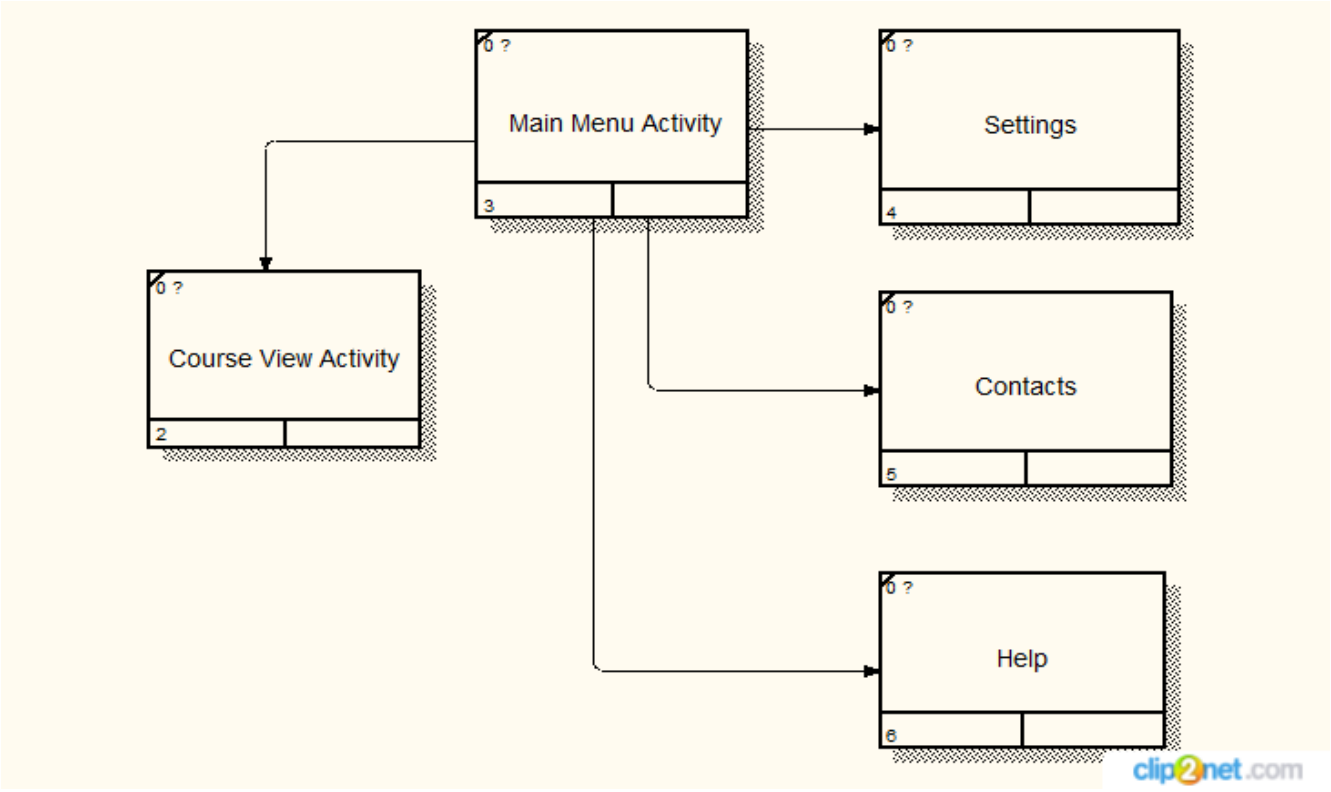

Рисунок 5.1 Схема структури системи.

5.1.Опис функціональності системи

Система самоосвіти містить у собі два актора, які взаємодіють із системою а також один з одним. Перший актор – це користувач який взаємодіє із системою за допомогою смартфона. Другий автор – це адміністратор додатку який наповнює систему контентом на надає необхідну технічну підтримку користувачу.

На рисунку представлена діаграма прецедентів, яка описує функції та дії акторів у системі дистанційного навчання.

