**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**
**„КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. ІГОРЯ СІКОРСЬКОГО»**
**Теплоенергетичний факультет**

**Кафедра автоматизації проектування енергетичних процесів і систем**

.

# Дипломна робота

**на тему:** ”Побудова системи дистанційного навчання ”

**Студент групи ТІ-**51 Бондаренко М.Є. _____
(прізвище, ім’я, по батькові) (підпис)

**Керівник роботи** доцент Ходаковський О.В. _____
(вчені ступінь та звання, прізвище, ініціали) (підпис)

**Кількість сторінок** 80

**Кількість ілюстрацій** 24

**Київ – 2019**

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено

Завідувач кафедри

_____ О.В. Коваль
(підпис) (ініціали, прізвище)

“___”_____2019р.

# ДИПЛОМНА РОБОТА
**на здобуття ступеня бакалавра**

з напряму підготовки 6.050103 “ Програмна інженерія “

на тему «Побудова системи дистанційного навчання»

Виконав: студент IV курсу, групи ТІ-51

Бондаренко Максим Євгенійович _____
(прізвище, ім’я, по батькові) (підпис)

Керівник доцент, Ходаковський Олексій Володимирович _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Київ – 2019

**Національний технічний університет України**
**“Київський політехнічний інститут імені Ігоря Сікорського”**

**Національний технічний університет України**
**“Київський політехнічний інститут імені Ігоря Сікорського”**

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший , бакалаврський

Напрям підготовки 6.050103 “ Програмна інженерія”

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ О.В. Коваль
(підпис)
”___”_____2019р.

# ЗАВДАННЯ

**на дипломну роботу студенту**

Бондаренко Максиму Євгенійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи «Розробка електронної системи дистанційного навчання»
керівник роботи Ходаковський Олексій Володимирович, к.т.н. доцент
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від 22 05 2019р. № 1325-с

2. Строк подання студентом роботи 10 червня 2019р.

3. Вихідні дані до роботи розроблений модуль написаний мовою Java під платформу Android

4.Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити) виконати аналіз існуючих систем дистанційного навчання, розробити програмну частину системи дистанційного навчання, здійснити програмну реалізацію спроектованої системи дистанційного навчання, провести тестування і налагодження застосунку.

5. Перелік ілюстративного матеріалу титульний аркуш, постановка задачі, існуючі системи дистанційного навчання, архітектура розроблюваної системи, функціонал клієнтського додатку, висновки.

6. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| | | | |

7. Дата видачі завдання ”_1_”___грудня_____2018__р.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів виконання дипломної роботи | Термін виконання етапів роботи | Примітки |
|---|---|---|---|
| 1. | Вивчення та аналіз задачі | 12.12.2018 | |
| 2 | Розробка архітектури та загальної структури системи | 08.02.19 | |
| 3. | Розробка структур окремих підсистем | 12.02.19 | |
| 4. | Програмна реалізація системи | 26.02.19 | |
| 5. | Оформлення пояснювальної записки | 07.06.2019 | |
| 6. | Захист програмного продукту | 18.05.2019 | |
| 7. | Передзахист | 01.06.2019 | |
| 8. | Захист | 27.06.2019 | |

Студент _____ Бондаренко М.Є._____
(підпис) (прізвище та ініціали,)

Керівник роботи _____ ____Ходаковський О.В.___
(підпис) (прізвище та ініціали,)

# АННОТАЦІЯ

Дипломну роботу виконано на 60 аркушах, вона містить 0 додатків та перелік посилань на використані джерела з 7 найменувань. У роботі наведено 24 рисунка. 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 60 sheets, it contains 0 applications and a list of references to used sources of 7 titles. The paper presents 24 figures. The implementation of 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.

# ЗМІСТ

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ПЗ – програмне забезпечення

ПК – персональний комп’ютер

ОС – операційна система

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

XML (англ. extensible markup language) – стандарт побудови мов розмітки ієрархічно структурованих даних для обміну між різними застосунками, зокрема, через мережу інтернет

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

SDK (англ. software development kit) – набір засобів розробки, який довзволяє спеціалістам з програмного забезпечення створювати додатки для певного пакета програм, апаратної платформи, комп’ютерної системи, ігрових консолей, операційних систем

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

APK (англ. android package) – формат архівних виконуваних файлів додатків, який включає в себе весь код додатку, ресурси, активи, файл маніфесту та нативні бібліотеки

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

AVD (англ. android virtual device) – спеціальний пристрій для емулятора Android, що має визначені властивості, для перевірки роботи розробленого продукту та його відладки

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

автоматизації збірки, відладчик

JVM (англ. java virtual machine) – віртуальна машина, на якій виконуються всі мови платформи Java

API (англ. Application Programming Interface) – набір готових класів, процедур, функцій, структур і констант, що надаються додатком (бібліотекою, сервісом) або операційною системою для використання у зовнішніх програмних продуктах

JIT (англ. Just-in-time compilation) – технологія збільшення продуктивності програмних систем, що використовують байт-код, шляхом компіляції байт-коду в машинний код або в інший формат безпосередньо під час роботи програми

GUI (англ. graphical user interface) – різновид призначеного для користувача інтерфейсу, в якому елементи інтерфейсу (меню, кнопки, значки, списки і т. п.), представлені користувачеві на дисплеї, виконані у вигляді графічних зображень

DPI (англ. dots per inch) – кількість точок на квадратний дюйм, означає роздільну здатність екрану девайса

# ВСТУП

З розвитком технологій використання смартфонів у повсякденному житті стає дедалі поширенішим. В наш час майже у кожної дорослої людини є свій смартфон. Різні мобільні додатки дозволяють значно спростити наше життя та вирішувати певні проблеми. Вони мають застосування майже у всіх областях нашого життя. Навчання та освіта є невід'ємною складовою нашого життя. Тому поява додатків, які дозволяють отримувати нові знання безпосередньо зі смартфону було лише питанням часу.

Сучасні інструменти розроблення мобільних додатків дозволяють розробникам реалізувати велику кількість ідей для різних цілей. В тому числі і використання смартфона в ролі засобу для здобуття нових знань. Ці інструменти дозволяють відтворювати інформацію у різних формах: текст, відео, звук, тощо.

На сьогоднішній день є певна кількість додатків в основі яких лежить система дистанційного навчання. Це надає вам змогу отримати знання з певної теми. Проте всі вони реалізовані на прикладі загальних курсів з тієї чи іншої області. А для опанування фундаментальних знань з обраної спеціальності необхідно пройти цілий цикл курсів, які пов'язані між собою та складають цілу картину знань. Саме таким чином реалізований процес навчання у вищих навчальних закладах. Отже, ми приходимо до висновку, що існує потреба у створенні програмного продукту який би поєднував переваги традиційного освітнього процесу та нових технологій.

Тому, було вирішено дослідити можливість створення системи дистанційного навчання, яка б дозволяла студентам (а також усім охочим) в реальному часі вивчати програму тієї чи іншої спеціальності згідно з планом навчального процесу за допомогою власного смартфону.

# 1. ЗАДАЧА ПОБУДОВИ СИСТЕМИ ДИСТАНЦІЙНОГО НАВЧАННЯ ДЛЯ ВИКОРИСТАННЯ В ВИЩИХ НАВЧАЛЬНИХ ЗАКЛАДАХ

Останнім часом розвиток смартфонів набуває все більших обертів. С кожним днем їх операційні системи стають досконалішими, мають все більше функцій та розширюють коло можливостей для реалізації додатків з використання найновіших інструментів розробника. Перші мобільні додатки почали з’являтися майже одночасно з появою операційних систем для мобільних пристроїв. Можливості та функціональність перших додатків були на досить низькому рівні. З кожною новою версією операційної системи кількість доступних інструментів для розробника зростала, Це дало змогу захоплювати все більше областей для застосування створених продуктів, а отже й інструментарій цих додатків ставав дедалі ширшим. Це сприяло створенню додатків для використання у різних областях нашого життя. На сьогоднішній день існують безліч додатків, які об’єднуються в певні категорії, наприклад: відео плеєри, редактори, навігатори, онлайн магазини, додатки для соціальних мереж. Вони використовуються повсюди. Майже кожен банк має власний клієнтський додаток, що значно спрощує взаємодію банків з фізичними та юридичними особами. Більше того сьогодні є навіть банки, які повністю перенесли свою діяльність в всесвітню мережу. В таких банках додаток є не допоміжним інструментом для клієнта, а основним способом взаємодії банка з клієнтами. Ще однією областю яка майже повністю перейшла в мобільні додатки є пасажирські перевезення. Більшість служб таксі мають додатки через які клієнт може замовити автомобіль, спостерігати за напрямком руху водія, а також оцінювати сервіс. Також додатки мають великі мережі супермаркетів. Через ці додатки можна переглянути весь асортимент товарів, діючи програми лояльності, актуальні знижки, а також здійснити замовлення товарів онлайн. Ці замовлення доставляють кур’єрські служби, які в свою чергу також мають власні додатки. Область застосування додатків дуже 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 девайс. 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.

Додатки такого типу розраховані на декілька категорій користувачів. Одна категорія користувачів користуються ними для розширення власного кругозору або поповнення знань в певних питаннях. Інша категорія користувачів, яка з певних причин не має можливості здобувати освіту в вищих навчальних закладах, використовує ці додатки як основне джерело нових знань та інструмент освоєння нової професії. Саме для такої категорії користувачів необхідно отримувати ці знання за програмою, яка створена для тієї чи іншої спеціальності. Це у свою чергу підвищує якість такого типу освоєння нових знань.

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

Отже, основною метою дослідження є створення програмного застосунка, який повинен реалізувати наступні функції:

- створення курсів на наповнення їх навчальними матеріалами за заданою структурою
- забезпечення користувача контентом , яким наповнений додаток
- сповіщення користувача про оновлення курсу
- структуризація курсів за спеціальностями
- зручний та зрозумілий графічний інтерфейс для комфортного користування

Отже, для виконання основної задачі необхідно:

- аналіз інструментів для реалізації додатку для дистанційного навчання в вищих навчальних закладах;
- аналіз існуючого програмного забезпечення для дистанційного навчання в вищих 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 СИСТЕМ ДИСТАНЦІЙНОГО НАВЧАННЯ

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

## 2.1. Поняття дистанційного навчання

На сьогоднішній день користувачам доступні різноманітні системи для навчання та вивчення певних матеріалів. Актуальність дистанційного навчання серед них важко переоцінити. Дистанційне навчання – взаємодія вчителя та учня на відстані, відображаюча всі компоненти навчального процесу (цілі, зміст, методи, організаційні форми, засоби навчання), яка реалізована з використанням інтерактивних методів навчання. Розвиток такого типу навчання було почато ще в 60-х роках минулого століття. З того часу дистанційне навчання значно збільшило свою область застосування та суттєво змінило методи взаємодії між учителем та учнем. Сучасне дистанційне навчання засноване 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 дистанційного навчання: в режимі онлайн та в режимі офлайн.

Дистанційне навчання має ряд суттєвих переваг :

- далекосяжність – студенти не обмежені відстанню та мають можливість навчатися незалежно від місця проживання
- гнучкість – студенти мають можливість отримувати знання в підходящий їм час в зручному для них місці

- економічність – значно знижуються витрати на дальні поїздки до місця навчання

В більшості випадків в наші дні дистанційне навчання реалізоване у якості додатків на мобільні пристрої та веб платформ. Бо при цьому вони залишаються 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 є вузьконаправлені 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 у вигляді 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 по

обраному курсу. Після завершення одного курсу, користувач може приступати до наступного. Курси формуються у вигляді списку і розбиваються на різні категорії. Наразі у вільному доступі користувачам доступні різні додатки такого типу і їх вибір доволі широкий.

## 2.2. Аналіз систем дистанційного навчання у вигляді онлайн курсів

Системи дистанційного навчання у вигляді онлайн курсів є найбільш вдалою реалізацією дистанційного навчання для мобільних девайсів. Користувач отримує знання в інтерактивному форматі та у різному вигляді, що значно покращує сприйняття інформації. У процесі пошуку інформації, та аналізу існуючих рішень систем дистанційного навчання, було виявлено, що на даний момент існуючі рішення не виконують певні вимоги і не надають можливості проходження всіх курсів певної спеціальності вищого навчального закладу.

### 2.2.1. Система дистанційного навчання Prometheus

Система Prometheus є українським громадським проектом доспутних онлайн курсів. Даний ресурс було створено в 2014 році. Саме в той час була відкрита реєстрація на першу четвірку відритих онлайн курсів цього ресурсу, що були розроблені викладачами з трьох провідних українських вищих навчальних закладів: КПІ ім. Ігоря Сікорського, КНУ ім. Тараса Шевченка а також Києво-Могилянської академії. В перші пів року з початку запуску на сайті проекту вже було зареєстровано більш 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

цикл курсів для школярів, які готуються до ЗНО.

Основною задачею цього ресурсу є надання безкоштовного доступу через мережу Інтернет до низки курсів всім охочим, а також впровадження можливості поширювати та публікувати власні курси провідним вищим навчальним закладам та підприємствам.

Система Prometheus представляє собою ресурс курсів в режимі онлайн доступу, які структуровані в системі за циклами. Це поєднує в собі декілька важливих аспектів. Перший із них це заздалегідь записані відео лекції, які можна переглянути в будь-який зручний для користувача час. Другий аспект це різноманітні інтерактивні завдання, які надають можливість перевірити, як користувачі засвоїли матеріал лекції. Третій аспект це форум для користувачів, на якому вони мають можливість ставити запитання, а викладачі курсів або інші слухачі відповідати на ці запитання. Четвертим аспектом є можливість отримати електронний сертифікат за підписом викладача, успішно пройшовши обраний курс. І п’ятий аспект це відсутність будь-яких обмежень для слухачів курсів.

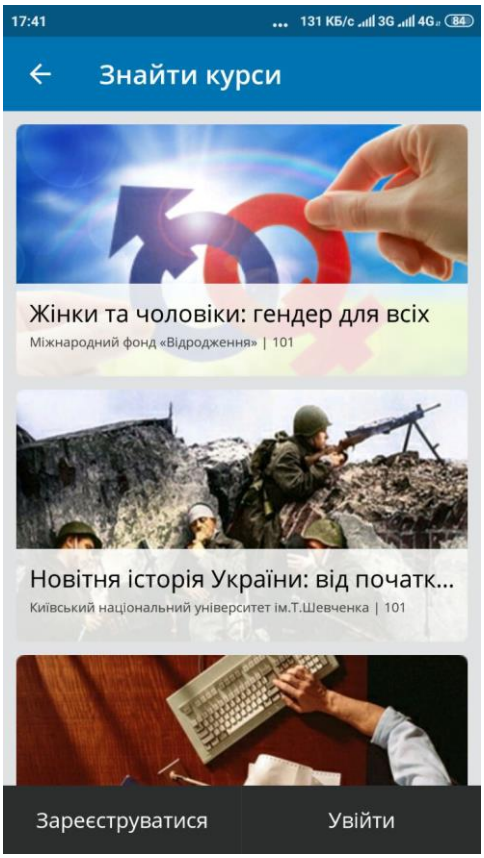


Рисунок 2.1 –Інтерфейс мобільного додатку Prometheus

Додаток має наступні функції:

- проходження курсів онлайн за допомогою бездротової мережі або мобільного зв’язку;
- завантаження відеолекцій для навчання в будь-який час без доступу до мережі Інтернет;
- перевірка знань за допомогою базових інтерактивних завдань та тестів у зоді проходження курсу;
- перегляд новин та роздаткових матеріалів курсів;
- проходження курсів в будь-якому місці за власним графіком та у зручний для користувача час;
- навчання у найкращих викладачів українських та закордонних вищих навчальних закладів;

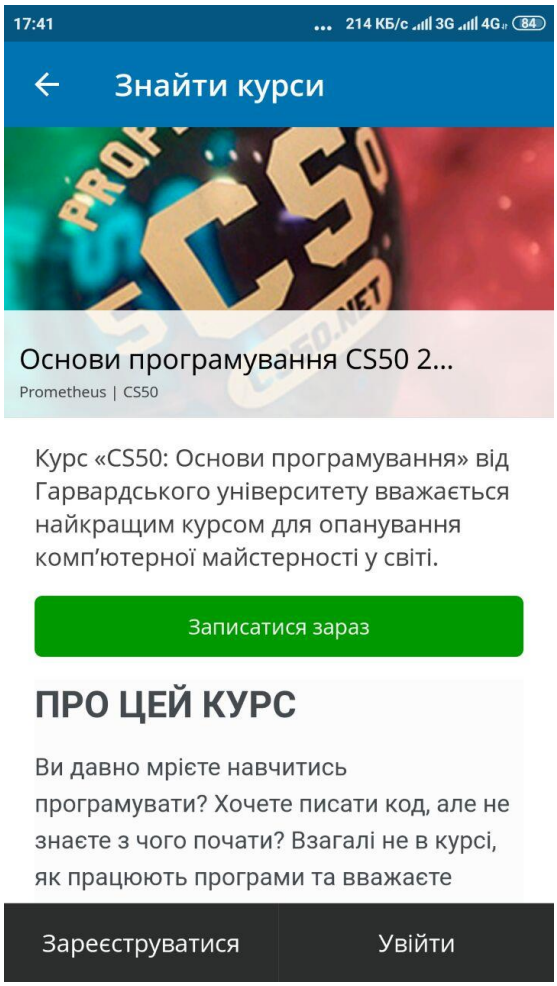


Рисунок 2.2 – Загальний вигляд обраного курсу Prometheus

### 2.2.2. Система дистанційного навчання TED

На ресурсі відео лекцій та конференцій TED зібрано виступи найвідоміших науковців світу про результати їх досліджень з усього світу. На цій платформі користувач може знайти важливі дослідження на дуже велику кількість різноманітних тем, які впливають на наше життя, таких як наприклад математичне пояснення почуттів кохання та привабливості або ж теорія суперструн та розвиток біотехнологій.

