UNICHECK

Ім'я користувача:
приховано налаштуваннями конфіденційності

ID перевірки:
1015591272

Дата перевірки:
13.06.2023 20:19:26 EEST

Тип перевірки:
Doc vs Library

Дата звіту:
13.06.2023 20:19:56 EEST

ID користувача:
100011372

Назва документа: **КН320 Когут Настя**

Кількість сторінок: **28** Кількість слів: **4785** Кількість символів: **37804** Розмір файлу: **1.12 MB** ID файлу: **1015240448**

6.35%
Схожість

Найбільша схожість: **3.09%** з джерелом з Бібліотеки (ID файлу: **1013082348**)

Пошук збігів з Інтернетом не проводився

6.35% Джерела з Бібліотеки | 158 ..... Сторінка 30

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи | 239

1 ОГЛЯД СУЧАСНИХ СИСТЕМ ОНЛАЙН НАВЧАННЯ

Актуальність сучасних онлайн систем навчання нещодавно значно збільшилась через розвиток технологій та поширення Інтернету. Онлайн системи навчання найкраще працюють як додатковий ресурс або альтернатива традиційному навчанню. Вони стали важливим інструментом для здобуття освіти і розвитку навичок в різних сферах життя.

1.1 Класифікація та характеристика сучасних онлайн систем навчання

Дистанційні системи навчання (Distance Learning Systems) - це системи, які дозволяють здійснювати навчання на відстані, без необхідності фізично присутнього контакту між викладачем та студентом. Основним принципом дистанційного навчання є використання технологій зв'язку та інформаційних технологій для забезпечення доступу до навчального матеріалу та взаємодії між учасниками навчального процесу.

Сучасні онлайн системи навчання можна класифікувати за різними критеріями. Ось декілька основних класифікаційних підходів:

1. Технологічний підхід:

- Синхронні системи навчання: Вони передбачають реальний час спілкування між викладачем і студентами, наприклад, за допомогою відеоконференцій або вебінарів. Всі учасники зустрічаються в один і той же час і можуть взаємодіяти між собою.
- Асинхронні системи навчання: Вони надають студентам можливість вивчати матеріали та виконувати завдання у зручний для них час. Комунікація з викладачем та іншими студентами може відбуватися через форуми, електронну пошту або інші інструменти.

2. Види систем:

- Відкриті масивні онлайн курси (MOOCs): MOOC-платформи, такі як Coursera, edX, FutureLearn, надають доступ до великої кількості безкоштовних або платних курсів з різних предметів. Ці платформи зазвичай пропонують відеоуроки, інтерактивні завдання, спільноти для обговорень та сертифікати після успішного завершення курсу.
- Віртуальні навчальні середовища (LMS): Це спеціальні платформи, які дозволяють установам освіти створювати, керувати та доставляти онлайн курси. Найпопулярніші LMS-системи включають Moodle, Blackboard, Canvas тощо. Вони мають функції для розміщення матеріалів, завдань, форумів, оцінювання та взаємодії зі студентами.
- Додаткові інструменти для навчання: Це різноманітні додаткові онлайн інструменти, які можуть використовуватися для навчання, такі як відеоплатформи, спеціалізовані додатки для практичних завдань, мобільні додатки тощо.

3. Орієнтація (зацікавлені сторони):

- Академічні системи навчання: Ці системи спрямовані на навчання в академічних середовищах, таких як університети, школи, коледжі. Вони надають інструменти для керування курсами, оцінювання студентів, спілкування з викладачами тощо.
- Корпоративні системи навчання: Ці системи призначені для навчання та розвитку персоналу в бізнес-середовищі. Вони надають інструменти для навчання, тренування, оцінювання та відстеження прогресу співробітників.
- Індивідуальні системи навчання: Ці системи використовуються для навчання на індивідуальному рівні, де студенти можуть працювати самостійно із доступом до навчального матеріалу та завдань.

4. Формат навчання:

- Онлайн-курси (Online Courses): Це самостійні курси, які студент може вивчати в Інтернеті. Вони можуть включати відеоуроки, тексти, завдання та оцінювання.

- Вебінари (Webinars): Живі онлайн-лекції або презентації, в яких учасники можуть брати участь в реальному часі, спілкуватися та задавати питання викладачу.
- Віртуальні класи (Virtual Classrooms): Це онлайн-середовища, де студенти та викладачі можуть взаємодіяти в режимі реального часу, проводити уроки, дискусії та спільні проекти.
- Соціальне навчання (Social Learning): Використання соціальних мереж та форумів для спілкування, обміну ідеями та співпраці між студентами та викладачами.

5. Функціональність:

- Відеоуроки та інтерактивні матеріали: Системи, які надають доступ до відеоуроків, інтерактивних матеріалів, тестів та іншого навчального контенту.
- Завдання та оцінювання: Системи, які дозволяють створювати та оцінювати завдання, тести та інші форми оцінювання студентів.
- Комунікація та співпраця: Системи, що надають інструменти для комунікації, співпраці та обміну ідеями між студентами та викладачами.

