МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Кваліфікаційна робота бакалавра

на тему: «Розробка мобільного застосунку «Помічник студента»

Виконав студент групи К-41
спеціальності 122 «Комп'ютерні науки»
Бахішов Яшар

Керівник ст.викл
Вохменцева Тетяна Борисівна

Консультант

Рецензент д.техн.н., професор
Мещеряков Володимир Іванович

Одеса 2021

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ООМ – об'єктно-орієнтована методологія;

UML – Unified Modeling Language – уніфікована мова моделювання) – мова графічного опису для об'єктного моделювання в області розробки програмного забезпечення, моделювання бізнес-процесів, системного проектування та відображення організаційних структур;

PHP – Hypertext Preprocessor –гіпертекстовий препроцесор), попередня назва: Personal Home Page Tools – скриптова мова програмування, була створена для генерації HTML-сторінок на стороні веб-сервера;

OCR – Optical Character Recognition;

CRC-карта – Class-responsibility-collaboration card – метод мозкового штурму, призначений для проектування об'єктно-орієнтованого програмного забезпечення;

Спринт – це фіксована ітерація роботи проекту, у кінці якого поставляється готовий функціонал, який при необхідності можна вивести на ринок;

Сайдбар (sidebar) – бічна панель, елемент, графічно відокремлений від решти;

Користувацькі історії – короткі текстові описи потрібного функціоналу програмного забезпечення;

Беклог – це журнал роботи, що залишилося зробити;

Swype – метод введення тексту, не відриваючи палець від «кнопок клавіатури» на сенсорному екрані.

ВСТУП

З розвитком технологій комп’ютерне обладнання та програмне забезпечення стали важливими інструментами не лише для науки та техніки, а й для бізнесу та навчання. У сучасному світі людині просто необхідно швидко обмінюватися повідомленнями, отримувати актуальну інформацію, бути в курсі. У наш час вирішувати цю задачу допомагає Інтернет. Однак, не завжди є можливість підійти до комп'ютера і відкрити браузер, тому все більше користувачів починають використовувати мобільні пристрої, так як вони завжди знаходяться під рукою.

Університетське середовище – це місце, де щомиті відбувається обмін величезними потоками інформації. Студенти і викладачі повинні бути в курсі розкладу, вони обмінюються інформацією про домашні завдання, консультаціях, події і т.д.

В контексті освіти мобільні пристрої пропонують різноманітні можливості для навчання, такі як портативність, соціальна інтерактивність, контекстна чутливість, зв’язаність, індивідуальність та доступність для людей, які перебувають в академічних умовах або поза навчальними закладами. Тому розробка мобільного застосування «Помічник студента» є дуже актуальним завданням, тому що дозволяє користувачу навчатися самостійно, не обмежуючись необхідністю присутністю у закладі.

Метою кваліфікаційної роботи є розробка мобільного застосунку «Помічник студента», який надає студенту доступ до свого навчального розкладу і полегшує його взаємодію з викладачами.

Для досягнення поставленої мети в роботі треба вирішити наступні завдання:

- провести аналіз предметної області;
- провести порівняльний аналіз існуючих програм-аналогів;
- обґрунтувати вибір програмних засобів розробки та технологій;

- провести проектування системи з використанням мови UML;
- виконати реалізацію мобільного застосунку;
- підготувати інструкцію користувача;
- виконати тестування мобільного застосунку.

Структура кваліфікаційної роботи бакалавра складається з вступу, чотирьох розділів, висновків, переліку посилань на 17 найменування, додатку. Повний обсяг роботи становить 60 сторінок, містить 34 рисунків і 3 таблиці.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ ПРОГРАМНИХ СИСТЕМ АНАЛОГІВ

1.1 Особливості використання мобільних технологій для навчання

Застосунки для освіти є надійним помічником у навчанні і роботі, оскільки вони не просто дають доступ до всієї необхідної в процесі навчання інформації – а пропонують різні методики її освоєння і використання. В Україні, як і в усьому світі, такі проекти стрімко набирають популярність серед людей різного віку і професій:

- учням, абітурієнтам та студентам допоможе підготуватися до іспитів;
- фахівцям – підтримувати високий кваліфікаційний рівень або отримувати новий;
- людям, які вирішили спробувати щось нове або отримати додаткову освіту, – освоїти необхідні навички і отримати нові знання;
- навчальним закладам вирішувати різні завдання навчання.

Крім стандартного набору інформації по предмету (методичні керівництва, підручники, атласи, словники), такий проект може включати аудіолекції, навчальні відео. Важливим компонентом є вправи, тести і ситуаційні задачі для оцінки засвоєння матеріалу і перевірки своїх знань. На основі статистичних даних користувач може дізнатися свій рівень володіння дисципліною (предметом).