Цей ресурс також представлений у вигляді мобільного додатку, в якому містяться тисячі відео виступів з конференцій TED, в яких науковці діляться результатами своїх досліджень. Завдяки високошвидкісному інтернету користувач може завантажити на власний мобільний пристрій обраний виступ та переглянути його дорогою на роботу чи навчання. Це дозволяє витратити час в дорозі з користю для власного кругозору та отриманням нових цікавих знань. Якщо ви не можете обрати виступ для перегляду, додаток має функцію яка сама підбере вам виступ. Користувачу лише залишається визначитися з темою виступу та кількістю часу, яку він може витратити на перегляд виступу.

Для тих користувачів, які не володіють англійською мовою в додатку міститься функція, яка дозволяє увімкнути субтитри, які доступні більш ніж на 90 мовах світу. Переглядати виступи можна як в режимі онлайн, так і завантажити виступ на додаток для перегляду без доступу до мережі. Також можна завантажити виступ у вигляді аудіофайлу, для зменщення розміру файлу. Також додаток надає широкі можливості для сортування виступів за такими критеріями, як час, теги, категорія, та популярність. В додаток вмонтовано радіостанцію конференції, яку можна слухати прямо з додатку. Користувач може поділитися обраними виступами с друзями або колегами надіславши їх по електронній пошті або через соціальні мережі.

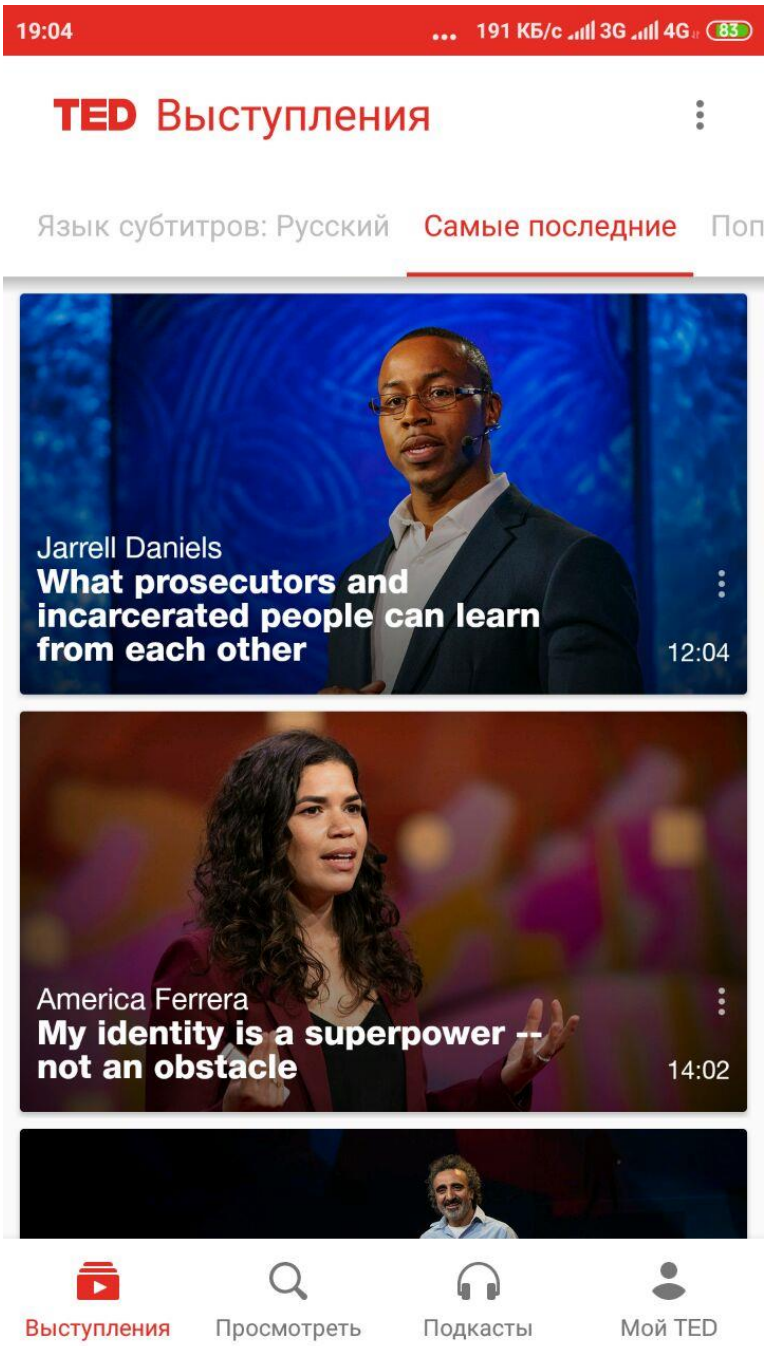


Рисунок 2.3 – Інтерфейс мобільного додатку TED

Додаток має наступні функції:

- перегляд всієї відео-бібліотеки TED Talks з субтитрами на більш ніж 100 мовах світу;
- прослуховування епізодів підкасту Radio Hour TED, спільного виробництва NPR та TED;
- вибір виступів за певною тематикою з створених плейлистів;

- завантаження відеолекцій або аудіозаписів на пристрій для відтворення в режимі офлайн;
- можливість додати виступ в закладки для відтворення пізніше;
- можливість авторизації свого профілю TED для синхронізації матеріалів на всіх пристроях;
- відтворення виступів на мобільному пристрої у фоновому режимі в форматі аудіозапису;
- створення спеціального списку відтворення для користувача, розробленого відвідно до встановлених меж часу;
- можливість відтворення виступів з додатку на телевізор;

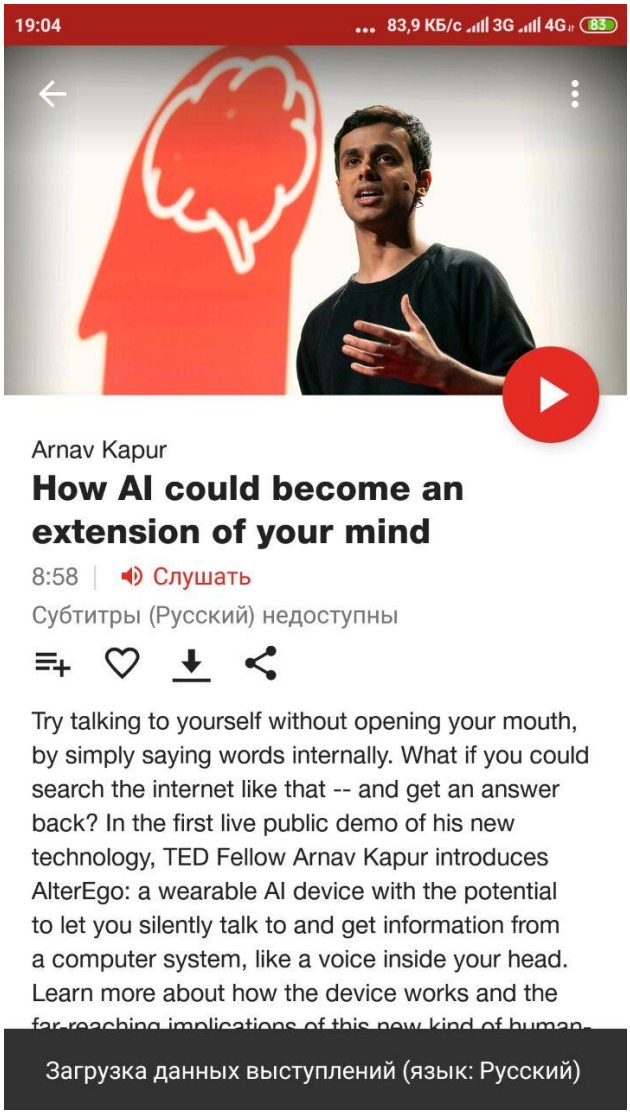


Рисунок 2.4 – Загальний вигляд обраного курсу TED .

### 2.2.3. Система дистанційного навчання Stepik

Платформа Stepik є російською освітньою платформою а також конструктором безкоштовних курсів та уроків в режимі онлайн. Система надає можливість зареєстрованим користувачам конструювати онлайн курси та уроки, використовуючи 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 навчання користувач має можливість спілкуватися з іншими користувачами та викладачами на форумі ресурсу. На платформі є велика кількість вже готових курсів з різних тем, наприклад математики, біології, розробки та навіть біоінформатики. На даний момент ця освітня платформа має більше 1 мільйона користувачів.

Платформу Stepik було створено Миколою Вяххі у 3013 році. Історія платформи почалась з того, що він разом з компанією JetBrains та лабораторією алгоритмічної біології створив безкоштовні курси з біоінформатики, які пізніше і лягли в основу молодої платформи.

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

Автор курсу, що був ним розроблений та розміщений на платформі має право зберігати за собою авторське право на цей курс, також має можливість без обмежень створювати навчальні матеріали у вигляді уроків та курсів та використовувати їх у власному освітньому процесі поза межами ресурсу, експортувати та будь-які інші ресурси чи платформи, мати доступ до статистики проходження його курсів. Всі

матеріали, які розміщені на платформі та 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 результаті чого їм буде видано відповідний сертифікат.

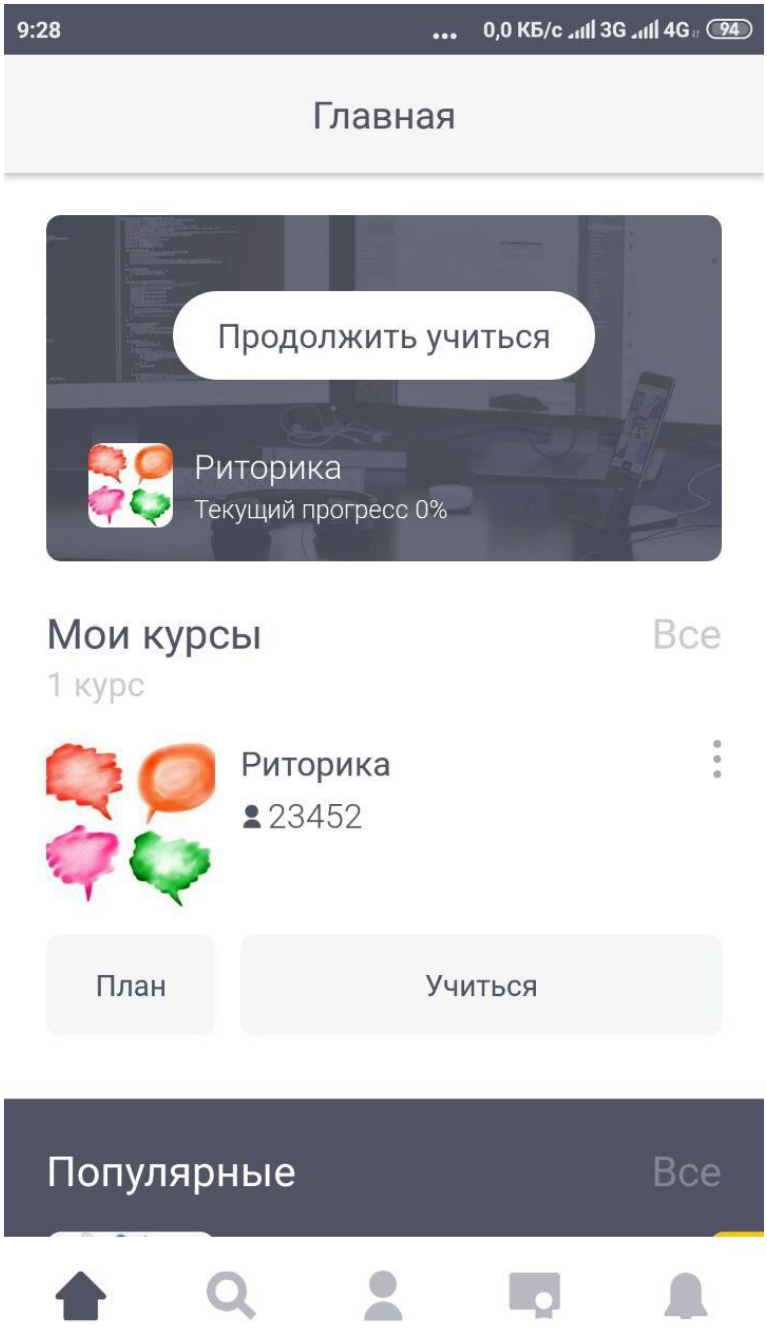


Рисунок 2.2 – Інтерфейс мобільного додатку Stepic

Додаток має наступні функції:

- проходження курсів онлайн за допомогою бездротової мережі або мобільного зв’язку;
- обговорення завдань та спілкування з іншими студентами беручи участь в розділі коментарів;
- можливість імпорту термінів проходження курсів в календар, що встановлений на мобільному пристрої;
- завантаження відеолекцій або аудіозаписів на пристрій для відтворення в режимі офлайн;
- можливість додати матеріал в закладки для відтворення пізніше;
- регулювання швидкості відео лекцій для оптимального навчання;
- встановлення нагадування при необхідності мотивувати себе та регулярно вивчати матеріал, щоб покращувати свої показники;

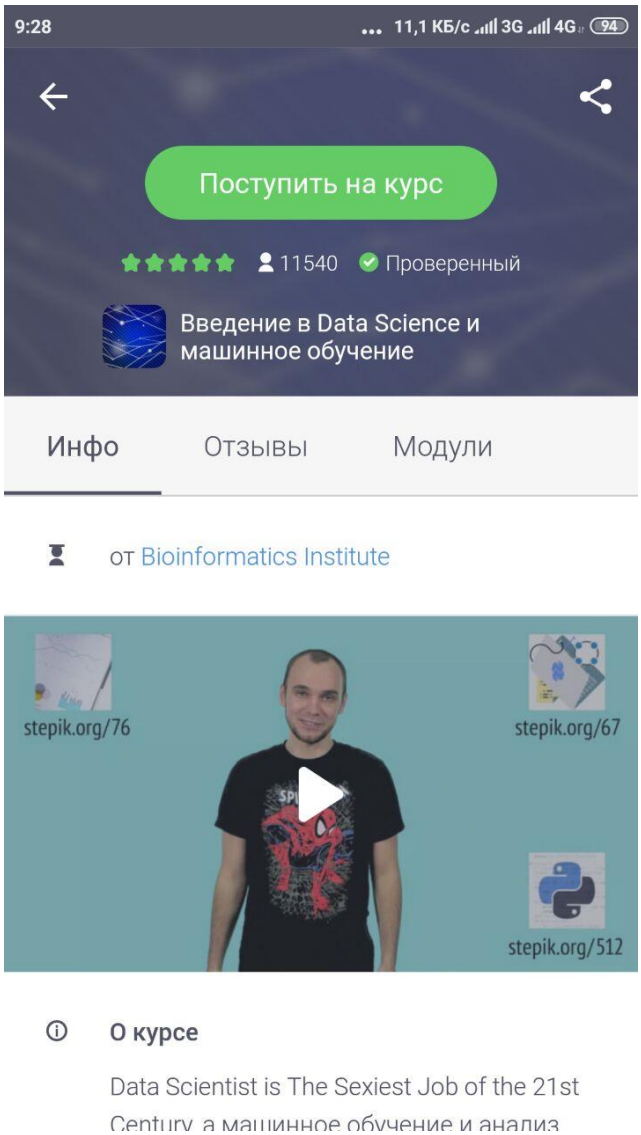


Рисунок 2.3 – Загальний вигляд обраного курсу Stepic

### 2.2.4. Система дистанційного навчання Coursera

Основним ресурсом для дистанційної освіти є саме Coursera. Це можна пояснити тим, що велика кількість найвідоміших університетів з усього світу (більше 115) викладають власні лекції у вільному доступі на даному ресурсі в мережі Інтернет. На цьому ресурсі можна обрати різноманітні курси з таких областей знань, як біологія, суспільствознавство, розробка, математика та багато чого іншого. В результаті проходження та курсу та у разі успішної здачі іспитів та тестів користувач має змогу отримати електронний сертифікат про успішне завершення курсу, який можна додати до свого резюме або профілю на сайті Linkedln. Незважаючи на те, що основна частина курсів викладаються англійською

мовою, ресурс має велику кількість користувачів з різних куточків світу. Тому ресурс почав додавати до курсів субтитри, щоб спростити використання ресурсу для цих людей.