6. Рівень інтерактивності:

- Пасивні системи: Ці системи надають студентам доступ до попередньо записаних відеолекцій, електронних матеріалів та завдань, але не передбачають активної взаємодії з викладачем або іншими студентами.
- Інтерактивні системи: Ці системи надають можливості для взаємодії між студентами та викладачами, такі як форуми обговорень, онлайн-чати, відеоконференції тощо. Вони сприяють активній спільноті навчання та обміну ідеями.

7. Модульні системи:

- Модульні системи навчання дозволяють студентам вивчати матеріал поетапно, розбиваючи його на окремі модулі або теми. Кожен модуль може містити відеоуроки, підручники, завдання та тести. Студенти можуть пройти модулі в певному порядку та переходити до наступного тільки після успішного виконання завдань або тестів попереднього модуля.

8. Блендед-лернінг:

- Блендед-лернінг (змішане навчання) поєднує елементи онлайн навчання та традиційного навчання в класі. Студенти можуть вивчати частину матеріалу онлайн, а потім зустрічатися з викладачами та співстудентами для дискусій, практичних занять або вирішення задач в класі.

Це лише кілька основних критеріїв класифікації дистанційних систем навчання, і на практиці багато систем можуть поєднувати різні характеристики та підходи.

Сучасні онлайн системи навчання мають ряд характеристик, які роблять їх ефективними, а також дозволяють забезпечувати зручне дистанційне навчання.

Відкритий доступ: Онлайн системи навчання надають можливість отримати освіту та навчатися з будь-якого місця з доступом до Інтернету. Вони дозволяють студентам вибирати навчальний матеріал із великого розмаїття курсів та програм з різних дисциплін.

Гнучкість: Системи навчання дають можливість студентам самостійно планувати свій час та темп навчання. Вони можуть вивчати матеріал в зручний для них час і темпі, що дозволяє підлаштовувати навчання під особисті потреби та графік.

Різноманітність навчального контенту: Онлайн системи навчання надають доступ до різноманітного навчального контенту, який може включати відеоуроки, тексти, інтерактивні матеріали, тести, завдання та багато іншого. Це дозволяє студентам отримувати знання різними способами і використовувати різні типи навчальних матеріалів.

Взаємодія та співпраця: Багато онлайн систем навчання надають інструменти для взаємодії та співпраці між студентами та викладачами. Це можуть бути форуми для обговорення, чати, відеоконференції, спільна робота над проектами тощо. Такі інструменти сприяють активній комунікації та обміну ідеями між учасниками навчального процесу.

Персоналізація: Деякі системи навчання надають можливості для персоналізованого навчання, де студенти можуть вибирати свої предмети, рівень

складності та шлях навчання. Вони також можуть отримувати рекомендації щодо навчальних ресурсів та матеріалів, що відповідають їхнім інтересам та потребам.

Онлайн оцінювання та зворотний зв'язок: Системи навчання можуть містити інструменти для оцінювання знань студентів, такі як онлайн тести, завдання, курсові роботи тощо. Вони також надають можливість для викладачів надавати зворотний зв'язок та оцінювати прогрес студентів.

Моніторинг та статистика: Багато систем навчання надають можливість моніторингу активності студентів та збирання статистичних даних. Це дозволяє викладачам відстежувати прогрес студентів, аналізувати їхні результати та вносити необхідні корективи до навчального процесу.

Підтримка технологій: Сучасні онлайн системи навчання підтримують різні технології, такі як комп'ютери, смартфони, планшети, що дозволяє студентам отримувати доступ до навчального контенту з різних пристроїв та місць.

Ці характеристики можуть варіюватися залежно від конкретної системи навчання. Онлайн системи навчання постійно розвиваються та вдосконалюються, щоб надати студентам якісні та зручні можливості для отримання знань та освіти.

1.2 Огляд популярних систем онлайн навчання

Сучасні системи навчання охоплюють широкий спектр технологій та методів, що дозволяють підвищити ефективність та доступність освіти. Деякі з найбільш популярних та інноваційних систем навчання включають наступні:

E-learning — це система навчання, що використовується для навчання в Інтернеті, де студенти можуть вчитися віддалено з будь-якого місця з доступом до Інтернету. E-learning включає в себе електронні курси, відеоуроки, тестування та інші інтерактивні матеріали.

Coursera — це онлайн-платформа з доступом до курсів з більше ніж 3900 тем, що пропонуються більш ніж 200 університетами та компаніями зі всього світу. Курси можна проходити безкоштовно або за плату, залежно від бажання отримати сертифікат про успішне завершення курсу.

edX — це онлайн-платформа з доступом до курсів з більше ніж 2 500 тем, що пропонуються більш ніж 140 університетами та компаніями зі всього світу. Курси можна проходити безкоштовно або за плату, залежно від бажання отримати сертифікат про успішне завершення курсу.

Udemy — це онлайн-платформа з доступом до курсів з більше ніж 155 000 тем, що пропонуються викладачами з усього світу. Курси можна проходити за плату, проте ціни зазвичай набагато нижчі, ніж університетські курси.

Skillshare — це онлайн-платформа з доступом до більше ніж 25 000 курсів зі світу дизайну, технологій, мистецтва, бізнесу та багатьох інших тем. Курси можна проходити за плату, або з використанням місячної підписки.