1.2 Аналіз існуючих програмних систем аналогів

Програма Timetable дозволяє швидко та зручно скласти розклад занять. Додаток виконаний у форматі щоденника і дозволяє зберігати в пам'яті смартфона не тільки актуальний розклад пар, а й фіксувати докладні відомості про навчальні предмети і викладачів, типі занять і їх тривалості, а

також інші відомості, які можна використовувати для самоорганізації і планування справ. За допомогою Timetable (рис.1) можна додавати домашні завдання і контролювати їх виконання, заздалегідь нагадувати про майбутні заняттях і переводити телефон в беззвучний режим під час уроків. Для зручності роботи з програмою передбачені функції кольорового оформлення розкладу, засоби синхронізації даних між різними пристроями користувача і вбудовані віджети, які можна винести на головний екран Android [1].

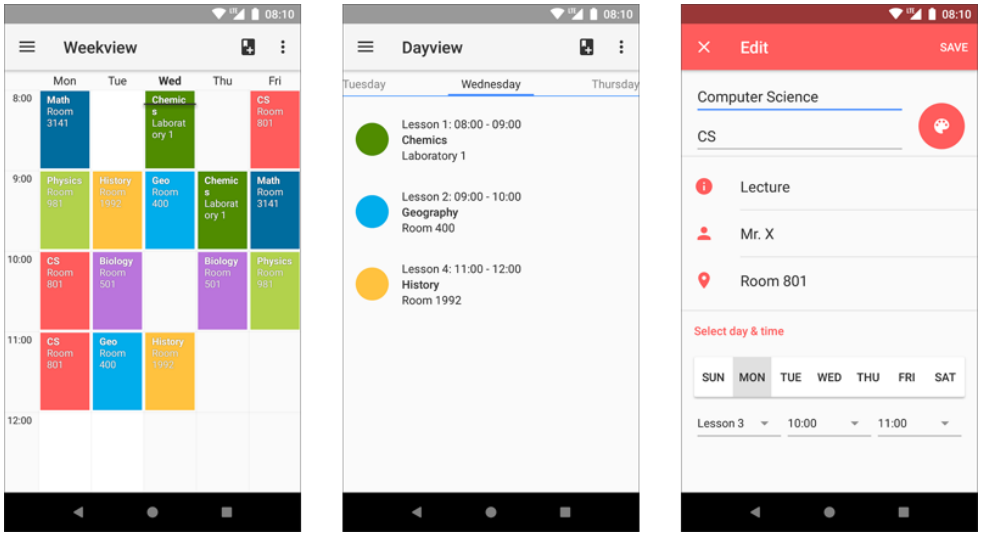


Рисунок 1 – Основні вікна програми Timetable

Photomath (Android, iOS) – додаток для вирішення математичних задач за допомогою камери смартфона і технології оптичного розпізнавання тестів Optical Character Recognition (OCR). Користуватися таким мобільним помічником дуже просто: достатньо навести камеру на задачку, і PhotoMath відразу ж видасть відповідь і призведе докладний покроковий рішення. Можливості програми дозволяють розпізнавати не тільки друкований, а й рукописний текст, а також вносити правки в відскановані формули і рівняння. PhotoMath вміє будувати графіки, вирішувати лінійні, квадратні, тригонометричні рівняння і як горіхи клацає завдання з корінням, модулями, ступенями, дробом, інтегралами, факторіалами, матрицями і полиномами [2].

Microsoft Math Solver (Android, iOS) [3]. Ще один програмний продукт, який надає допомогу у вирішенні завдань, пов'язаних з арифметикою, алгеброю, тригонометрією, обчисленням, статистикою та іншими напрямками. Як і в випадку зі згаданим вище Photomath, розроблений редмондской компанією Math Solver використовує OCR-движок для розпізнавання друкованих і рукописних текстів. При цьому рішення супроводжуються не тільки розгорнутими покроковими поясненнями, але і посиланнями на аналогічні приклади в інтернеті і відеолекціями, детально роз'яснюють пов'язані з завданням математичні поняття (рис. 2).