Всі начальні курси, які розміщені на Coursera є безкоштовними. Проте, ті користувачі, які хочуть отримати сертифікат з персональною верифікацією, можуть замовити платний сертифікат. Користувач також має можливість отримати сертифікат з персональною верифікацією безкоштовно, але для цього йому необхідно довести що він не має змоги придбати цей сертифікат та що користувач завершить курс в повному обсязі .

Час, виділений на проходження курсу коливається в залежності від кокретного курсу. На проходження найкоротших курсів користувачеві необхідно бути витратити 3 тижні. Проте найбільш популярними є курси, котрі тривають 6 тижнів. Також приблизно четверта частина всіх курсів тривають понад 10 тижнів.

В процесі проходження курсу студенту кожного тижня надсилаються відео лекції тривалістю приблизно 15 хвилин або більше, які йому необхідно переглянути. Також студент має виконувати завдання, які йому надсилаються, а також читати статті, рекомендовані ресурсом. Завдання надсилаються студенту в різному вигляді, це можуть бути як стандартні тести чи проекти, так і написання есе чи творчі завдання. Перевірки пройдених користувачем тестів проводиться автоматично, а для перевірки творчих завдань чи проектів Coursera впровадила власну систему перевірки. Суть цієї системи полягає в тому, що користувачів оцінюють інші користувачі цього ресурсу. Спочатку студент виконує завдання та надсилає його, потім йому приходять декілька анонімних робіт інших студентів. Система встановлює певні критерії та чіткий алгоритм оцінки роботи іншого студента. Також студент може додоати власні коментарі для того, чию роботу він оцінює. Після завершення цього процесу, студент може бачити свою оцінку , яка є середнім арифметичним оцінок інших користувачів.

.

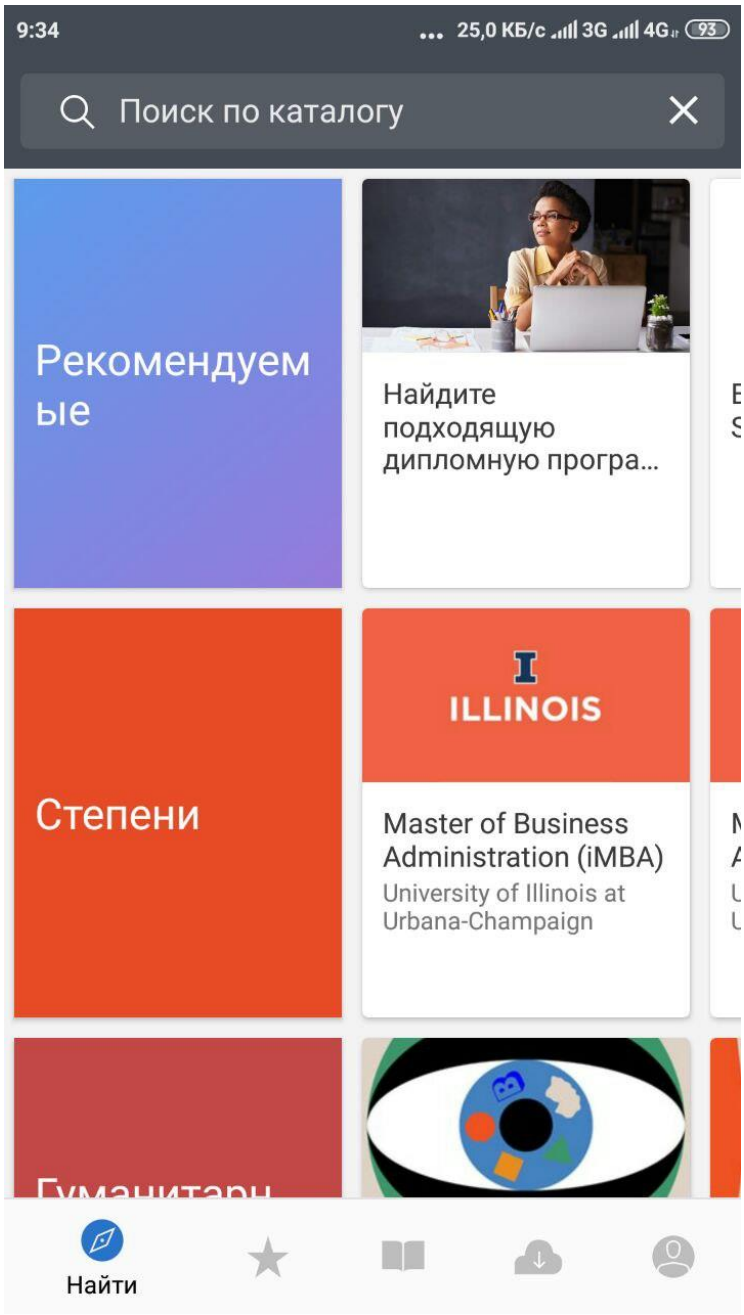


Рисунок 2.3 – Інтерфейс мобільного додатку Coursera

Додаток має наступні функції:

- проходження курсів онлайн за допомогою бездротової мережі або мобільного зв’язку;
- завантаження відеолекцій або аудіозаписів на пристрій для відтворення в режимі офлайн;
- обговорення завдань та спілкування з іншими студентами беручи участь в розділі форуму;

- можливість отримання сертифікатів після успішного проходження курсу не менше ніж на мінімальний бал;
- дослідження програм навчання з провідних університетів всього світу;
- огляд курсів в різних предметниз областях,від математики та музики до медицини;
- можливість проходження курсів власною мовою, на Coursera впроваджуються курси різними мовами;

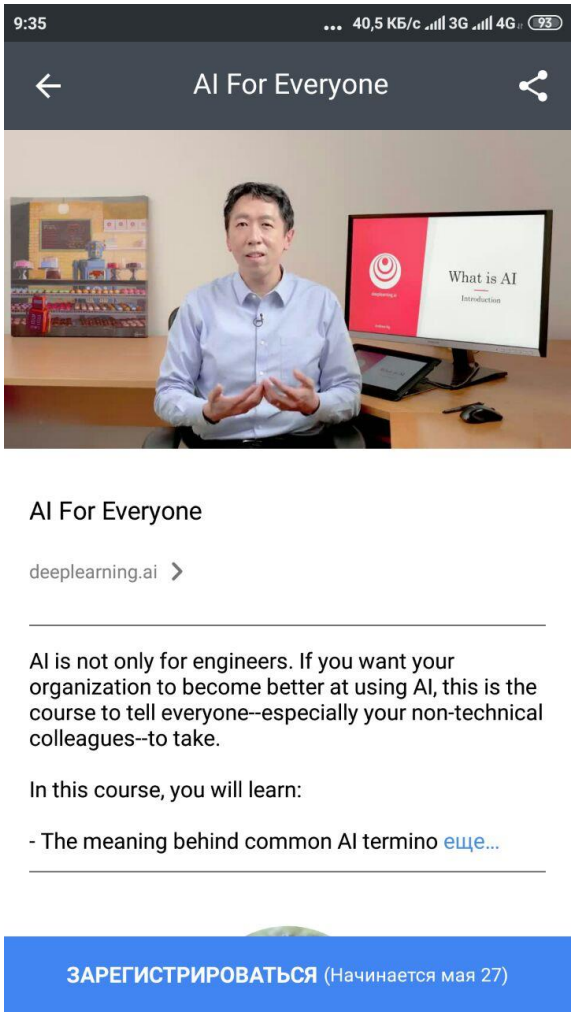


Рисунок 2.3 – Загальний вигляд обраного курсу Coursera

## 2.3. Висновки до розділу

В даному розділі було розглянуто основні доступні на ринку системи дистанційного навчання. Оскільки можливостей розробки додатків для мобільних

пристроїв стає дедалі більше, створення нової системи дистанційного навчання є актуальною задачею на сьогоднішній день. Ця система має бути зручна, проста та зрозуміла для всіх користувачів.

Найбільш вірогідним рішенням є проектування та розробка нової системи дистанційного навчання для використання на мобільних пристроях, яка буде відповідати всім вимогам і задовольняти потреби користувача в якісному отриманні знань.

# 3. МЕТОДИ РЕАЛІЗАЦІЇ ПРОГРАМНОЇ СИСТЕМИ

В процесі розробці даного програмного продукту була використана модель бізнес логіки програмного забезпечення. Такий підхід надає суттєву можливість розширювати функціональну оболонку системи в процесі подальшого життєвого циклу програмного продукту без суттєвого перепроектування, можливість паралельної роботи різних груп розробників.

В процесі аналізу вимог було прийнято рішення реалізовувати систему дистанційного навчання на мобільній платформі. Головною перевагою смартфону є саме мобільність, що дозволяє отримати доступ до програмного продукту в будь-якому місці і в будь-який час. Також перевагою мобільної платформи є портативність пристроїв. Це дозволяє користувачу завжди бути біля пристрою і постійно вивчати новий матеріал.

Програмний продукт має бути побудований на мобільній платформі “Android” для використання на мобільних пристроях, що працюють на цій системі. В якості мови програмування було обрано об’єктно орієнтовану мову програмування “Java”. Для створення додатку було обрано інтегроване середовище розробки “Android Studio”. Побудова проекту відбувається за допомогою системи автоматичної збірки “Gradle”.

## 3.1. Оцінка вимог

Для вирішення задачі необхідні наступне інструменти:

- персональний комп’ютер;
- середовище розробки Android Studio;
- система автоматизованої збірки Gradle;
- мобільний пристрій на платформі Android.

## 3.2. Платформа Android

Платформа “Android” є продуктом групи “Open Handset Alliance” , яка ставить перед собою ціль створити найбільш досконалу мобільну систему. З точки зору розробки програмного забезпечення “Android” знаходиться в самому центрі розробки відкритого програмного забезпечення. Перши й пристрій під управлінням цією системою був випущений у 2008 році і єдиним інструментом розробки програм для нього були послідовні випуски пакета розробки “SDK”.

За шириною можливостей платформа “Android” не поступається місцем операційним системах персональних комп’ютерів. Це багаторівнева система на основі ядра “Linux” з широкими функціональними можливостями. В підсистему користувацького інтерфейсу входять вікна, представлення та віджети. “Android” володіє широким спектром можливостей підключення, таких як “Wi-Fi”, “Bluetooth” та протоколи передачі даних через стільникову мережу. В стек програмного забезпечення “Android” входить і підтримка сервісів, заснованих на визначенні місцеположення та акселерометрів , хоча не всі пристрої на цій платформі устатковані необхідним обладнанням. Також наявна підтримка відеокамери. Історично двома областями, це мобільні системи відставали від настільних були графіка та способи збереження даних. “Android” вирішує проблему графіки завдяки вбудованій підтримці графіки, включаючи бібліотеку “OpenGl”. Задача збереження даних спрощується завдяки наявності в платформі “Android” популярної бази даних з відкритим кодом “Sqlite”.

Операційна система “Android” працює поверх ядра “Linux”. Додатки пишуться на мові програмування “Java” і виконуються в віртуальній машині. Важливо уточнити, що віртуальна машина це не “JVM”, як можна було б очікувати, а відкрита технологія “Dalvik Virtual Machine”. Кожний додаток “Android” запускається всередині екземпляра “Dalvik VM” , який в свою чергу укладений в рамках контрольованого ядром “Linux” процесу.

Компанія Google вклала багато ресурсів в скорочення платформи та зростання її ефективності. Оптимізація в першу чергу направлена на зменшення розміру,

збільшення швидкості, економію заряду акамулятора та зниження витрат пам’яті. Вони провели роботу на декількох рівнях стеку. Віртуальна машина Dalvik була ретельно розроблена з метою задоволення цих вимог до продуктивності та має декілька незвичайних характеристик. На відміну від звичайного виконуваного середовища Java, Dalvik заснований на регістрі, а не на стеку і не підтримує JIT компіляцію. Кожен окремий додаток виконується в окремому екземплярі віртуальної машини Dalvik і пам’ять розподіляється між цими екземплярами для зменшення витрат. Для того щоб скоротити час запуску додатка, Dalvik має компонент під назвою Zygote, який попередньо ініціалізує екзмепляри віртуальних машин і розгалуджує їх, якщо в цьому виникає необхідність.

Радикально інший підхід Android до розробки мобільних додатків пропонує деякі унікальні переваги, але він також створює багато проблем для сторонніх розробників. Найбільшою перевагою в такому підході є те, що він забезпечує високий рівень однорідності. Переважна більшість додатків Android зможуть без проблем працювати практично на будь-якому пристрої на базі Android, не потребуючи при цьому подальших змін.

До складу “Android” входить комплект базових додатків: клієнти електронної пошти, календар, різноманітні карти, браузер, програма для керування контактами тощо.

“Android” дозволяє використовувати всю потужність “Android” , що використовуються в додатках ядра. Архітектура побудована таким чином, що будь який додаток може використовувати уже реалізовані можливості іншого додатка, при умові, що останнє відкриє доступ до використання своєї функціональності. Таким чином архітектура реалізує принцип багаторазового використання компонентів і додатків. Нижче, на рисунку 3.1. наведено архітектуру системи.