Khan Academy — це онлайн-платформа з доступом до безкоштовних курсів з математики, науки, програмування та інших тем. Khan Academy спрямовується на надання безкоштовної освіти в усьому світі.

Codecademy — це онлайн-платформа з доступом до курсів з програмування Пропонує курси з більше ніж 12 різних мов програмування, включаючи Python, Java, JavaScript та інші. Курси можна проходити безкоштовно або за плату з отриманням сертифікату.

Treehouse — це онлайн-платформа з доступом до курсів з програмування, веб-дизайну та розробки мобільних додатків. Курси можна проходити за плату з місячною або щорічною підпискою.

Pluralsight — це онлайн-платформа з доступом до більше ніж 7 000 курсів з програмування, технологій, кібербезпеки та бізнесу. Курси можна проходити за плату з місячною або щорічною підпискою.

Lynda — це онлайн-платформа, що належить LinkedIn, з доступом до більше ніж 6 000 курсів з програмування, дизайну, бізнесу та інших тем. Курси можна проходити за плату з місячною або щорічною підпискою.

LinkedIn Learning — це онлайн-платформа, що належить LinkedIn, з доступом до більше ніж 16 000 курсів з програмування, технологій, бізнесу та інших тем. Курси можна проходити за плату з місячною або щорічною підпискою.

Google Classroom — безкоштовний веб-сервіс, створений Google для освітніх закладів з метою спрощення створення, поширення і класифікації завдань безпаперовим шляхом. Він об'єднує в собі: Google Drive для створення і обміну завданнями, Google Docs, Sheets and Slides для написання, Gmail для спілкування і Google Calendar для розкладу. Учитель може відстежувати прогрес кожного учня, а після оцінки його роботи, вчитель може повернути її разом з коментарями.

Microsoft Teams — центр для командної роботи в Office 365 від Microsoft, який інтегрує користувачів, вміст і засоби, необхідні команді для ефективнішої роботи. Застосунок об'єднує все в спільному робочому середовищі, яке містить чат для нарад, файлообмінник та корпоративні програмами.

2 ВИБІР ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ СТВОРЕННЯ ДИСТАНЦІЙНОГО КУРСУ

2.1 Вибір середовища розробки Android Studio

Android Studio є одним з найпопулярніших середовищ програмування для розробки мобільних додатків під платформу Android.

Рисунок 2.1 – Логотип середовища розробки Android Studio

Android Studio — це середовище розробки, призначене для широкого кола розробників, включаючи невеликі команди або навіть окремих розробників, які працюють над мобільними додатками, а також великі міжнародні організації, які використовують системи контролю версій, такі як GIT. Досвідчені розробники мають гнучкість у виборі інструментів, які найкраще відповідають їхнім потребам, особливо під час роботи над масштабними проектами. Рішення для Android розробляються в Android Studio за допомогою Java або C++.

В основі робочого процесу Android Studio закладено концепт безперервної інтеграції, що дозволяє відразу виявляти наявні проблеми.

Тривала перевірка коду забезпечує можливість ефективного зворотного зв'язку з розробниками. Така опція дозволяє швидше опублікувати версію мобільного додатка в GooglePlayAppStore. Для цього є також підтримка інструментів LINT, Pro-Guard і AppSigning.

За допомогою засобів оцінки продуктивності визначається стан файлу з пакетом прикладних програм. Візуалізація графіки дає змогу дізнатися, чи програма відповідає орієнтиру Google в 16 мілісекунд.

За допомогою інструмента для візуалізації пам'яті розробник дізнається, коли його додаток буде використовувати занадто багато оперативної пам'яті і коли відбудеться складання сміття.

Інструменти для аналізу батареї показують, яке навантаження лягає на пристрій.

AndroidStudio сумісна з платформою GoogleAppEngine для швидкої інтеграції у хмари нових API та функцій. У розробці є різні API, такі як Google Play, Android Pay і Health.

Є підтримка всіх платформ Android, починаючи з Android 1.6.

Ось до прикладу ще кілька причин, чому Android Studio є кращим вибором для розробки Android-додатків порівняно з іншими середовищами програмування:

- Офіційнесередовище: AndroidStudioрозробленокомпанієюGoogleякофіційнеінтегрованесередовищерозробки (IDE) дляAndroid. Цеозначає, щовонопрямопідтримуєтьсятаоновлюєтьсяGoogle, щозабезпечуєактуальністьтехнологійтавідповідністьстандартамплатформиAndr .
- Повнийфункціонал: AndroidStudioмаєбагатопотужнихфункційтаінструментів, спеціальнопризначенихдлярозробкиAndroid-додатків. Включаємосюдиредакторкодузпідсвічуваннямсинтаксису, автодоповненнямтапідтримкоювідлагодження, графічнийредакторінтерфейсукористувача, інструментидлярозгортаннядодатківтабагатоіншого.
- Ефективність: AndroidStudioзабезпечуєоптимізованийрозробчийпроцес. Вонопропонуєшвидкузбіркуізапускдодатків,

вбудованусистемузасобівдляпрофілюваннятавідлагодженняпродуктивності, щодопомагаютьвиявлятийусуватипроблемишвидкодії.