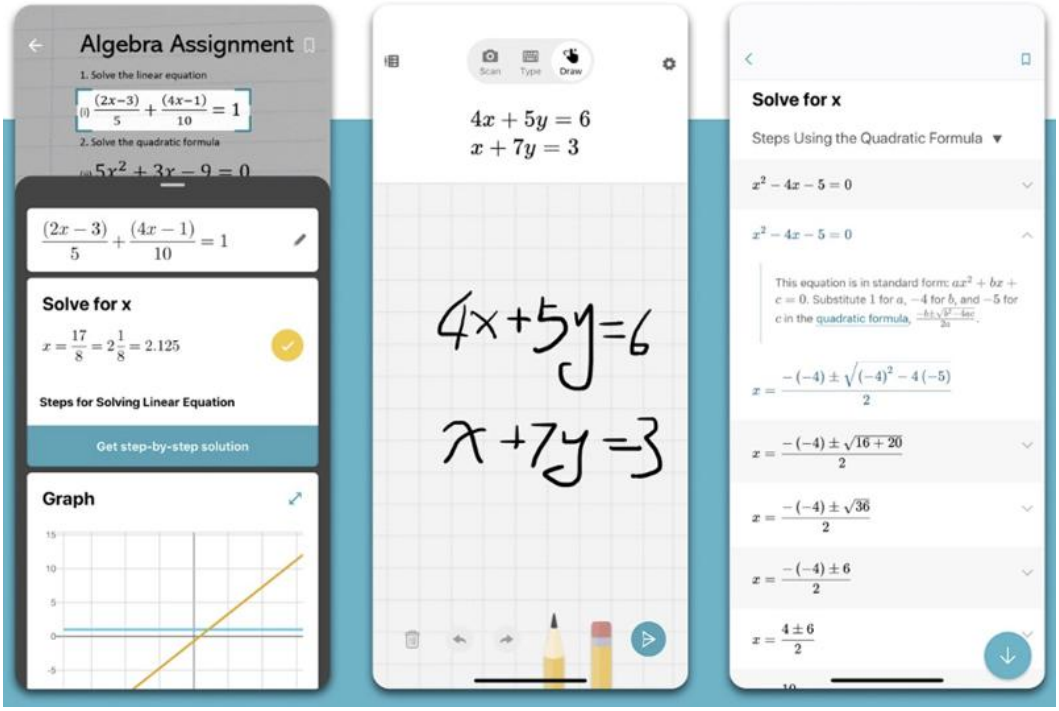


Рисунок 2 – Приклад роботи програми Microsoft Math Solver

Brainly (Android, iOS). Додаток, що функціонує в парі з платформою взаємодопомоги школярів і студентів Znanija.com. Особливістю сервісу є значна база знань, що охоплює досить широке коло освітніх напрямків: від точних і гуманітарних наук до суспільствознавства і лінгвістики. Практично на будь-яке питання в програмі Brainly (рис. 3) можна знайти докладну

відповідь, а якщо підходящого матеріалу не виявиться в архіві, завжди можна звернутися за допомогою до учасників спільноти Znanija.com [4].