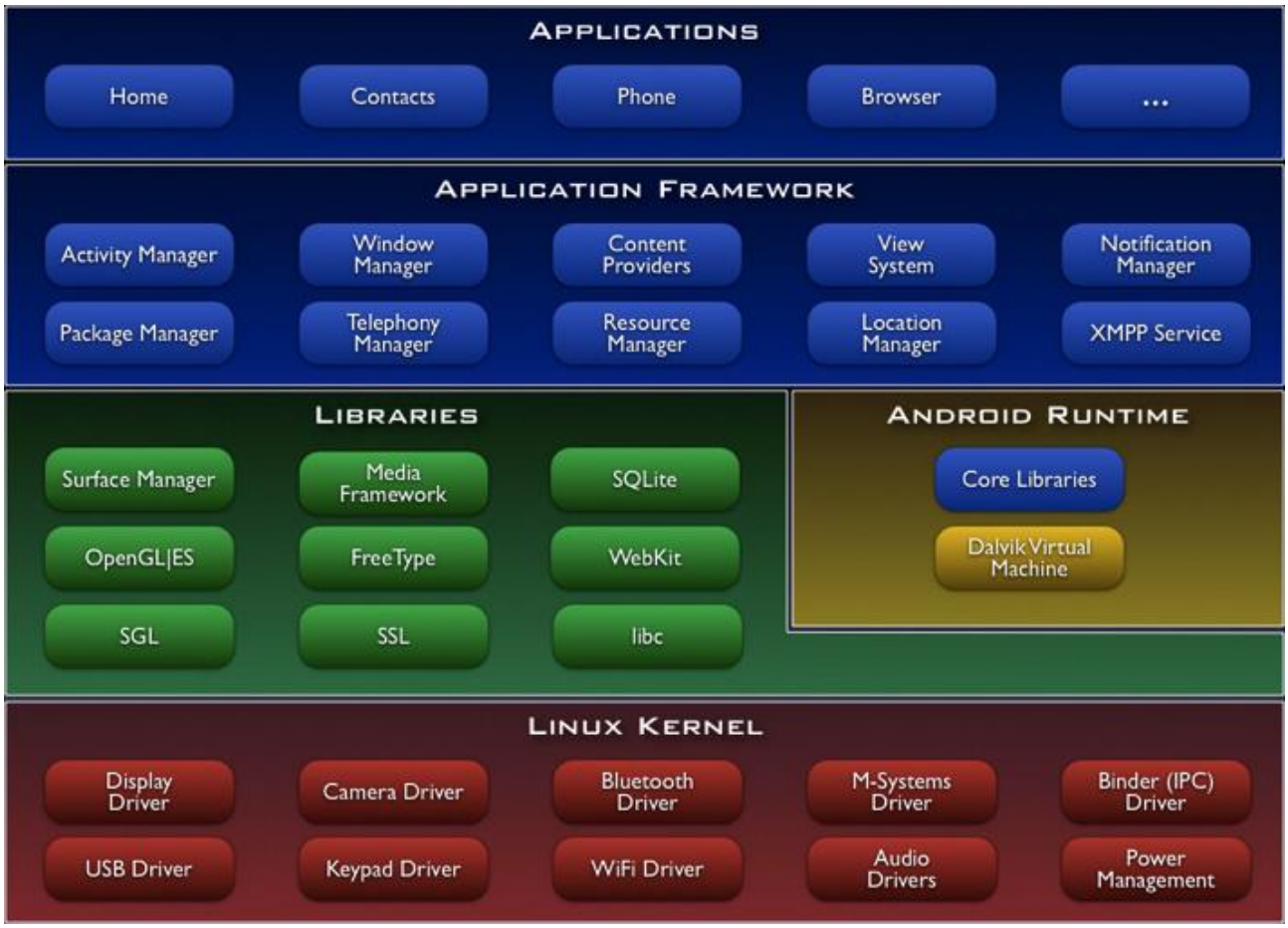


Рисунок 3.1 — Схема архітектури системи

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

Сторонні додатки можуть реплікуватися або взаємодіяти практично з усіма основними компонентами платформи. Наприклад, Android надає методи для отримання інформації з списку контактів користувача та для розширення системи контактів новими полями даних. Також легко зробити новий інтрефейс дзвінка або реалізувати користувацьку поведінку для системних подій, таких як вхідні повідомлення. Зокрема, API дійсно дозволяють створювати додатки, які повністю інтегруються з рештою платформи. На рисунку 3.2 зображено приклад використання API.

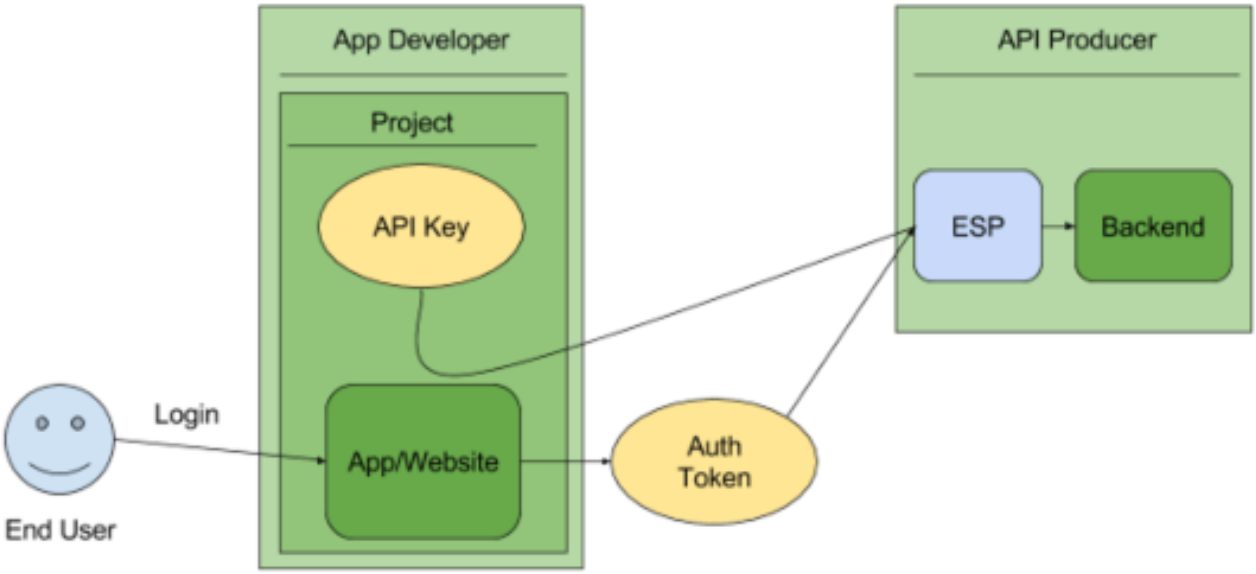


Рисунок 3.2 — Приклад використання API

Набір інструментів для віджетів Android надає велику кількість дуже корисних компонентів відразу. Окремі віджети розроблені спеціально для зручної взаємодії з пальцями, з вже вбудованими функціями, такими як кінетична прокрутка. Розробники використовують мову опису користувацького інтерфейсу на основі XML для визначення макету та атрибутів віджетів, при цьому описи XML завантажуються в програму через систему ресурсів Android.

На окремі віджети, описані в в макеті XML, можна посилатися по ідентифікатору в програмі. Також є можливість створення віджетів програмно і маніпулювати ними користувацьким інтерфейсом в процесі виконання програми. Найбільш правильним способом створення макетів є написання описів XML вручну. Пакет Android SDK надає вбудований інструмент візуального макета, проте він не підтримує всі віджети і не завжди працює здагоджено.

Додатки Android складаються з провайдерів , служб, приймачів та активностей. Всі ці компоненти мають окрему задачу в стеку додатків Android. Спосіб, яким вони можуть взаємодіями один з одним це те, що ноповнює Android його модульністю.

Провайдери контенту слугують рівнем абстракції для взаємодії з

різноманітними джерелами даних і для обміну постійними даними між додатками. Вони надають потрібну інформацію через стандартизований інтерфейс запитів. Запити описуються за допомогою синтаксису URI, але розробники додатків зазвичай використовують багаторівневі класи оболонки , які генерують рядки запитів і керують ними. Коли програма надсилає запит провайдеру контента, він повертає відповідь у вигляді об'єкту-курсору, який надає програмний базисний доступ до базового контенту. Також можна підключити механізми моніторингу до провайдерів контенту, для того щоб ваш додаток міг отримувати повідомлення про зміну даних.

Система провайдера контенту пропонує розробникам Android додатків декілька суттєвих переваг. Найбільш явною перевагою є те, що система забезпечує високий рівень взаємодії, дозволяючи додаткам обмінюватися даними досить уніфікованим методом. Декілька ключових джерел даних платформи, включаючи список контактів та системи збереження мультимедіа, за замовчуванням доступні через інтерфейси провайдерів контенту, тому їх легко використовувати з сторонніх додатків для реалізації певних функцій.

Важливою особливістю, яка відрізняє Android від інших платформ є те, що Android офіційно підтримує фонові процеси в сторонніх додатках. Вони реалізовані за дпомогою сервісного компоненту Android. Сервіси – це операції без заголовку, які виконуються в фоновому режимі протягом тривалого часу.

Система повідомлень Android приблизно схожа з аналогічною системою сигналів в Linux. Розробники вбудовують широкомовні приймачі, які можуть визначати, коли з'являються визначені системні повідомлення чи події, і винонувати певні дії у відповідь. Багато базових системних подій можна відслідковувати за допомогою системи мовлення, тому її можна використовувати для відстежування таких речей, як зміна стану акумуляторної батареї, отримання вхідних SMS повідомлень, натискання апаратних кнопок, зміна з'єднання, встановлення сховища та затемнення екрану.

Система Android Intents є ключем до розуміння того, як всі ці частини використовуються разом. Об'єкти Intents, які містять ідентифікатор дії та URI даних,

використовуються для виклику дій, служб та отримувачів, Ідентифікатор дії вказує бажану поведінку, а необов'язковий URI даних надає місце розташування цих даних, над якими має виконуватися дія. Нарисунку 3.3 зображено загальну смеху компонентів Android додатку.

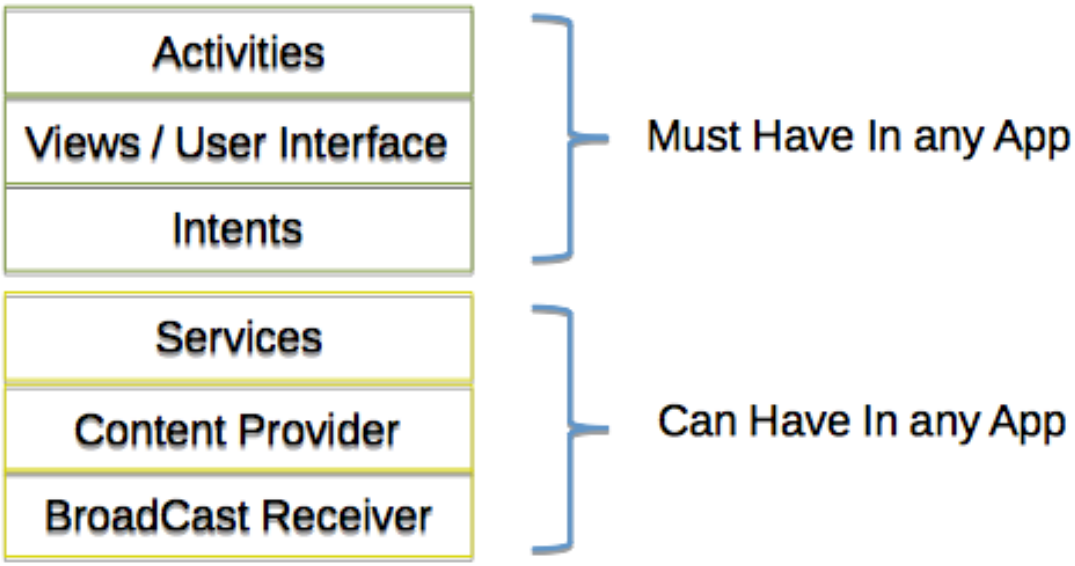


Рисунок 3.2 — Приклад використання API

Одна із головних переваг системи Android – її відкритість. Операційна система Android побудована на основі відкритого висхідного коду і розповсюджується на вільній основі. Це дозволяє розробникам отримати доступ до висхідного коду і зрозуміти яким чином реалізовані властивості і функції додатків. Кожен користувач може прийняти участь в удосконаленні операційної системи.

## 3.3. Мова програмування Java

Для розробки проекту було використано мову програмування Java. На сьогоднішній день Java є однією з найбільш розповсюджених і найпопулярніших

мов програмування. Розробка мови Java почалася в 1991 році Джеймсом Гослінгом та його колегами, які працювали в компанії Sun Microsystems. На розробку першої робочої версії вони витратили 18 місяців. Спочатку мова мала назву Oak, але пізніше була переіменована в Java. Першочерговою причинию, яка сприяла створенню цієї мови, слугував не Інтернет, а потреба в незалежній від кокретної платформи мові, яку можна було б використовувати для розробки програмного забезпечення для різних побутових приладів, таких яких мікрохвильова і піч і різноманітні дистанційні пристрої. В контроллерах таких пристроїв застововуються різні процесори з різною архітектурою. Недоліком існуючих мов С та С++ було те, що розроблені програми на цих мовах мали компілюватися для певної платформи. Розробка компілятора під кокретний процесор була дуже витратною, як з фінансової точки зору, так і з точки зору часу.

До створення мови Java призвела потреба в незалежній від кокретних процесорів мові, яку можна було в використосувати для створення програмного коду який можна було б виконувати на різних платформат та в різних середовищах. Приблизной в той самий час, коли формувалися основні особоливості мови Java, з’явилась і всесвітня мережа Інтернет. Вона значним чином вплинула на майбутнє мови програмування Java, яка могла так і залишитись досить корисною, проте непримітною мовою для написання програмного коду для побутових пристроїв. Всесвітня мережа посприяла значному зростанню популярності мови Java серед розробників, а Java дуже сильно вплинула на Інтернет. Мова Java не тільки значним чином спростила процес створення програм для Інтернету вцілому, а й зумовила появу прикладних програм нового типу, призначних для роботи в мережі які змінили поняття змісту середовища мережі. Більше того, мова Java дала можливість вирішити дві найбільш гострі проблеми програмування того часу пов’язані з Інтернетом – це безпека та кросплатформність.

Основна особливість Java, яка вирішує обидні проблеми, полягає в тому, що компілятор Java видає не виконуваний код, а байт-код, який представляє собою дуже оптимізований набір інструкцій, призначених для виконання в виконуючій системі Java, яка називається віртуальною машиною Java (JVM – java virtual machine). Власне

кажучи, початкова версія віртуальної машини JVM розроблялася в якості інтерпретатора байт-коду. Це може викликати недорозуміння, адже для забезпечення максимальної продуктивності компілятори багатьох сучасних мов програмування призвані створювати виконуваний код. Але те, що програма на Java інтерпретується віртуальною машиною JVM, як раз і допомогає вирішити основні проблеми розробки програмних продуктів для Інтернету.

Трасляція програми Java в байт-код значно спрощує її виконання в середовищах різного типу, оскільки на кожній платформі необхідно реалізувати тільки віртуальну машину JVM. Якщо в окремій системі міститься виконуючий пакет, в ній можна виконувати будь-яку програму на мові програмування Java. Однак, слід мати на увазі що всі вітуальні машини JVM на різних платформах та процесорах здатні правильно інтерпретувати один і той самий байт-код, незважаючи на деякі відмінності і особливості реалізації. Якщо б програма на мові Java компілювалася в машинозалежний код, то для кожного типу процесорів, підключеного до Інтернету, повинні були б існувати окремі версії однієї і тієї ж самої програми. Очевидно, що таке рішення неприйнятне. Таким чином, організація виконання байт коду віртуальною машиною JVM є найпростішим способом створення по справжньому кросплатформних програмних продуктів.

Той факт, що програма на мові програмування Java виконується віртуальною машиною JVM, сприяє також підвищенню її безпеки. Віртуальна машина JVM управляє виконанням програми, тому вона має можливість ізолювати програму та заважати появі побічних ефектів від її виконання за межами даної системи. Цілий ряд обмежень, які існують в Java, також сприяють підвищенню безпеки системи.

Загалом, коли програма компілюється в проміжну форму, а після цього інтепретується віртуальною машиною JVM, вона виконується менш швидше, ніж якщо б вона була скомпільована в виконуваний код. Проте в Java цю різницю в продуктивності не дуже помітно. Байт-код значним чином оптимізований, а тому його застосування дозволяє віртуальній машині JVM виконувати програми значно швидше, ніж того можна було б очікувати, враховуючи наведену вище особливість виконання програм на мові програмування Java.