- ПідтримкаKotlin: AndroidStudioволодієпотужноюпідтримкоюмовипрограмуванняKotlin. KotlinєофіційноюмовоюрозробкидляплатформиAndroid, іAndroidStudioнадаєрозширеніможливостідлярозробкидодатківнаKotlin.
- СтворенняпідAndroid: AndroidStudioнадаєповнийнабірінструментівдляствореннярізноманітнихтипівдодатківдляAndroid, включаючимобільнідодатки, планшетнідодатки, AndroidTV-додатки, AndroidWear-додаткитощо.
- Живаспільнотарозробників: AndroidStudioмаєвеликутаактивнуспільнотурозробників, деможназнайтидопомогу, поради, кодовіфрагментитарішенняпроблем. Це сприяє навчанню та розвитку, а також допомагає вирішувати складні завдання.

Хоча існують інші IDE, які підтримують розробку Android-додатків, AndroidStudio видається найбільш повним, потужним та оновлюваним середовищем програмування для розробки під платформу Android.

При створенні нового проекту (рис. 2.2) або відкритті існуючого – відкривається головне вікно AndroidStudio, яке зображене на рисунку 2.3.

Рисунок 2.2– Створення нового проєкту в Android Studio

Рисунок 2.3 – Головне вікно проекту в середовищі AndroidStudio

В головному вікні ми можемо виділити наступні елементи:

- Рядок меню позначеним червоним кольором. Він містить в собі ряд меню, для виконання задач в межах середовища;
- Панель інструментів позначена голубим кольором. Являє собою ряд кнопок для дій, які часто виконуються. Цю панель можна налаштувати під себе;
- Панель навігації позначена жовтим кольором. Вона надає зручний доступ пересування по каталогам та файлам і є альтернативою вікна інструментів проекту;
- Вікно редактора позначене фіолетовим кольором. Воно відображає вміст файлу в якому знаходиться розробник і працює в даний час.
- Рядок стану позначений зеленим кольором. Він відображає інформаційні повідомлення про проект та дій AndroidStudio.
- Вікно інструментів позначене синім кольором. Воно надає нам ієрархічний вид файлу проекту та дозволяє здійснювати навігацію до певних файлів та каталогів.

AVD — це вбудований в середовище AndroidStudio емуляторпристрою. У процесі розробки додатків для Android в Android Studio необхідно буде скласти і запускати його чимало разів. AVD – це, по суті, віртуальний пристрій, що дозволяє додаткам Android проходити тестування без необхідності встановити його на фізичнийпристрій на базі Android.Певні пристрої мають свої специфікації, характеристики та додаткові функції, такі як акселерометр, NFC-оплата та інші. Саме тому, перевагою емулятора є те, що віртуальний пристрій ми можемо налаштувати під себе, враховуючи їх. Приклад AVD зображений на рисунку 2.4.

Рисунок 2.4 – Емулятор Android-пристрою

Візуальний інтерфейс Android-додатків завантажується із спеціальних файлів з розширенням .xml, які зберігають розмітку. Ці файли містять код написаний на мові розмітки XML. Оголошення користувацького інтерфейсу в XML-файлах дозволяє відділити інтерфейс додатку від коду. Тобто ми можемо змінювати інтерфейс без змін в Java-коді. Крім того, оголошення розмітки в XML-файлах дозволяють суттєво полегшити процес візуалізації структури інтерфейсу та полегшити його налагодження.

XML розшифровується як eXtensible Markup Language – “розширювана мова розмітки”. Іншими словами – XML– це мова розмітки для опису даних. Важливо розуміти, що XML – це не виконуваний код, так як після опису даних за допомогою XML, потрібно написати код, який зможе ці дані відправити, прийняти або обробити.

В середовищі AndroidStudioфайли розмітки графічного інтерфейсу розміщені в проекті в каталозі res/layout. В файлі визначаються всі графічні елементи та їх атрибути, з яких складається інтерфейс. По замовчуванню, при

створенні проекту в AndroidStudio вже є один файл activity_main.xml, який зображений на рисунку 2.5. Кожен XML-файл має містити в собі один корінний елемент, який має представляти об’єкт Viewабо ViewGroup. Об’єкт Viewформує на екрані елемент, з яким користувач може взаємодіяти. Об’єкт ViewGroupмістить в собі інші об’єкти View(та ViewGroup) для визначення макету інтерфейсу. Android Studio надає колекцію підкласів View та ViewGroup, яка містить як звичайні елементи вводу, такі як кнопки та текстові поля, так і різні моделі готових макетів.

Рисунок 2.5 – Файл activity_main.xml

Користувацький інтерфейс для кожного компонента додатку визначається за допомогою ієрархії об’єктів Viewта ViewGroup, яку ми можемо побачити на рисунку 2.6.

Рисунок 2.6 – Ієрархія визначення макету інтерфейсу

Кожна група перегляду є невидимим контейнером, в якому об’єднані дочірні види, при цьому, дочірні види можуть представляти собою елементу вводу та виводу або інші віджети, які складають частину користувацького інтерфейсу. Ця ієрархія може бути настільки простою або складною, наскільки це потрібно.