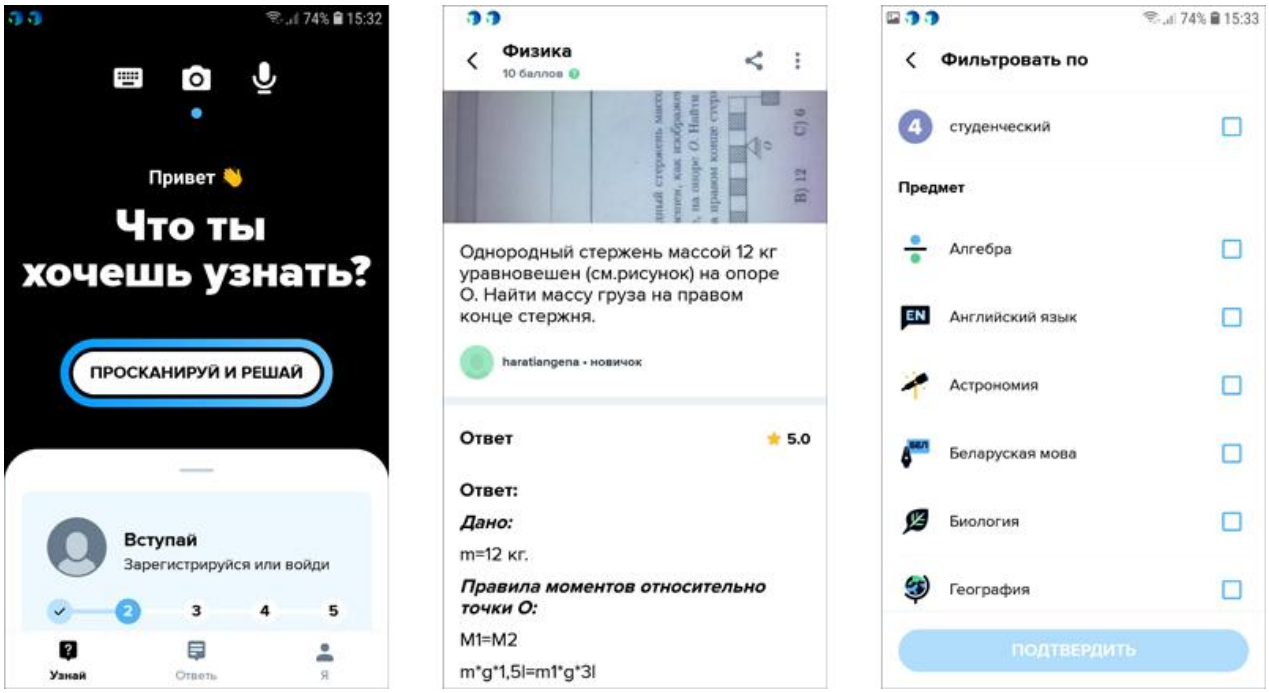


Рисунок 3 – Приклад роботи Brainly

Перераховані програмні продукти знаходять широке застосування в освітньому середовищі і приносять користь у навчані.

1.3Вибір архітектури програмної системи

Архітектурний шаблон – це узагальнене рішення поширеного завдання в архітектурі ПЗ, що часто використовується в заданому контексті. Передбачається, що мобільний за стосунок, що розроблюється буде мати клієнт-серверну архітектуру.

Згідно парадигмі клієнт-серверної архітектури один або кілька клієнтів і один або кілька серверів спільно з базовою операційною системою і середовищем взаємодії утворюють єдину систему, що забезпечує розподілені обчислення, аналіз і представлення даних (рис. 4). Функціональність

програми повинна бути розподілена між клієнтської і серверної частиною [5, 6].

Недоліком архітектури «клієнт-сервер» є те, що сервер може бути вузьким місцем щодо продуктивності, а також єдиною точкою відмови. Змінювати прийняте рішення про розміщення функціональних можливостей (на клієнті або на сервері) після створення системи – це зазвичай складно і дорого.

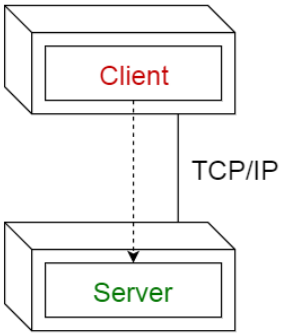


Рисунок 4 – Клієнт-серверна архітектура

На сьогоднішній день можна виділити основні (часто використовувані) підходи:

- багаторівнева архітектура;
- архітектура, керована подіями (Event-Driven Architecture);
- сервіс-орієнтована (мікросервісная) архітектура.

Багаторівнева (багатоланкова) архітектура є однією з найвідоміших архітектур, в якій кожен шар виконує певну функцію. Залежно від потреб можна реалізувати будь-яку кількість рівнів. Перевагами застосування такої архітектури є простота розробки (в основному через те, що цей вид архітектури всім знаком) і простота тестування. Серед недоліків можна виділити можливі складнощі з продуктивністю і масштабуванням – всьому виною необхідність опрацювання запитів і даних по всіх рівнях (знову ж таки, в тому випадку, якщо всі рівні є закритими) [6].

Найбільш відома структура це триланкова архітектура, що складається з клієнту, сервера додатків та серверу БД. Клієнт – це інтерфейсний компонент, додаток для кінцевого користувача. Сервер додатків розташовується на другому рівні і тут зосереджена більша частина бізнес-логіки. Сервер бази даних забезпечує зберігання даних і виноситься до третього рівеня. Зазвичай це стандартна реляційна або об'єктно-орієнтована СУБД.

Шаблон «шина подій» в основному, взаємодіє з подіями і складається з 4 основних компонентів: джерело події, прослуховувач події, канал і шина подій. Джерела розміщують повідомлення для певних каналів на шині подій. Прослуховувачі підписуються на певні канали. Прослуховувачі отримують повідомлення про появу повідомлень, розміщених на каналах з їх підписки. Шаблон часто використовується для розробки на Android, наприклад, при створенні сервісу повідомлень. Недоліком архітектури керованої подіями є низька продуктивність і відновлення після помилок [6].

Шаблон на основі мікросервісов дозволяє створювати додатки у вигляді наборів сервісів: кожен сервіс розгортається і масштабується незалежно і має власну «границю», яка обслуговується за допомогою API. Архітектуру на основі мікросервісов використовують для розгортання серверних корпоративних додатків, що підтримують різні браузери і нативні мобільні клієнти [6].