Мова Java була задумана як інтерпретована, але ніщо не заважає їй оперативно виконувати компіляцію байт-кода в машинозалежний код для підвищення продуктивності. Тому через деякий час після появи Java з'явилася технологія HotSpot, яка надає динамічний компілятор ( або так званий JIT-компілятор) байт-коду. Якщо динамічний компілятор входить до складу віртуальної машини JVM, то обрані фрагменти байт-коду компілюються в виконуватий код по частинах, в реальному часі і за вимогою. Важливо також розуміти, що одночасна компіляція всієї програми Java у виконовуний код недоцільна, оскільки Java проводить різноманітні перевірки, які можуть бути проведені тільки в процесі виконання. Замість цього динамічний компілятор компілює код під час виконання у разі виникнення необхідності. Більше того, компілюються  не всі фрагменти байт-коду, а тільки ті, яким компіляція надасть перевагу , а решта програмного коду інтерпретується. Проте принцип динамічної компіляції забезпечує значне підвищення продуктивності. Навіть при динамічній компіляції байт-коду характеристики кросплатформності та безпеки  зберігаються, оскільки віртуальна машина JVM надалі відповідає за цілісність виконуваного середовища.

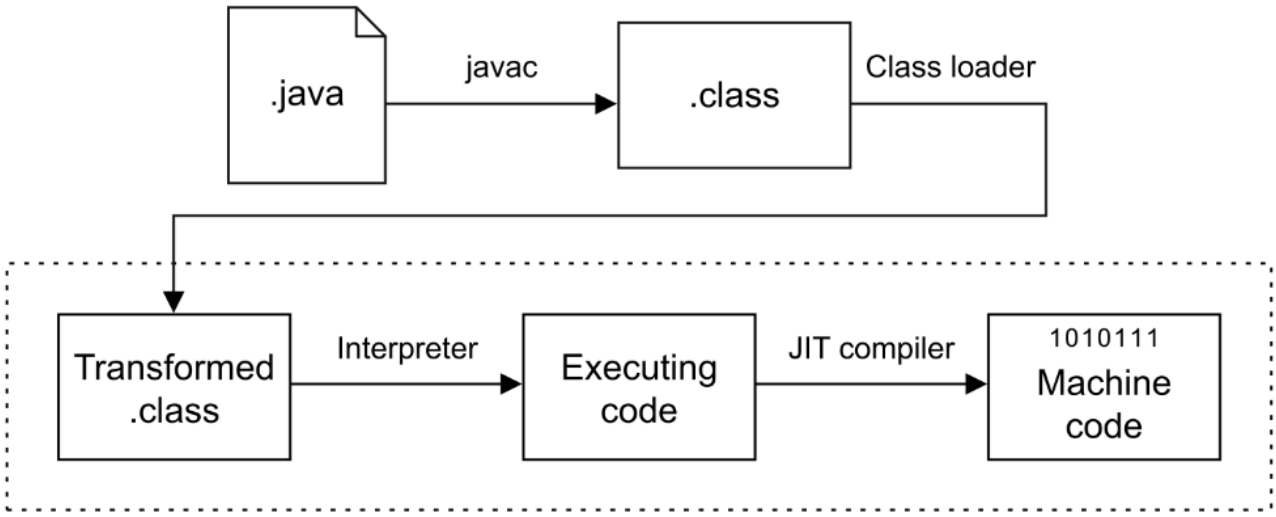


Рисунок 3.3 — Компіляція програмного коду в Java

Основним фактором, який зумовив створення мови програмування Java, стала необхідність в забезпеченні кросплатфомності та безпеки програм. Проте в свою чергу роль в формуванні остаточної версії мови зіграли й інші фактори, такі як: простота, безпека, кросплатформність, об'єктна орієнтованість, надійність, багатопотічність, архітектурна нейтральність, інтерпретація, висока продуктивність, розподільність, динамічність.

Мова програмування Java задумана як проста у вивченні та ефективна у використанні професіональними розробниками. Опанувати мову Java тим, хто вже має певний досвід у розробці програм, буде не складно. Якщо ж ви вже ознайомлені і з принципами об'єктно-орієнтованого програмування то буде ще простіше. А від тих, хто має досвід розробки програмного забезпечення на мові C++ перехід на мову требує мінімум зусиль. Мова Java наслідує синтаксис C++ та більшість об'єктно-орієнтованих властовостей C++, тому для більшості розробників програмного забезпечення вивчення Java не складе великих труднощів.

Хоча попередники мови Java і вплинули на його архітектуру та синтаксис, при її проектуванні не ставилась задача сумісності по висхідному коду з якою небудь іншою мовою. Це надало можливість команді розробників створювати Java з чистого аркушу. Одним із наслідків цього є чіткий, прктичний, пргаматичний підхід до об'єктів. Окрім того, що мова Java запозичила властивості багатьох об'єктно-орієнтованих середовищ, розроблених протягом останніх десятиріч, в ній вдалося досягнути золотої середини між строгим дотриманням принципу, що всі елементи програми є об'єктами, та більш прагматичним принципом. Об'єктна модель в Java проста та легко легко розширюється. В той же час такі елементарні типи даних як цілочисельні, зберігаються у вигляді високопродуктивних компонентів, які не є об'єктами.

Багатоплатформне середовище веб пред'являє до програм високі вимоги, оскільки вони повинні надійно виконуватися в системах різного типу. Тому можливість створення надійних програм була одним із головних пріоритетів при розробці мови Java. Задля забезпечення надійності в Java накладається ряд обмежень в декількох найбільш важливих областях, що дозволяє розробникам програмного

забезпечення виявляти помилки на ранніх етапах розробки програми. Втой же час Java звільняє від необхідності турбуватися з приводу багатьох помилок програмування, які зустрічаються найчастіше. А оскільки Java суворо типізована мова програмування, то перевірка коду виконується під час компіляції. Але код перевіряється і під час виконання. В результаті помилки, які дуже важко виявити та які часто призводять до появи складно відтворюваних ситуацій під час виконання, неможливі в програмі на Java. Передбачуваність програмного коду в різних ситуаціях є однією с основних особливостей Java.

Мова Java була розроблена у відповідь на потребу створювати інтерактивні мережеві програми. З цією метою в Java підпримується написання багатопотічних програм, здатних одночасно виконувати декілька дій. Виконуюча система Java містить в собі складне рішення задачі синхронізації багатьох процесів, які надають можливість будувати стійко працюючі інтерактивні системи. Простий підхід до організації багатопотічної системи, якмй реалізований в Java, дозволяє розробникам програмного забезпечення зосереджувати основну увагу на кокретній поведінці програми, а нена створенні багатопотічної підсистеми.

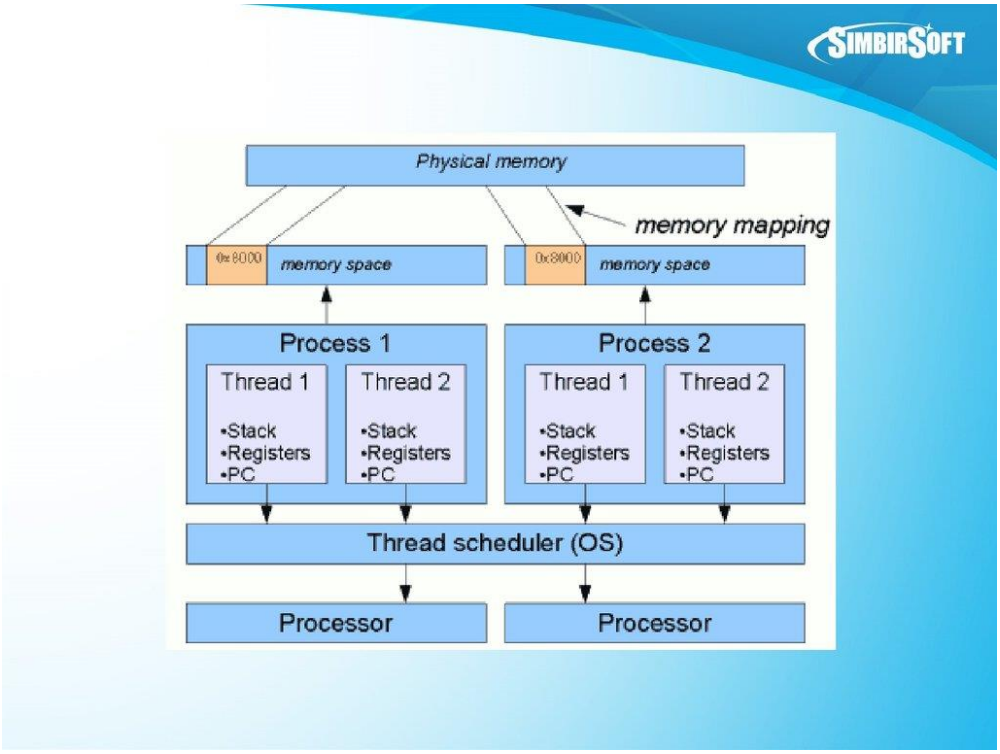


Рисунок 3.4 — Приклад багатопотічності в Java

Основна задача, яку ставили перед собою розробники Java, булло забезпечення довговічності та кросплатформності коду. Головною проблемою, яка стояла перед розробниками в процесі створення Java, була відсутність будь яких гарантій, що код який написаний сьогодні, буде успішно виконуватися завтра навіть на одному і тому самому ком'ютері. Процесори та операційні системи постійно вдосконалюються і будь-які зміни в основних системних ресурсах можуть стати причиною непрцездатності програмного продукту. Намагаючись яким небудь чином змінити цю ситуацію, розробними прийняли ряд жорстких рішень в самій мові Java і в віртуальній машині JVM. Вони поставили перед собою мету, щоб те що було написано одного разу виконувалося всюди, в будь-який час і завжди. І в значній мірі цієї мети було досягнуто.

Як згадумалося раніше, виконуючи компіляцію програм в проміжне представлення, яке називається байт-кодом, Java дозволяє створювати міжплатформні програми. Такий код може виконуватися в будь-якій системі, на якій реалізована віртуальна машина JVM. З перших ж спроб розробити міжплатформні рішення вдалося досягнути поставленої мети, хоч і за рахунок зниження продуктивності системи. Байт-код Java був ретельно розроблений таким чином, щоб його можливо було б з висококою часткою ефективності перетворювати безпосередньо в машинозалежний код на конкретній платформі за допомогою динамічного компілятора. Виконуючі системи Java, які забезпечують таку можливість, зберігають всі переваги коду, який не залежить від конкретної платформи.

Мова Java призначена для розподіленого чередовища Інтернету, оскільки мова підтримує сімейство мережевих протоколів TCP/IP. По суті, звернення до ресурсу по уніфікованому вказівнику ійнформаційного ресурсу (URL) мало чим відрізняється від звернення до файлу. В мові Java також підтримується віддалений виклик методів програми (RMI – remore method invocation). Така властивість дозволяє викликати методи програм через мережу.

Програми на мові Java містять значний об'єм даних динамічного типу, що використовуються для перевірки повноважень і дозволу доступу до об'єктів під час

виконання програми. Це в свою чергу дозволяє безпечно та раціонально виконувати динамічну зв’язку коду. Ця обставина виключно важлива для стійкості середовища Java, де невеликі фрагменти байт-коду можуть динамічно оновлюватися в дійючій системі.

## 3.4. Середовище розробки Android Studio

В наш час Створення програмного забезпечення в багатьох здійснюється з використанням інтегрованого середовища розробки (IDE). В IDE автоматизований процес компіляції, збірки та запуску додатків, що значно спрощує роботу розробнику програмного забезпечення і довзволяє розробнику початківцю без значних сузиль створювати свої перші додатки.

Інтегроване середовище розробки зазвичай має декілька основних влаастивостей які спрощують розробку програмного забезпечення. Підсвічення синтаксису та нумерація рядків значним чином допомогають зоєрінтуватися в програмного коді. Майже кожне IDE має автоматичний відладник додатків. Також дуже важливою є інтеграція з системою контролю версій. Ця функція використовується, коли над проектом працює не один розробник а ціла команда розробників. Ці системи дозволяють на відстані розробникам один і той самий код разом і не переписувати правки один одного. На сьогоднішній день кожна мова програмування має інтегроване середовище розробки.

В якості середовища розробки було обрано “Android Studio”. Це інтегроване середовище розробки виробництва компанії “Java”, за допомогою якого розробникам стають доступні інструменти для створення додатків. Це середовище можна встановити на різні операційні системи. “Android Studio” можна завантажити і користуватись безкоштовно. В ньому присутні макети для створення користувацького інтерфейсу, з чого зазвичай і починається робота над додатком. Середовище містить інструменти для створення додатків на смартфони, планшети, годинники та автомобілі.

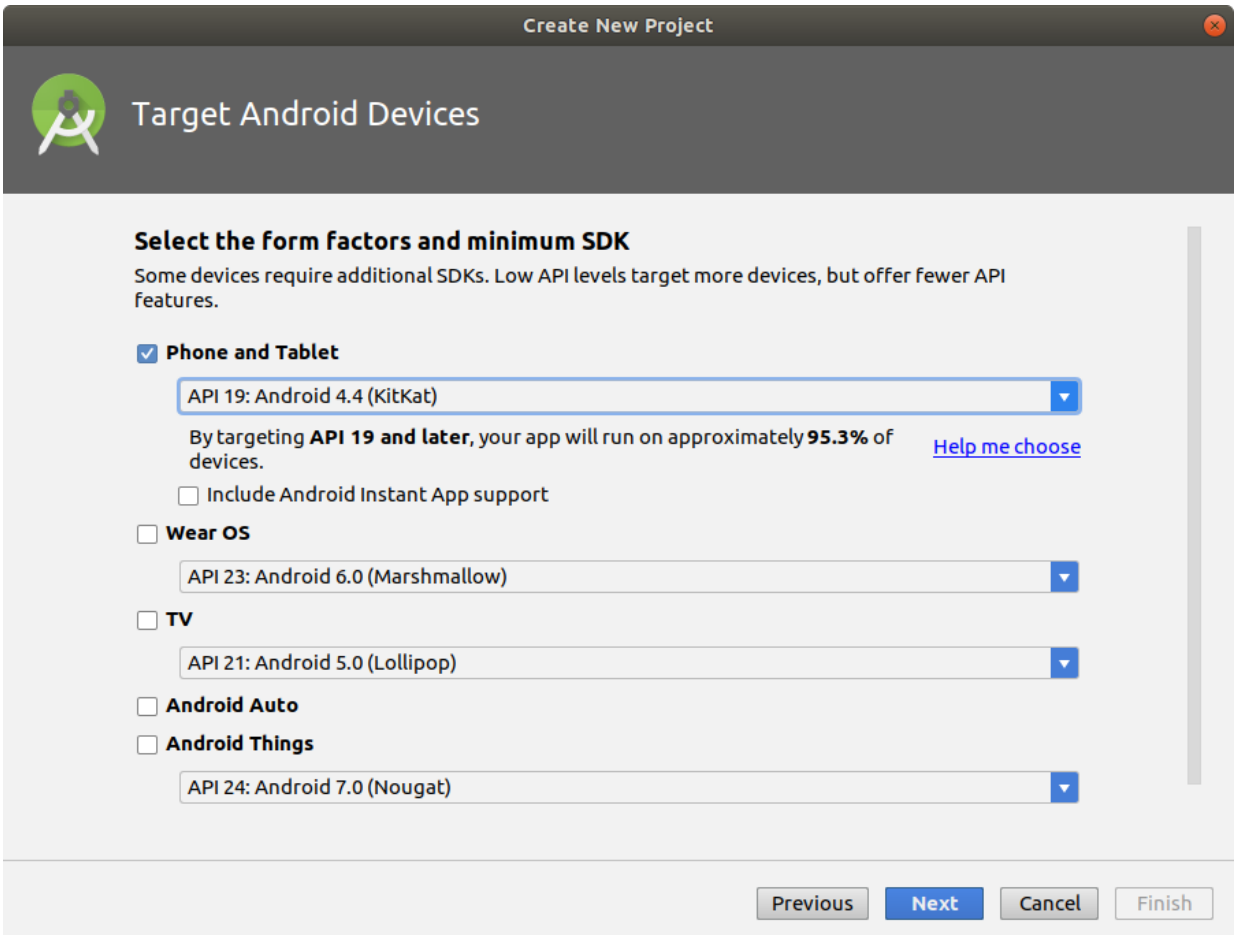


Рисунок 3.3 — Екран вибору форм фактору в Android Studio

Середовище Android Studio – це офіціальна інтегроване середовище розробки для проектування та розробки додатків Android. Воно побудоване на основі Intellij IDEA, інтегрованого середовища розробки для Java, і включає в себе інструменти редагування коду та середовище розробки додатків. Вперше Android studio було анонсовано на Google I/O в травні 2013 року, а перша стабільна збірка була випущена в грудні 2014 року. Середовище Android Studio доступне для таких настільних платформ як Windows, Mac та Linux. Воно замінило Eclipse в якості основного інструменту для розробки додатків на Android.