2.2 Розробка структури Web–додатку

Розробка структури додатку включає все, що стосується його змісту та інформаційної стратегії, яка визначає, як повинна бути організована подача інформації, задля прискорення та полегшення її пошуку майбутніми користувачами. Вона вимагаєуважногоплануваннятааналізувимог. Нижче наведено кроки, які можна виконати при розробці структури веб-додатку:

- Визначення вимог: вивчення вимог до веб-додатку. Це можуть бути функціональні вимоги (такі як функції, які має виконувати додаток) і технічні вимоги (такі як мови програмування, фреймворки, база даних тощо).
- Створення концептуальної моделі: розробка концептуальної моделі додатку, яка включає основні компоненти і їх взаємозв'язки. Наприклад, модель може включати клієнтську частину (frontend), серверну частину (backend), базу даних і зовнішні сервіси.
- Розробка архітектури: вибір відповідної архітектури для додатку. Наприклад, популярні архітектурні підходи включають клієнт-сервер, одношарову

архітектуру, багатошарову архітектуру (наприклад, MVC або MVVM) тощо. Розробка схеми взаємодії між компонентам.

- Вибір технологій: вибір технології, яка найкраще підходять для реалізації додатку. Це можуть бути мови програмування, фреймворки, бази даних, бібліотеки тощо. Врахування вимог до продуктивності, масштабованості та інші фактори при виборі технологій.
- Розбиття на компоненти: поділ додатку на окремі компоненти або модулі. Наприклад, клієнтська частина (frontend) може містити різні сторінки, компоненти і файли ресурсів. Серверна частина (backend) може мати окремі модулі для обробки запитів, бізнес-логіки та доступу до бази даних.
- Організація файлової структури: Створіть логічну файлову структуру для вашого додатку. Врахуйте модулярність, повторне використання коду та легкість розширення. Наприклад, розділіть файли на основі функціональності або компонентів.
- Взаємодія між компонентами: Визначте способи взаємодії між різними компонентами вашого додатку. Наприклад, як клієнтська частина взаємодіє з сервером за допомогою API, як дані передаються між клієнтом і сервером, як база даних використовується для зберігання і отримання даних тощо.
- Тестування: планування включення тестування у процес розробки. Розробка стратегії тестування, яка включає функціональні тести, тести одиниць, інтеграційні тести тощо. Визначення того, як будуть автоматизовані тести.
- Розгортання та моніторинг: процес розгортання веб-додатку на виробничому сервері. Включення механізми моніторингу для відстеження продуктивності та проблем.

При створенні веб-додатків особлива увага приділяється декільком аспектам, щоб забезпечити успіх та зручність використання додатку користувачами. Ось деякі аспекти, до яких приділяється особлива увага при розробці веб-додатків:

- Користувацький досвід (User Experience, UX): Успішний веб-додаток повинен мати зручний та привабливий інтерфейс користувача. Розробка веб-

додатку повинна враховувати потреби, очікування та поведінку користувачів. Уява потрібна для створення простого у використанні та інтуїтивно зрозумілого інтерфейсу, який дозволяє користувачам легко навігувати, взаємодіяти з функціями та знаходити потрібну інформацію.

- Візуальний дизайн: Веб-додатки потребують естетичного та привабливого візуального дизайну, який залучає користувачів та створює позитивне враження. Розробка дизайну включає в себе вибір кольорової палітри, типографіки, іконок, графічних елементів та компонентів інтерфейсу. Важливо використовувати уяву, щоб створити привабливий та консистентний вигляд додатку.
- Відповідність різним пристроям: Веб-додатки повинні бути адаптивними та реагувати на різні розміри екранів та пристрої. З урахуванням широкого спектру пристроїв, таких як настільні комп'ютери, ноутбуки, планшети та смартфони, розробка веб-додатків вимагає уяви для створення адаптивного дизайну, який забезпечує оптимальний користувацький досвід на будь-якому пристрої.
- Інтерактивність та анімація: Використання анімації та інтерактивних ефектів може зробити веб-додаток більш привабливим та захоплюючим для користувачів. Уява допомагає враховувати можливості для використання анімаційних переходів, плавних рухів, ефектів наведення курсору тощо, щоб зробити взаємодію з додатком цікавою та приємною.
- Оптимізація продуктивності: Уява також важлива при плануванні та розробці оптимізації продуктивності веб-додатку. Потрібно уявити різні сценарії використання та навантаження на додаток, щоб виявити можливі проблеми продуктивності та вжити заходи для покращення швидкодії та реактивності.
- Безпека: При розробці веб-додатків важливо приділяти увагу безпеці. Уява допомагає уявити потенційні точки вразливості та ризики для забезпечення належного рівня захисту додатку та даних користувачів.

Уява грає важливу роль у всіх аспектах розробки веб-додатків, допомагаючи створити приємний та ефективний досвід користувачів, а також забезпечити високу якість та функціональність додатку.

Тому дизайн сторінки є ще одним важливим аспектом у розробці веб-додатку. Добре розроблений дизайн сторінки сприяє досягненню цілей веб-додатку. Незалежно від того, чи це продаж товару, реєстрація на сайті або звернення до контенту, дизайн повинен спрямовувати користувачів і надихати їх до виконання бажаних дій.Колірна схема, типографіка, використання логотипів та графічних елементів допомагають впізнаваності й створюють єдиний стиль.обре розроблений дизайн сторінки сприяє досягненню цілей веб-додатку. Незалежно від того, чи це продаж товару, реєстрація на сайті або звернення до контенту, дизайн повинен спрямовувати користувачів і надихати їх до виконання бажаних дій.