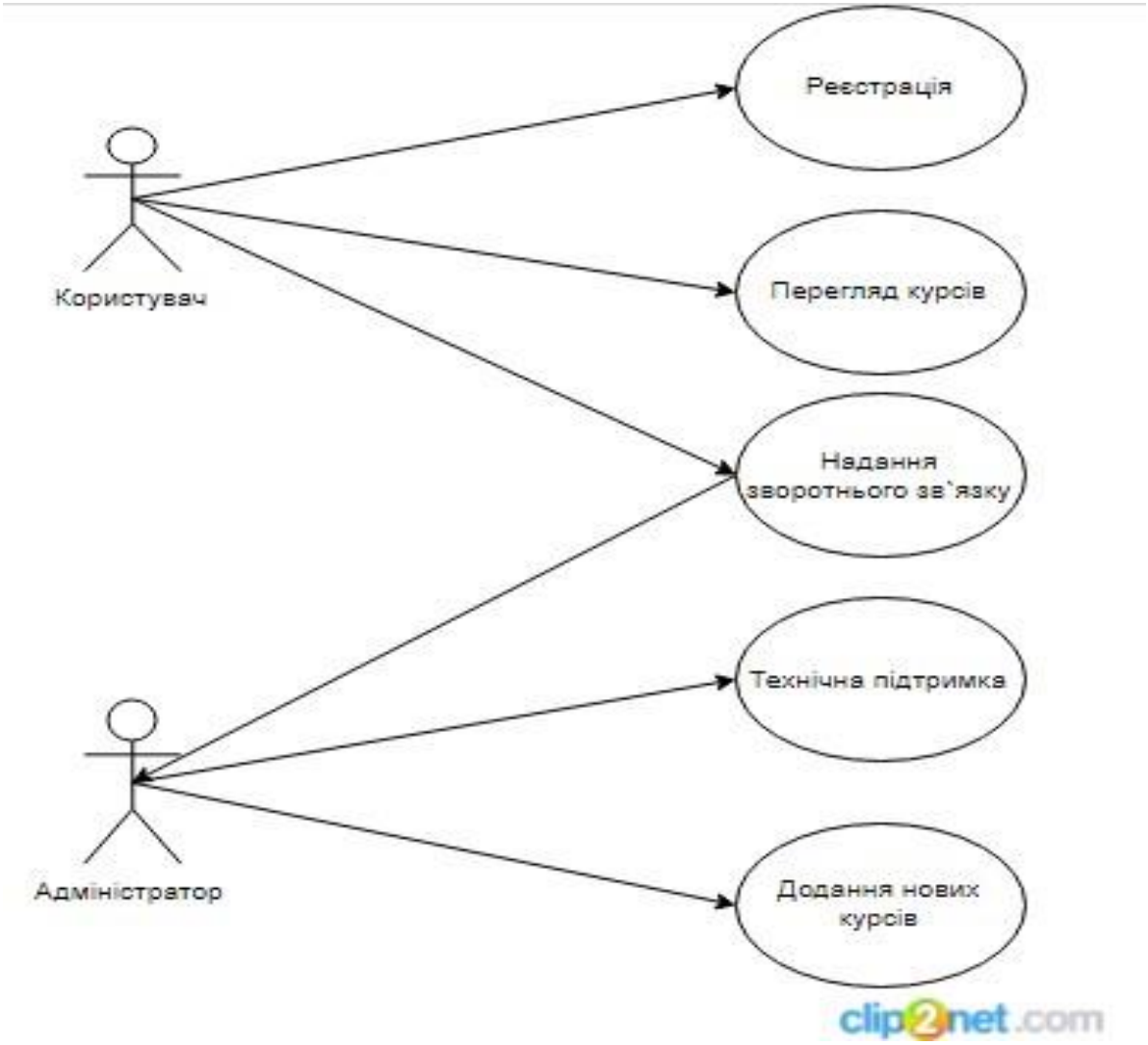

Рисунок 5.2 Діаграма прецедентів системи.

ВИСНОВКИ

Мета цієї дипломної роботи полягає в створенні мобільного додатка для розвиваючого освітнього центру, для більш ефективного навчання користувачів, підвищення інформованості користувачів в предметах які вони хочуть вивчати, а також закріплення практичних навичок з тих предметів які вони хотіли б вивчати в майбутньому.

Для досягнення поставленої мети були сформульовані наступні завдання:

1. Провести розрахунок економічних показників ефективності створення мобільного застосування.

2. Провести пошук і аналіз способів підвищення потоку клієнтів за допомогою мобільного додатку.

3. Провести аналіз вимог до мобільного додатку.

4. Спроектувати призначений для користувача інтерфейс.

5. Створити мобільний додаток.

У цій дипломній роботі був проведений аналіз і пред'явлені вимоги до мобільного додатку, проаналізовані функціональні вимоги до інтерфейсу, за якими був створений прототип призначеного для користувача інтерфейсу. За допомогою, якого в подальшому проводилася розробка мінімальної версії мобільної програми для ОС Android с застосуванням новітніх технологій в області мобільних технологій.

При вирішенні задачі були вивчені теоретично методи пред'явлення і аналізу вимог до програмного забезпечення, призначеного для користувача інтерфейсу. Отримано практичний навик роботи з графічним редактором Adobe Photoshop, в якому був розроблений для користувача інтерфейс відповідає вимогам стандартів. Було розроблено програму, яку відповідає пред'явленим вимогами дипломною

роботою, і функціонує відповідно до описаним вимогам. Також були закріплені і отримані нові навички роботи в середовищі розробки Android Studio.

Таким чином, завдання були вирішені в повному обсязі, мета досягнута.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Брюс Эккель — Философия Java [Електронний ресурс]. — 2019. — Режим доступу: https://bit.ly/2JbSgHe.

2. Katty Sierra — Head first Java, 2nd edition [Електронний ресурс]. — 2005. — Режим доступу: https://www.amazon.com/dp/0596009208.

3. Нейл Тереза — Мобильная разработка. Галерея шаблонов [Електронний ресурс]. — 2013. — Режим доступу: https://www.ozon.ru/context/detail/id/19701938/

4. Ян Клифтон — Проектирование пользовательского интерфейса в Android [Електронний ресурс]. — 2017. – Режим доступу:https://www.ozon.ru/context/detail/id/140152223/.

5. П. Дейтел, Х. Дейтел, А. Уолд — Android для разработчиков [Електронний ресурс]. — 2016. — Режим доступу: https://www.ozon.ru/context/detail/id/136331151/

6. *Bill Phillips, Chris Stewart & Kristin Marsicano* — **Android Programming: The Big Nerd Ranch Guide** [Електронний ресурс]. — 2015. — Режим доступу: https://www.amazon.com/Android-Programming-Nerd-Ranch-Guide/dp/0134171454.