Для того щоб почати роботу в Android Studio спочатку необхідно встановити безкоштовний пакет для розробника JDK (Java Development Kit). Після встановлення та налаштування цього пакету необхідно завантажити Android Studio з офіційного сайту. Далі необхідно знайти завантажений виконуваний файл інсталяції Android Studio та запустити процес встановлення. Коли з’явиться майстер

налаштування Android Studio необхідно налаштувати установку відповідно до вимог користувача з точки зору розташування файлової системи, в яку повинне бути встановлене середовище Android Studio та доступу іншими користувачами системи. Після встановлення інтегрованого ередовища розробки можна починати створювати додатки.

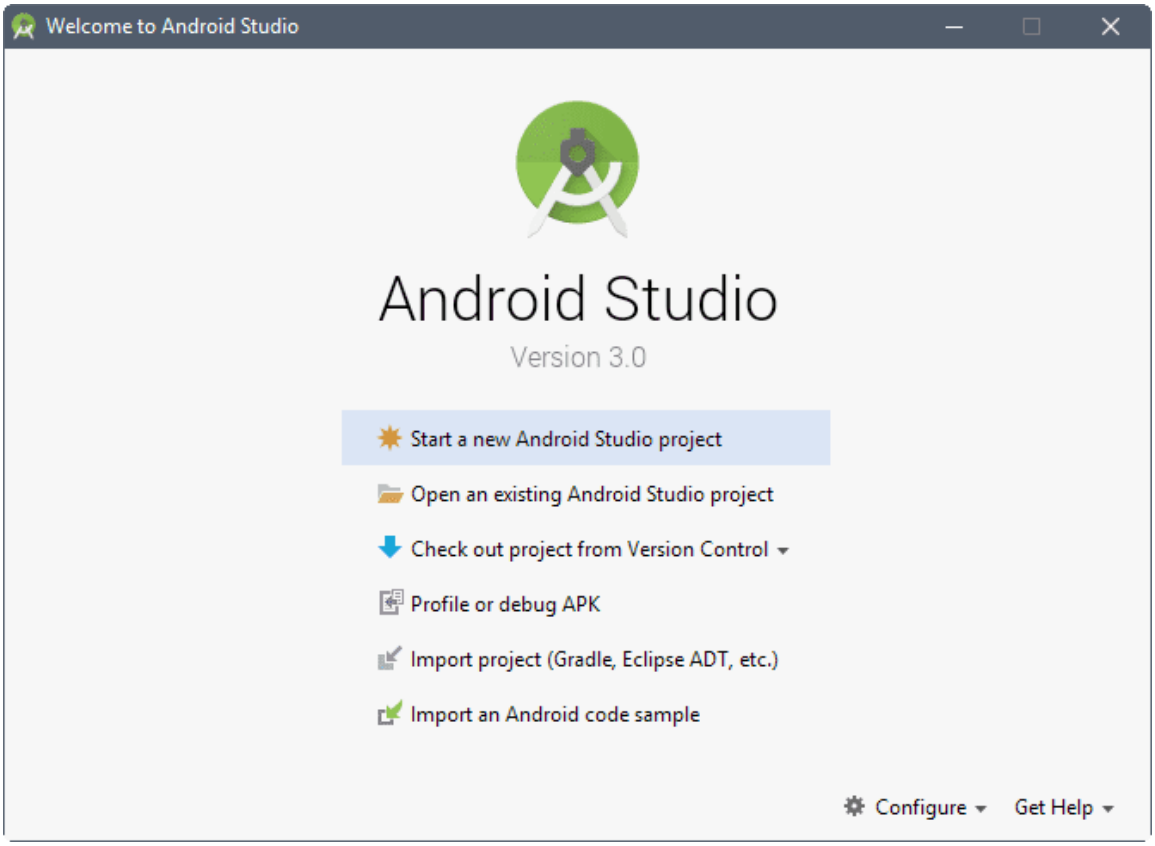


Рисунок 3.4 — Стартовий екран проекту в Android Studio

Інтегроване середовище розробки Android Studio є віконним. Для того щоб максимально використати простір екрану і щоб не перевантажувати розробника Android Studio відображає лише невелику частину доступних вікон в будь-який час. Деякі вікна є контекстними і з’являються тільки в разі контекстного виклику з певних інструментів. Також Android Studio містить приховані вікни, які розрбник може активувати або деактивувати з меню налаштувань. Однією ж найважливіших функцій будь якого інтегрована середовища розробки є навігація. Проекти Android зазвичай складаються з багатьох тек, каталогів та файлів, що організовані між собою. Програмні додатки з багатька висхідними файлами коду та ресурсами організовані в рамках

структури проекту. Це зроблено для кращої класифікації файлів і визначення компіляції вихідного коду та генерації байт-коду. Вцілому ця файлова структура і називається проектом, який є організаційною одиницею, що являє собою повне програмне рішення. Проекти додатків зазвичай складаються з файлів вихідного коду Java або Kotlin, файлів конфігурації XML, зображення, стилі та інші ресурси в організаційній структурі. При створенні програми Android Studio допомогає у структуризації проекту. На етапі збірки проекту створюються файли конфігурації та відбувається структуризація каталогів за ієрархією.

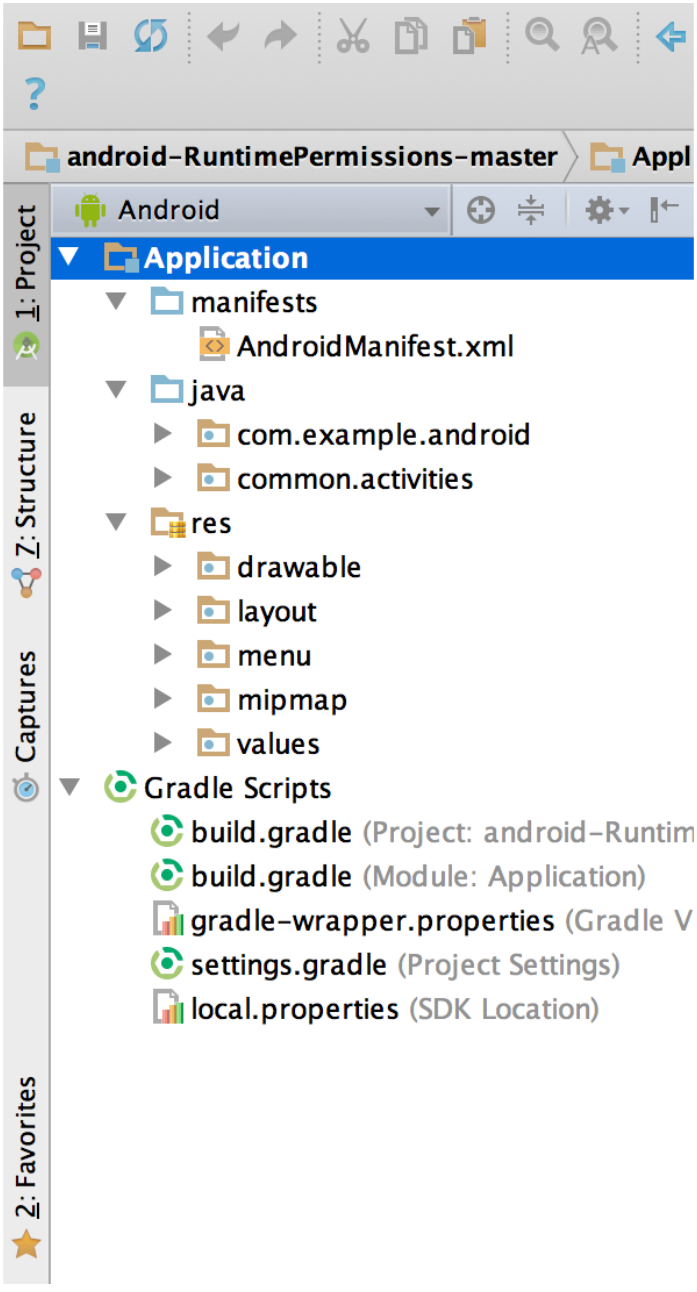


Рисунок 3.5 — Структура проекту в Android Studio

В процесі розробки додатків в інтегрованому середовищі розробки Android Studio необхідно буде компілювати та запускати додаток декілька разів. Розроблену програму Android можна перевірити, встановити та запускати на фізичному пристрої або на віртуальному пристрої Android (AVD – android virtual device), який називають емулятором. Перед тим як використовувати віртуальний пристрій його необхідно завантажими та налаштувати. Емулятор надає практично всі можливості реального пристрою Android. Користувач може імітувати вхідні дзвінки так текстові повідомлення, вказувати місцезнаходження пристрою, імітувати обертання екрану та інші апаратні датчики, мати доступ до Google Play та інше. Тестування додатку на віртуальному пристрої в деякій мірі зручніше та швидше, ніж на фізичному пристрої. Наприклад, ви можете передавати дані в емулятор швидше, ніж на підключений фізичний пристрій.


Рисунок 3.6 — Віртуальний девайс в Android Studio

## 3.5. Система автоматизованої збірки Gradle

Процес створення програмного продукту для розробників та тестувальників програмного забезпечення без засобів автоматизації є дуже повторюваним, нудним та значно підвищує ризик помилок. Кожен крок в процесі розробки програмного продукту, починаючи з джерела компілювання коду до завершальної збірки програмного забезпечення для релізу та перевірки на виробництві потрібно робити вручну. Автоматизація проекту допомагає знизити кількість ручного втручання в проект, робить процес розробки більш ефективним та мінімізує можливість виходу програмного забезпечення з ладу.

Автоматизація проекту має кілька очевидних переваг для процесу розробки програмного продукту. Перша із них це запобігання ручному втручанню в процес збірки. Необхідність вручну виконувати дії для створення та постачання програмного забезпечення вимагає багато часу і підвищує ризик виникнення помилок. Розробник та системний адміністратор мають багато ізних задач, які вони могли б вирішувати в той час коли займаються компіляцією вручну. Буль-який крок в процесі розробки який може бути автоматизований має бути автоматизований. Наступною перевагою є створення повторюваних збірок. Звичайна побудова програмного проекту відповідає попередньо визначеним і впорядкованим крокам. Наприклад, розробнику необхідно спочатку скомпілювати вихідний код, потім запустити тести і нарешті зібрати проект з отриманням певного результату. Доведеться виконувати ті самі кроки знову і знову кожного дня. Цей процес має бути таким же легким як натискання кнопки. Результат цього процесу має повторюватися для всіх, хто запускає збірку проекту. Також до переваг можна віднести портативні збірки. Можливість запуску збірки з IDE дуже обмежена. По-перше, розробнику програмного продукту необхідно встановити на своєму пристрої  певний продукт. По-друге, IDE може бути доступна лише для певної операційної системи. Автоматизована збірка не вимагає певного середовища виконання, незалежно від того, чи це операційна система або середовище розробки. Оптимально, автоматизовані завдання повинні виконуватися

з командного рядка, який дозволяє запускати процес збірки з будь-якого пристрою та в будь-який час.

Автоматизація проекту в свою чергу ділиться на декілька основних типів. Перший із них це збірка на вимогу. Типовий випадок використання автоматизації за вимогою, це випадок коли користувач запускає побудову на певному пристрої. Зазвичай система контролю версій VSC керує версифікацією збірок та файлами висхідного коду. У більшості випадків користувач виконує скрипт у командному рядку, який виконує завдання у попередньо визначеному порядку. Наприклад, компіляція вихідного коду, копіювання файлу з одного каталогу до іншого або збірка результату. Зазвичай цей тип автоматизації виконується по кілька разів на день. Наступним типом автоматизації є запущені збірки. Якщо розробник займається гнучкою розробкою програмного забезпечення, його цікавить швидке отримання зворотного зв’язку про стан проекту. Також важлива інформація про те, чи може вихідний код бути зібраним без будь-яких помилок та інформація про наявність в проекті потенційного програмного дефекту помічений в помилці блоку чи тесті інтеграції. Цей тип автоматизації зазвичай спрацьовує  в результаті перевірки коду системою контролю версій. Ще одним типом автоматизації є заплановані збірки. Задумка полягає в плануванні автоматизації як планування завдань на основі часу. Вона виконується в певні проміжки часу або в певний конкретний час. Запланована автоматизація зазвичай виконується на виділеному сервері. Такий вид автоматизації особливо корисний для створення звітів або документації для проекту. Практика реалізації запланованих та запущених збірок, зазвичай визначається як безперервна інтеграція CI.

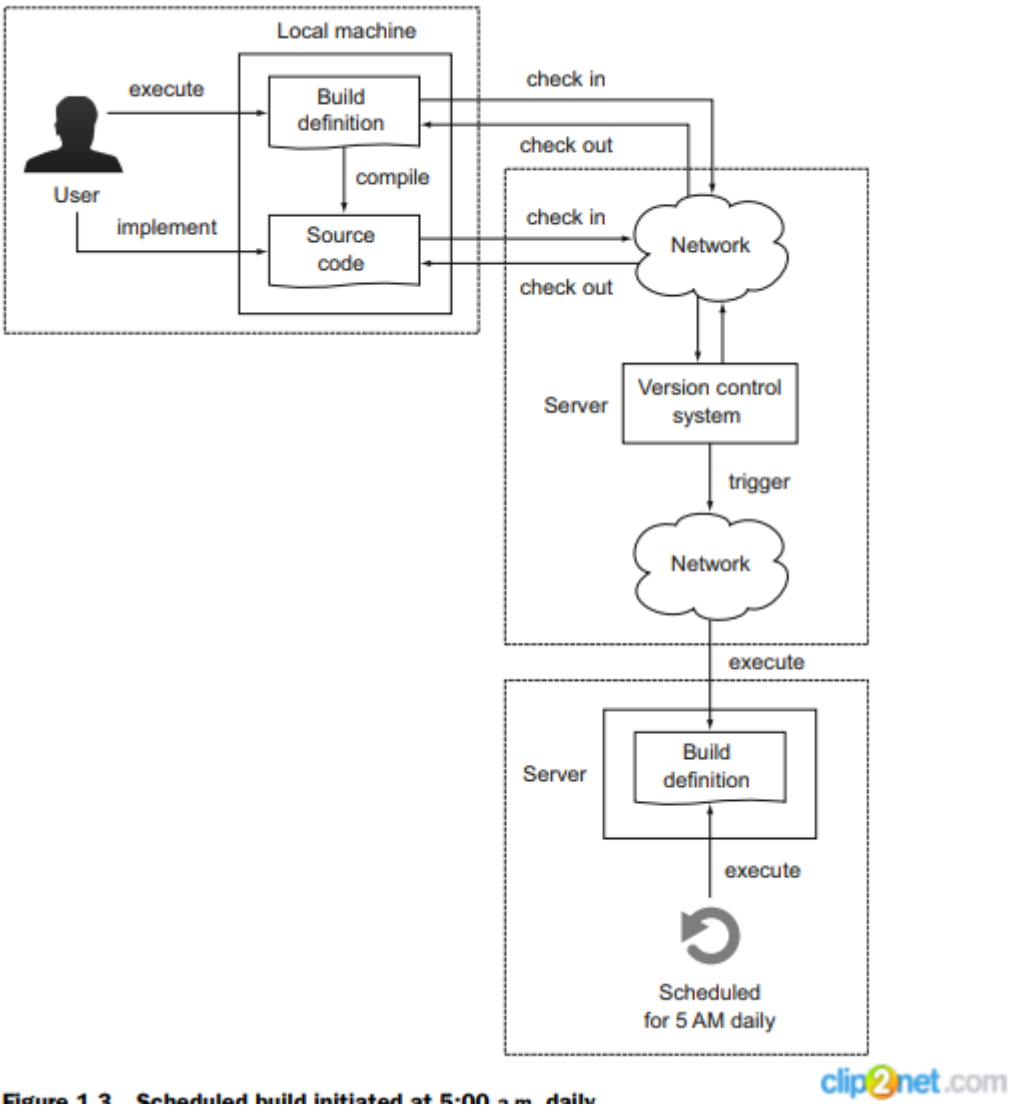


Рисунок 3.7 — Приклад запланованої збірки

Протягом багатьох років збірки мали прості вимоги до складання та пакування програмного забезпечення. Проте ландшафт сучасної розробки програмного забезпечення змінився і з’явилася потреба в автоматизації побудови проектів. В наш час проекти включають великі та різноманітні програмні стеки, включають велику кількість мов програмування та застосовують широкий спектр стратегій тестування. З підвищенням гнучких практик, збірки повинні підтримувати якнайшвидшу інтеграцію коду, а також швидке та легке постачання продукту в тестові та виробничі середовища. Встановлені інструменти побудови зазвичай не досягають цих цілей в простому вигляді. Оскільки розробнику програмного продукту потрібно чимало часу для того щоб розібратися з XML кодом збірки. Також не було можливості задати

власну логіку побудови, при додаванні сценарію збірки його неможливо було змінити.