Отже, на основі аналізу всіх вище вказаних даних, а також сучасних систем онлайн навчання, було розроблено наступну структуру для андроїд-додатку.

UIчастина повинна містити наступні елементи:

- головна сторінка;
- навігація для переходу до уроків/тестів.

Для комфортної розробки потрібно визначити всі функції, які має виконувати додаток. Для цього було створено головну структурну схему розробки додатку, яка зображена на рисунку 2.7.

Рисунок 2.7 – Структура веб-додатку

Головна сторінка
Інформаційне забезпечення
Тести для перевірки
Корисні посилання для поглибленого вивчення

3 РОЗРОБКА ANDROID-ДОДАТКУ ДЛЯ ОРГАНІЗАЦІЇ ДИСТАНЦІЙНОГО НАВЧАННЯ З ДИСЦИПЛІНИ «WEB-ДИЗАЙН» НА МОВІ JAVA

3.1 Розробка мобільного додатку для дистанційного навчання

Дослідивши різні середовища для онлайн навчання та середовища розробки та створивши загальну структуру веб-додатку, потрібно її втілити фактично.

Для початку потрібно створити головну сторінку. Як шаблон, було використано EmptyViewsActivity (рис. 1).

Рисунок 3.1 – Вибір шаблону в AndroidStudio

Після початку створення проєкту, у файлі build.gradle (додаток А) потрібно підключити бібліотеку CardView. "CardView" є одним з компонентів Material Design, який надає можливість створення карток з тінями та закругленими куточками. Він зазвичай використовується для відображення окремих елементів або списків елементів у вигляді карток. Далі необхідно підключити механізм, якийдозволяє замінити традиційний підхід до пошуку та звертання до елементів інтерфейсу користувача (UI) в Android, такий як findViewById(), на більш зручні та безпечніші методи. Коли включено viewBinding true у конфігураційному файлі

build.gradle (додаток А), Android Studio генерує прив'язки до елементів UI в роздутих (inflated) макетах (layouts) автоматично. Це означає, що ви можна звертатись до елементів UI безпосередньо за допомогою згенерованих прив'язок, що забезпечує більш чистий та безпечний код.

Також було налаштовано колірну гаму для інтерфейсу програми у файлахcolor.xml тathemes.xml(додаток Б). У color.xml прописано кольори в шістнадцятковому коді. Файл themes.xml використовується для визначення тем оформлення (themes) для додатку Android. Він містить набір стилів, які використовуються для вигляду та поведінки різних компонентів додатку. Теми визначають зовнішній вигляд додатку, включаючи кольори, форма зображень тощо. Файл themes.xml містить різні стилі, які можуть бути використані для різних компонентів додатку або для всього додатку в цілому.

Підключивши всі необхідні бібліотеки, можна приступати до створення основної сторінки.

Спочатку було задано рамку для назви додатку. У файлі lavender_border.xmlдля цього було задано наступні параметри:

```
<stroke
    android:color="@color/lavender"
    android:width="2dp"/>
<corners
     android:topRightRadius="20dp"
     android:bottomRightRadius="20dp"/>
```

Після цього було розпочато роботу зі створення основного макету головного вікна (файл activity_main.xml — додаток В). Як фон використано зображення. Меню створене за допомогою властивості GridLayout. Ця функція дозволяє визначити розташування та організацію елементів на веб-сторінці або в інтерфейсі користувача (UI) з використанням сітки з колонок і рядків. Застосування цієї властивості дозволяє встановити більш точне і гнучке розташування елементів на сторінці. Для кнопок було використано векторні зображення. Для цього необхідно зайти в папку drawable, натиснути правою

кнопкою мишки викликаючи контекстне меню. Тоді обрати New → VectorAsset (рисунок 3.2).

Рисунок 3.2 – Створення значків (icons)для меню

Макет головної сторінки зображено на рисунку 3.3:

Рисунок 3.3 – Макет головної сторінки

Наступним кроком було зробити так, щоб ці кнопки були перехідними. Для цього у файлі MainActivity.java (додаток В) було використано метод .setOnClickListener. Коли користувач натискає на кнопку або інший елемент, якому було призначено .setOnClickListener, спрацьовує обробник подій, який було визначено. Ось приклад коду для кнопки з назвою «Навчання»:

```
lessonCard.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
```

```
        Intent intent = new Intent(MainActivity.this, lessonActivity.class);
        startActivity(intent);
    }
});
```

Налаштувавши головну сторінку, потрібно переходити до налаштувань сторінки кожної кнопки. Їх порядок налаштувань такий самий, як і попередньої сторінки — спочатку макет дизайну, а потім функціонал.