Проте є спосіб зробити ці речі простими і виразними, це – Gradle. Система автоматизованої збірки Gradle вперше була випущена в 2007 році. Gradle є наступним еволюційним кроком в інструментах побудови JVM. Ця система спирається на досвід вже існуючих раніше інструментів автоматизованої побудови, таких як Ant та Maven, та переносить їхні найкращі ідеї на наступний рівень. Слідуючи підходу по побудови, Gradle дозволяє декларативно моделювати вашу предметну область за допомогою потужної та експресивної доменної мови DSL, реалізованої в Groovy замість XML. Оскільки Gradle є рідним до JVM, вона дозволяє писати власну логіку мовою Groovy або Java. В мові Java існує неймовірно велика кількість бібліотек і фреймворків. Управління залежностями використовується для автоматичного завантаження цих артефактів зі сховища і надання їх до коду програми. Дізнавшись про недоліки існуючих рішень управління залежностями, Gradle забезпечує власну реалізацію. Він не тільки добре налаштований, але й прагне бути максимально сумісним з існуючими інфраструктурами управління залежностями, такими як Maven та Ivy. Gradle надає потужну підтримку для визначення та організації багато проектних збірок, а також моделювання залежностей між проектами.

Для розробника автоматизація проекту є частиною повсякденної роботи. Сценарії побудови Gradle є декларативними, читабельними і чітко виражають свій намір. Написання коду в Groovy замість XML з елементами філософії Build-by-Convention, дозволяє значно скоротити розмір сценарію збирання і є набагато більше читабельним , що зображено на рисунку 3.8

Maven

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
                             http://maven.apache.org/xsd/maven-4.0.0.xsd">
   <modelVersion>4.0.0</modelVersion>
   <groupId>com.mycompany.app</groupId>
   <artifactId>my-app</artifactId>
   <packaging>jar</packaging>
   <version>1.0-SNAPSHOT</version>

   <dependencies>
      <dependency>
         <groupId>junit</groupId>
         <artifactId>junit</artifactId>
         <version>4.11</version>
         <scope>test</scope>
      </dependency>
   </dependencies>
</project>
```

Gradle

```
apply plugin: 'java'
group = 'com.mycompany.app'
archivesBaseName = 'my-app'
version = '1.0-SNAPSHOT'

repositories {
   mavenCentral()
}

dependencies {
   testCompile 'junit:junit:4.11'
}
```



Рисунок 3.8 — Порівняння скриптів Maven та Gradle

Кожна побудова Gradle починається з сценарію. Стандартне найменування сценарію для скрипту Gradle є build.gradle. При виконанні базової команди в оболонці система збірки шукає файл з саме таким іменем. Якщо його не буде знайдено буде відображено повідомлення про помилку. Після створення цього файлу необхідно визначити одне або декілька завдань.

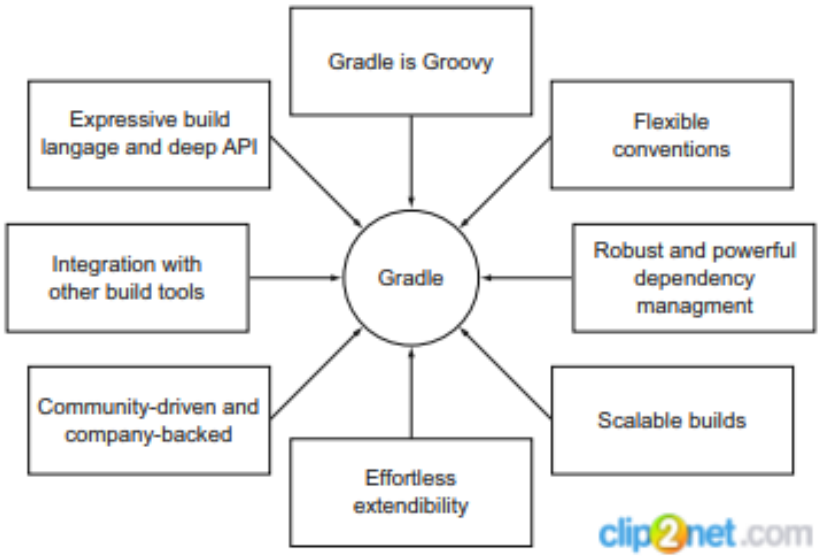


Рисунок 3.9 — Набір функцій системи Gradle

Підводячи підсумок, можна визначити, що Gradle – це автоматизована система збірки, яка походить від декларативного і виразного Groovy DSL. Вона поєднує в собі гнучкість і простоту розширення з ідеєю про конфігурацію і підримку традиційного управління залежностями. Gradle стає першочерговим вибором для побудови багатьох проектів з відкритим кодом.

## 3.6. Висновки до розділу

В даному розділі було виконано аналіз та огляд програмних технологій та засобів, які були використані в процесі створення програмного продукту, обґрунтований вибір використовуваних засобів при розробці програмного додатку.

# 4. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Система дистанційного навчання 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 4.1 наведена схема структури системи, на якій розташовані всі програмні модулі.

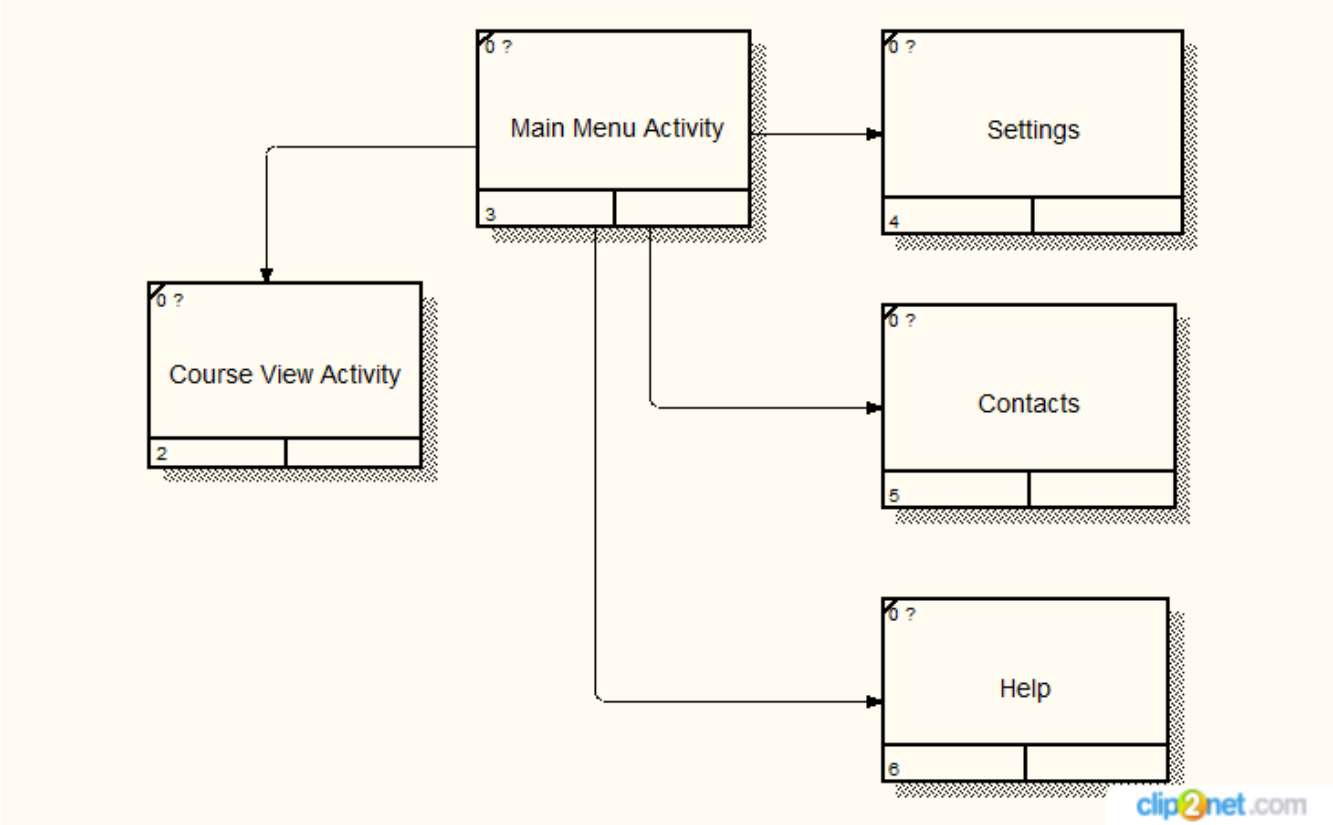


Рисунок 4.1 — Схема структури системи

## 4.1. Опис функціональності системи

Система дистанційного навчання 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 4.2 представлена діаграма прецедентів, яка описує функції та дії акторів у системі дистанційного навчання.

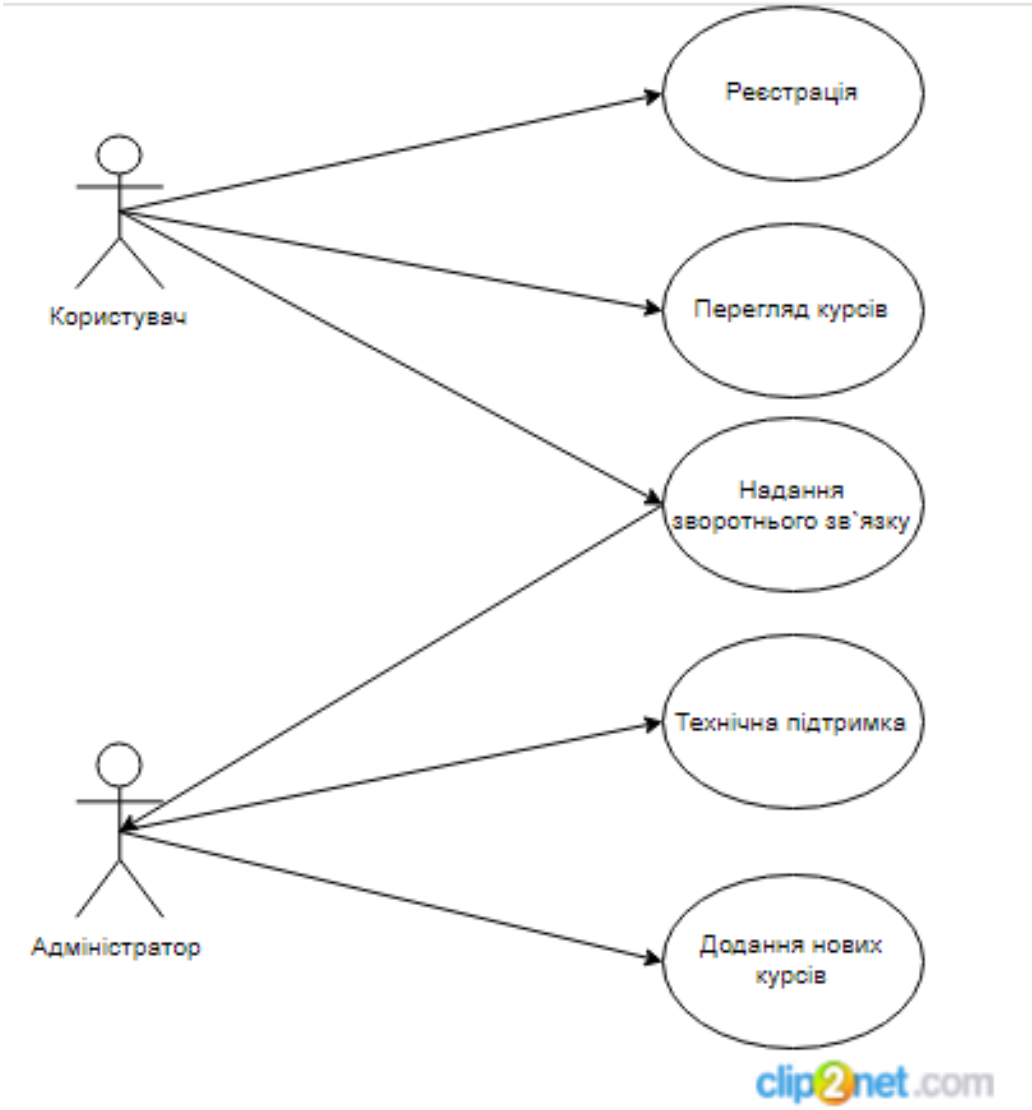


Рисунок 4.2 — Діаграма прецедентів системи

## 4.2.Розробка інтерфейсу користувача

Графічний інтерфейс користувача має відповідати наступним вимогам: ясність, відповідність, виразність, послідовність. Для забезпечення цих вимог був розроблений інтерфейс програмного продукту. Він включає в себе декілька модулів. Це головне меню, сторінка курсу, налаштування, контакти та довідник з користування системою. Коли користувач відриває додаток перед ним з'являється головне меню в якому він може обрати певний курс. Всі курси розташовані по порядку. Після того як користувач обирає курс, він переходить на вікно курсу, в якому відображається вся інформація про курс, а також навчальні матеріали. Допоміжне вікно налаштувань дозволяє користувачу персоналізувати додаток для зручності використання. Вікно контактів надає користувачу інформацію про способи зв'язку з розробниками. Вікно довідника створено для надання інформації та інструкції з використання системи.

### 4.2.1.Вікно головного меню

Дане вікно слугує для виведення на екран всіх курсів і надання користувачу можливості вибору одного із них. Воно складається з декількох елементів, а саме:

— назва додатку у верхній панелі ;

— кнопка виклику контекстного меню з допоміжними вікнами;

— список доступних курсів;

— клавіша текстового повідомлення.

### 4.2.2.Вікно Курсу

Дане вікно призначене для виведення інформації про курс на екран , а також контенту що міститься в курсі. Воно складається з структури елементів курсу, а саме:

— зображення курсу;

— заголовок курсу;

— назва спеціальності, для якої був створений курс ( кожна спеціальність відповідно має свій код);

— детальний опис курсу;

— інформація про викладача курсу;

— зміст курсу;

— панель для виведення відео лекцій;

### 4.2.3.Вікно налаштувань

Дане вікно призначене для зміни налаштувань системи та її персоналізації. Воно складається з декількох елементів, а саме:

— зміна оформлення ;

— зміна вигляду головного меню;

— зміна розташування відеолекцій в курсі;

— зміна шрифту та його розміру.

### 4.2.4.Вікно контактів

Дане вікно призначене для надання користувачу інформації для зв’язку з розробниками . Воно складається з декількох елементів, а саме:

— контактний адрес поштової скриньки;

— контактний телефон;

— адреса веб-сторінки.

### 4.2.5.Вікно довідника

Дане вікно призначене для надання користувачу інформації з користування системою . Воно складається з наступних елементів:

— версія системи;

— опис користування системою;

— інформація про розробника.

## 4.3. Висновки до розділу

У розділі було з’ясовано роль розроблюваного програмно-апаратного агенту у складі системи дистанційного навчання, виділені шляхи розвитку агенту у подальших розробках. Приведена архітектура, опис основних змінних і функцій агенту, схема збору апаратної частини розроблюваної системи.

# 5. РОБОТА КОРИСТУВАЧА В СИСТЕМІ

Система дистанційного навчання розроблена для використання на мобільних пристроях (смартфони та планшети) під управлінням операційної системи “Android” з використанням підключення до мережі інтернет ті взаємодією з інтерактивними елементами.

## 5.1 Інсталяція та системні вимоги

Додаток встановлюється на пристрій користувача шляхом встановлення “apk” файлу та його автоматичного налаштування. Оскільки додаток розроблений на платформі “Android”, пристрій користувача повинен цю операційну систему на своєму пристрої. Для коректної роботи додатку необхідно мати версію операційної системи не нижче 4.0.0. Також на пристрої користувача повинен бути доступ до мережі Інтернет.

## 5.2 Інструкція з використання програмного продукту

При вході в систему дистанційного навчання перед користувачем з’являється головне меню, в якому користувачу необхідно обрати один із запропонованих курсів, або знайти курс за обраною ним спеціальністю. Курси розташовані у формі списку. Головне меню меню зображено на рисунку 5.1.


Рисунок 5.1 — Головне меню

Після того як користувач обрав один із запропонованих курсів, він натискає на цей курс. Перед ним з’являється вікно курсу. Користувачеві надається інформація про курс, про викладача курсу а також змісту курсу. Для перегляду відео лекції користувачеві необхідно натиснути на її зображення. Вікно курсу зображено на рисунку 5.2.

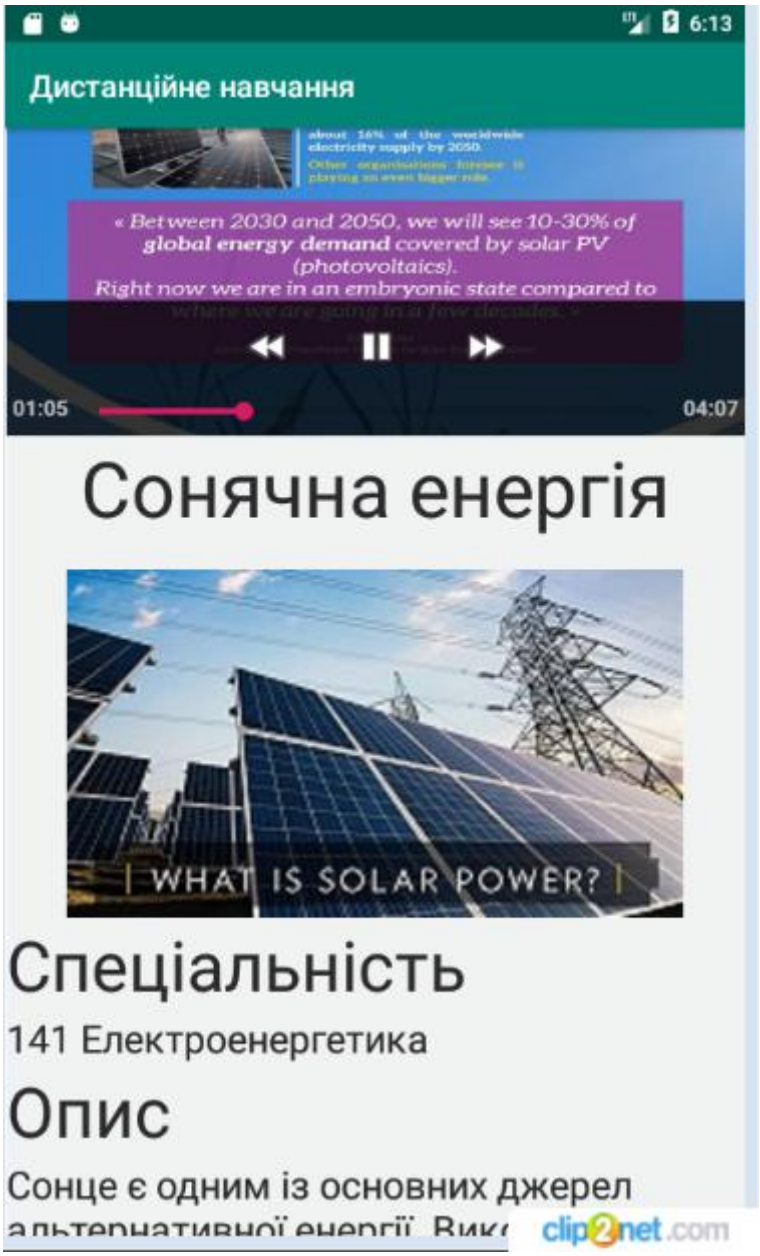


Рисунок 5.2 — Вікно курсу

## 5.3. Висновки до розділу

Розроблений інтерфейс зручний у користуванні, стійкий до внесення користувачем невірних даних та помилкових дій, при виникненні помилок у роботі системи виводить повідомлення про способи їх усунення, має приємне кольорове оформлення.

# ВИСНОВКИ

Під час практики був проведений аналіз існуючих систем дистанційного навчання, вивчення їх функціональних можливостей та методів реалізації. В результаті цього аналізу було виявлено, що існуючі системи не відповідають вимогам, а також здебільшого мають складну структуру.

В процесі проходження практики був спроектований та розроблений програмний продукт, що повністю задовольняє вимоги. Додаток надає користувачу доступ до Відеозаписів лекцій курсів, що викладаються у вищих навчальних закладах. Це дозволяє в повному обсязі отримати знання з певної спеціальності.

Було досліджено методи і засоби програмної реалізації продукту. В результаті чого було обрано створення додатку для мобільної платформи “Android” на основі об’єктно орієнтованої парадигми програмування. Додаток розроблений на мові “Java” з використанням платформи “Android”. Це дає змогу підвищити гнучкість та зручність системи в процесі розробки а також у процесі використання.

Розроблена система було випробувана на протестована і задовольняє всім програмним вимогам.

Додаток був розроблений для людей, які прагнуть отримувати освіту якості вищого навчального закладу, але через певні причини не можуть навчатися в університеті. Навчання відбувається дистанційно, при цьому не втрачається матеріал і структура навчального процесу.

Отже, в результаті практики було розроблено мобільний додаток, що задовольняє всім вимогам, Також було отримано нові знання з розробки програмних продуктів для мобільних систем. Було досліджено обсяг аудиторії яка використовує ті чи інші мобільні додатки та вплив структури системи на комфорт користування системою.

# 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

7. Martin Fowler — GUI Architectures. Часть 2 [Електронний ресурс]. — 2009 — Режим доступу: https://habr.com/post/53536/.

8. Benjamin Muschko — Gradle in Action [Електронний ресурс]. — 2014. — Режим доступу: https://www.manning.com/books/gradle-in-action

9. Murat Y. Expert Android Studio / Y. Murat, D. Onur., 2016.

10. Dream(sheep++): A developer’s introduction to Google Android [Електронний ресурс] – Режим доступу до ресурсу: https://arstechnica.com/gadgets/2009/02/an-introduction-to-google-android-for-developers/.

# ДОДАТОК 1

Розробка електронної системи дистанційного навчання

Специфікація

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ41161__18Б

Аркушів 1

Київ 2019

| Позначення | Найменування | Примітки |
|---|---|---|
| Документація | | |
| УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТІ41161_18Б 81-1 | Записка.docx | Пояснювальна записка |
| УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ41161 _18Б 13-1 | About.docx | Опис програмного модуля |
| Компоненти | | |
| УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ41161_ 18Б 12-2 | App-debug.apk | Клієнтський додаток |

# ДОДАТОК 2

Розробка електронної системи дистанційного навчання

Текст програмного модуля

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ41161_ 18Б 12-2

Аркушів 13

Київ 2019