Для розділу «Навчання» було застосовано віджет ListView (файл activity_lesson.xml — додаток Г). Він дозволяє користувачеві прокручувати та взаємодіяти з великим списком даних, наприклад в нашому випадку — список порядку уроків, тощо. Кожен елемент списку відображається в окремому рядку, і можна визначити вигляд кожного рядка за допомогою адаптера. Властивості кожного рядка описано у файлі list_item.xmlза допомогою віджету RelativeLayout,який використовується для створення гнучкого та реактивного макету, де елементи розташовуються відносно один одного або відносно батьківського контейнера. Він дозволяє контролювати розміщення елементів на екрані та налаштовувати їх положення, вирівнювання та відступи.Результат роботи цих двох файлів можна побачити на рисунку 3.4:

Рисунок 3.4 – Макет сторінки зі списком

Для загальноговигляду макету кожної з частин списку було створено файл activity_detailed.xml (додаток Г). Використання елементу LinearLayoutдопомагає організувати елементи інтерфейсу в лінійній структурі. Він надає простий та гнучкий спосіб розташовування елементів, що дозволяє вам контролювати їх положення, відступи та вирівнювання. Для того, щоб вмістити довгий текст та для його прокручування, застосовано віджет ScrollView. Загальний вигляд макету зображено на рисунку 3.5.

Рисунок 3.5– Макет файлу activity_detailed.xml

Далі для зберігання рядків тексту, який буде використовуватись у подальшому, було використано файл strings.xml. Цей файл є частиною ресурсів (resources) проекту і дозволяє централізовано керувати текстовими рядками, що використовуються в додатку. Основна мета використання strings.xml полягає в тому, щоб відокремити текстові рядки від коду додатку. Використання strings.xml дозволяє легко змінювати текстові рядки в додатку, використовувати їх повторно і локалізувати додаток на різні мови, просто додавши нові файли strings.xml для кожної локалізації. Це добра практика в розробці Android-додатків, оскільки допомагає підтримувати код чистим, полегшує локалізацію і сприяє ефективній роботі з рядками тексту додатку.

Створивши потрібні файли для розмітки, було розроблено фунціонал додатку.

У файліListData.java (додаток Г)клас ListDataпредставляє об'єкт з даними для списку. Клас містить три поля: name (назва), lesson (кількість уроків) і image (зображення). Конструктор ListData приймає параметри name, ingredients і image і встановлює їх значення для відповідних полів об'єкта. Це дозволяє створювати об'єкти ListData і ініціалізувати їх поля з вказаними значеннями.

Файл ListAdapter.java (додаток Г)визначає клас ListAdapter, який розширює клас ArrayAdapter для використання зі списком (ListView) в Android. ListAdapter використано для зв'язування даних з елементами інтерфейсу списку.Основна роль ListAdapter полягає в тому, щоб визначити, які дані з'являються в кожному елементі списку і як вони відображаються. Для цього використано метод getView(), який перевизначений у класі ListAdapter.

При створенні об'єкта ListAdapter передаються контекст (Context), що представляє посилання на поточний стан додатку, і ArrayList даних (dataArrayList), які будуть відображені в списку.

У методі getView() відбувається заповнення елементів інтерфейсу (ImageView та TextView) для кожного елемента списку. Він отримує значення з ListData для поточної позиції, заповнює зображення та текст елемента списку з цими значеннями і повертає вигляд (View) для відображення.

Цей ListAdapter може бути використаний для налаштування відображення списку (ListView) з даними з ListData відповідно до визначеного макету (R.layout.list_item).

Файл lessonActivity.java (додаток Е) визначає клас lessonActivity, який є активністю (Activity) у додатку Android. Основна роль цього класу полягає в керуванні інтерфейсом користувача, пов'язаним з активністю, і взаємодії з користувачем. Цей код встановлює інтерфейс користувача, заповнює його даними і надає можливість переходу до детальної інформації про вибраний елемент списку.

Файл DetailedActivity.java(додаток Д) відображує детальну інформацію про вибраний елемент списку. Після кліку на елемент списку у lessonActivity, активність DetailedActivity запускається і відображає додаткові дані про вибраний елемент.

Основні дії, які виконуються в методі onCreate(), включають: прикріплення інтерфейсу, описаного у файлі ActivityDetailedBinding, до активності (binding = ActivityDetailedBinding.inflate(getLayoutInflater())).Встановлення вигляду активності за допомогою методу setContentView(binding.getRoot()).Отримання даних про вибраний елемент списку, переданих з попередньої активності (lessonActivity), за допомогою Intent.Встановлення отриманих даних відповідним елементам інтерфейсу (binding.detailName, binding.detailLesson, binding.detailImage).

Отже, цей код відображає детальну інформацію про вибраний елемент списку у новій активності (DetailedActivity) на основі даних, переданих з попередньої активності.

Створивши розділ «Навчання», переходимо до налаштувань розділу «Тестування». Для його інтерфейсу було створено xml-файл з назвою activity_test.xml (додаток Ж). Цей макет використовується в testActivity для відображення інтерфейсу тестування, де користувач може бачити питання. Основні елементи макету включають: TextView з ідентифікатором total_question, який відображає текст "Test" і розташовано горизонтально по центру макету. TextView з ідентифікатором question, який відображає заголовок питання і розташовано по центру макету знизу від choices_layout. LinearLayout з ідентифікатором choices_layout, який містить кнопки з варіантами відповідей (ans_A, ans_B, ans_C). Лінійний макет розташовано по центру макету. Button з ідентифікатором submit_btn, який представляє кнопку "Submit" і розташовано його знизу від choices_layout.