```
package ua.bondar.project3;

import android.content.Intent;
import android.graphics.drawable.Drawable;
import android.os.Bundle;
import android.support.design.widget.FloatingActionButton;
import android.support.design.widget.Snackbar;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.TextView;

import static ua.bondar.project3.R.id.php;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {

    TextView php;
    TextView solar;
    TextView finance;
    TextView automotive;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toolbar toolbar = findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
```

```
php = (TextView) findViewById(R.id.php);
solar = (TextView) findViewById(R.id.solar);
finance = (TextView) findViewById(R.id.finance);
automotive = (TextView) findViewById(R.id.automotive);




php.setOnClickListener(new View.OnClickListener() {
   public void onClick(View view) {
     Intent myIntentphp = new Intent("ua.bondar.intent.action.phpAction");
          startActivity(myIntentphp);
     //courseMain.setCompoundDrawablesWithIntrinsicBounds(R.drawable.viper1,0,0,0);
   }
});
solar.setOnClickListener(new View.OnClickListener() {
   public void onClick(View view) {
      Intent myIntentsolar = new Intent("ua.bondar.intent.action.solarAction");
      startActivity(myIntentsolar);
   }
});
finance.setOnClickListener(new View.OnClickListener() {
   public void onClick(View view) {
      Intent myIntentphp = new Intent("ua.bondar.intent.action.financeAction");
      startActivity(myIntentphp);
   }
});
automotive.setOnClickListener(new View.OnClickListener() {
   public void onClick(View view) {
      Intent myIntentphp = new Intent("ua.bondar.intent.action.automotiveAction");
```

```
            startActivity(myIntentphp);
        }
    });



    FloatingActionButton fab = findViewById(R.id.fab);
    fab.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Snackbar.make(view, "Replace with your own action", Snackbar.LENGTH_LONG)
                    .setAction("Action", null).show();
        }
    });
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    int id = item.getItemId();

    //noinspection SimplifiableIfStatement
    if (id == R.id.action_settings) {
        return true;
    }

    return super.onOptionsItemSelected(item);
```

```
        }

        @Override
        public void onClick(View v) {

        }
    }
```

```
package ua.bondar.project3;

import android.content.Intent;
import android.net.Uri;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.MediaController;
import android.widget.TextView;
import android.widget.VideoView;

public class CourseActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_course);
        Intent intent = getIntent();
        String action = intent.getAction();
        VideoView videoView = (VideoView) findViewById(R.id.videoView1);
        TextView main_title = (TextView) findViewById(R.id.courseMain);
        TextView speciality_title = (TextView) findViewById(R.id.speciality_title);
        TextView speciality = (TextView) findViewById(R.id.speciality);
        TextView description_title = (TextView) findViewById(R.id.description_title);
        TextView description = (TextView) findViewById(R.id.description);
        TextView insctructor_title = (TextView) findViewById(R.id.instructor_title);
        TextView instructor = (TextView) findViewById(R.id.instructor);
```

```
TextView content_title = (TextView) findViewById(R.id.content_title);
TextView content = (TextView) findViewById(R.id.content);


if (action.equals("ua.bondar.intent.action.phpAction")) {
    main_title.setText(getString(R.string.course1));
    main_title.setCompoundDrawablesWithIntrinsicBounds(0,0,0,R.drawable.php1);
    speciality.setText(getString(R.string.speciality1));
    description.setText(getString(R.string.description1));
    instructor.setText(getString(R.string.instructor1));
    content.setText(getString(R.string.content1));
}
else if (action.equals("ua.bondar.intent.action.solarAction")) {
    main_title.setText(getString(R.string.course2));
    main_title.setCompoundDrawablesWithIntrinsicBounds(0,0,0,R.drawable.solar1);
    speciality.setText(getString(R.string.speciality2));
    description.setText(getString(R.string.description2));
    instructor.setText(getString(R.string.instructor2));
    content.setText(getString(R.string.content2));
}
else if (action.equals("ua.bondar.intent.action.financeAction")) {
    main_title.setText(getString(R.string.course3));
    main_title.setCompoundDrawablesWithIntrinsicBounds(0,0,0,R.drawable.finance1);
    speciality.setText(getString(R.string.speciality3));
    description.setText(getString(R.string.description3));
    instructor.setText(getString(R.string.instructor3));
    content.setText(getString(R.string.content3));
}
else if (action.equals("ua.bondar.intent.action.automotiveAction")) {
    main_title.setText(getString(R.string.course4));
    main_title.setCompoundDrawablesWithIntrinsicBounds(0,0,0,R.drawable.automotive1);
    speciality.setText(getString(R.string.speciality4));
    description.setText(getString(R.string.description4));
    instructor.setText(getString(R.string.instructor4));
    content.setText(getString(R.string.content4));
```

```
        }
        String videoSource ="android.resource://" + getPackageName() + "/" + R.raw.solarvideo1;
        Uri uri = Uri.parse(videoSource);
        videoView.setVideoURI(uri);
        MediaController mediaController = new MediaController(this);
        videoView.setMediaController(mediaController);
        mediaController.setAnchorView(videoView);




    }
}
```

# ДОДАТОК 3

Розробка електронної системи дистанційного навчання

Опис програмного модуля

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ41161_ 18Б 13-1

Аркушів 10

Київ 2019

# АНОТАЦІЯ

У додатку надана інформація про програмну частину системи дистанційного навчання.

Програмний продукт являє собою клієнтський додаток, створений на мові програмування Java з використанням системи автоматизованої збірки Gradle на базі програмної платформи Android.

Клієнтський додаток надає можливість користувачеві проходими дистанційно навчальні курси використовуючи мобільний девайс, знаходити курси за певною спеціальністю.

# ЗМІСТ

# ЗАГАЛЬНІ ВІДОМОСТІ

Програма являє собою мобільний додаток - електронну систему дистанційного навчання

Для використання програми необхідний мобільний девайс під управлінням системи Android.

Програмний продукт клієнтський додаток, створений на базі програмної платформи Android за допомогою інтерфейсу програмування додатків Android Studio. Клієнтський додаток запрограмований на мові Java.

Для розробки програмної системи було використано середовище розробки Android Studio IDE.

Для запуску клієнтського додатку необхідна операційна система сімейства Android версії не нижче 4.0 .

# ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Основним призначенням даного програмного забезпечення є надання споживачу електричної енергії зручного засобу для деталізованого відслідковування і збереження інформації про електричне споживання і навантаження.

Мобільний додаток реалізує наступні функції:

- створення курсів на наповнення їх навчальними матеріалами за заданою структурою
- забезпечення користувача контентом , яким наповнений додаток
- сповіщення користувача про оновлення курсу
- структуризація курсів за спеціальностями
- зручний та зрозумілий графічний інтерфейс для комфортного користування

# ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Мобільний додаток взаємодіє з користувачем за допомогою екрану пристрою та оброблення натискань.

При запуску активності виконується функція protected void onCreate() в якій задаються основні параметри та формат активності.

Функція setContentView() встановлює розмітку обраної активностей з ресурсів мобільного додатку.

Функція setOnClickListener обробляє натискання користувача та в залежності від місця натискання виконує відповідні дії .

# ВИКЛИК І ЗАВАНТАЖЕННЯ

Для встановлення мобільного додатку необхідно встановити завантажуваний APK файл. Після цього запустити його та чекати встановлення програми на мобільний девайс.

Коли мобільний додаток встановлено, його можна запускати за допомогою зображення на робочому столі.

# ВХІДНІ ТА ВИХІДНІ ДАНІ

Мобільний додаток приймає в якості вхідних даних дії користувача, і в залежності від конкретної дії обробляє натискання . Також вхідними даними є дані користувача. В якості вихідних даних додаток надає користувачу контент, який містять в собі курси.  Контент може бути різного формату, наприклад, текст, аудіо, відео.