Після цього, було створено файл QuestionAnswer.java(додаток Ж) для створення питань, варіантів відповідей та правильних їх варіантів. У класі QuestionAnswer є було створено 3 масиви:

- question: масив рядків, який містить питання для тестування.
- choices: двовимірний масив рядків, який містить варіанти відповідей для кожного питання. Кожен рядок відповідає варіанту відповіді для певного питання.
- correctAnswers: масив рядків, який містить правильні відповіді для кожного питання.

Далі було створено файл testActivity.java (додаток Ж), для того, щоб користувач міг обирати відповідь та відстежувати свою успішність. Для цього було використано наступні методи:

- totalQuestionTextView: об'єкт TextView для відображення загальної кількості питань.
- questionTextView: об'єкт TextView для відображення поточного питання.
- A, B, C: об'єкти Button для відображення варіантів відповідей A, B і C.
- submitBtn: об'єкт Button для подання вибраної відповіді.
- score: змінна для зберігання поточного балу користувача.
- totalQuestion: змінна для зберігання загальної кількості питань.
- currentQuestionIndex: змінна для відстеження поточного індексу питання.
- selectedAnswer: змінна для зберігання вибраної користувачем відповіді.

Основні методи та функціональність включають:

- onCreate: метод, який викликається при створенні активності. Встановлює макет із activity_test.xml, знаходить віджети за їх id, налаштовує обробники подій для кнопок та завантажує перше питання.
- onClick: метод, який викликається при клацанні на кнопку або подачі відповіді. Здійснює перевірку правильності відповіді, оновлює бал та індекс питання, та завантажує наступне питання або завершує тестування.
- loadNewQuestion: метод для завантаження нового питання та відповідей на екран.

- finishQuiz: метод, який викликається після завершення всіх питань. Виводить діалогове вікно зі статусом проходження тесту та кількістю правильних відповідей. Дозволяє розпочати тестування знову.
- restartQuiz: метод для перезапуску тестування, скидає бал та індекс питання та завантажує перше питання.

Створивши форму для опитування, залишилось налаштувати інтерфейс останнього розділу — «Додатки» з корисними посиланнями. Для цього було створено файл activity_add.xml (додаток И).

3.2 Розробка системи логування та тестування

У процесі розробки програми проводилося поетапнетестування з метою виявлення програмних помилок і невідповідностей поставленого завдання. Для цього були створені емулятори смартфонів, планшетів, годинників та інших пристроїв з різними діагоналями екрану. Додаток послідовно запускався на цихемуляторах, аналізувався, і при наявності помилокбули внесені зміни в код. Далі кожна активність була піддана юніт-тестування з метоювиявлення помилок, викликаних невідповідністю очікуваних іотриманих параметрів. Для цього для кожної активності був взятий спеціальний unitclass, який посилає в активність різні вірні і хибні параметри. При помилках активності, помилка була виявлена та виправлялася. Програму було запущено на пристроях, що працюють підуправлінням різних версій Android з метою виявлення особливостейроботи додатка, запущеного в різних операційних системах. В наступному етапі, додаток тестувавсяна реальних пристроях.

Тестування не виявило жодних помилок.

Схожість

Джерела з Бібліотеки 158

1	Студентська робота	ID файлу: 1013082348	Навчальний заклад: Poltava National Technical Yuri Kon	38 Джерело	3.09%
2	Студентська робота	ID файлу: 1011569030	Навчальний заклад: Vasyl Stus Donetsk National Universi	3 Джерело	2.09%
3	Студентська робота	ID файлу: 1005778134	Навчальний заклад: Yuriy Fedkovych Chernivtsi Nationa	19 Джерело	1.53%
4	Студентська робота	ID файлу: 1004006508	Навчальний заклад: National Aviation University	28 Джерело	0.98%
5	Студентська робота	ID файлу: 1008408920	Навчальний заклад: National University of Water Manag	13 Джерело	0.96%
6	Студентська робота	ID файлу: 1005726052	Навчальний заклад: National Technical University of Ukr	17 Джерело	0.77%
7	Студентська робота	ID файлу: 1000050559	Навчальний заклад: Lviv Polytechnic National University		0.59%
8	Студентська робота	ID файлу: 1014962913	Навчальний заклад: State University Kyiv National Econ	28 Джерело	0.19%
9	Студентська робота	ID файлу: 1008299878	Навчальний заклад: Lviv Polytechnic National University	5 Джерело	0.19%
10	Студентська робота	ID файлу: 1015214787	Навчальний заклад: Vasyl Stus Donetsk National Universi	2 Джерело	0.19%
11	Студентська робота	ID файлу: 1015145008	Навчальний заклад: National Aviation University	2 Джерело	0.19%
12	Студентська робота	ID файлу: 2042794	Навчальний заклад: Lviv Polytechnic National University		0.19%
13	Студентська робота	ID файлу: 1014697548	Навчальний заклад: State University Kyiv National Economic Univ...		0.17%