Недоліки: системи повинні бути спроектовані так, щоб витримувати збої в роботі сервісів, а це вимагає більш ретельного моніторингу систем. Додаткові витрати ресурсів на організацію роботи сервісів і взаємодії подій. Крім того, буде потрібно більше пам'яті.

Таким чином, на підставі проведеного аналізу для реалізації мобільного додатку була обрана триланкова клієнт-серверна архітектура: клієнт – сервер – сервер БД. Клієнтом є android-додаток. Це одна з розповсюджених нескладних архітектур, яка найбільш підходіть для затребуваного функціоналу додатку, що розроблюється.

1.4 Вибір технологій та засобів розробки клієнтської частини

1.4.1 Опис операційної системи Android

Існує кілька найпоширеніших операційних систем для смартфонів, такі як, IOS, Android, WindowsPhone, BlackBerry, Simbian.

Для створення клієнтської частини мобільного додатку була обрана операційна система Android, тому що:

Android – операційна система з відкритим вихідним кодом;

поширеність ОС Android (рис. 5);

доступ до розробки будь-якому користувачеві;

абсолютно безкоштовна для розробки.

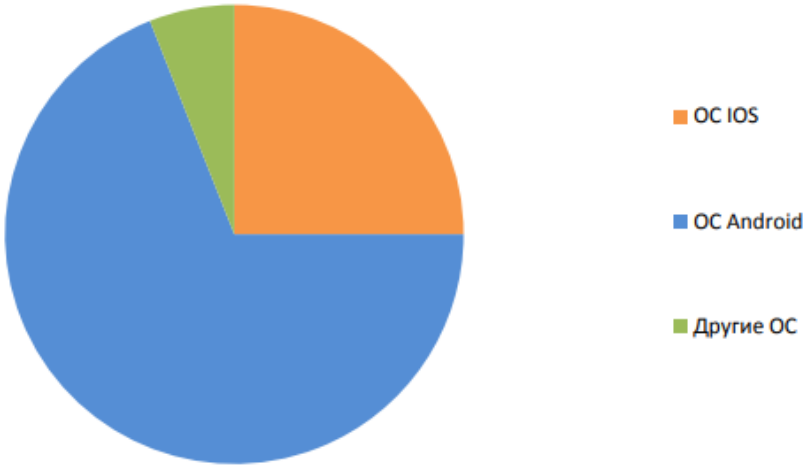


Рисунок 5 – Частка пристроїв на різних ОС

Android – операційна система для смартфонів, планшетних комп'ютерів, електронних книг, цифрових програвачів, "розумних" наручних годинників, ігрових приставок, нетбуків, смартбуків, окулярів Google, телевізорів, систем автоматичного керування автомобілем та інших пристроїв. ОС заснована на ядрі Linux і власної реалізації віртуальної машини Java від Google. Спочатку розроблялася компанією AndroidInc., Яку в 2005 році купила Google. Згодом Google ініціювала створення альянсу

OpenHandsetAlliance (OHA), який зараз займається підтримкою і подальшим розвитком платформи. Android дозволяє створювати Java-додатка, що керують пристроєм через розроблені Google бібліотеки. AndroidNativeDevelopmentKit дозволяє перенести (але не налагоджувати) бібліотеки і компоненти додатків, написані на Сі та інших мовах. ОС Android встановлена на 86% смартфонів [7].

Android є найпоширенішою операційною системою (ОС) для мобільних пристроїв – телефонів і планшетів. Дана система має безліч характерних рис, які роблять її популярною і привабливою для великої кількості користувачів по всьому світу.

Операційна система Android є невимогливою і здатна працювати на різних конфігураціях. Саме тому більшість світових виробників оснащують свої пристрої даною ОС, оскільки інші програмні продукти призначені для окремих апаратів, що відповідають певній специфікації. Така гнучкість Android пов'язана з тим, що система побудована на ядрі Linux, що має відкритий програмний код, що дає необмежені можливості розробникам. Android може бути запущений на пристроях, що мають об'єм оперативної пам'яті менше 256 Мб. Найбільш нові версії системи вимагають 512 Мб оперативної пам'яті, що також є невеликим значенням для сучасних апаратів.

Система не вимагає наявності високопродуктивного процесора і може працювати на пристроях, оснащених ядром з частотою 600 МГц. Операційна система дає можливість установки додатків з офіційного репозиторію Google, який надає найбільшу в світі базу програм. Це пов'язано з тим, що кожен розробник може самостійно написати будь-яку програму для апарату і розмістити її в магазині. Можливість також реалізована завдяки відкритості операційної системи. Варто відзначити, що додатки на пристрої під управлінням Android можуть бути встановлені як безпосередньо з телефону або планшета, так і через комп'ютер шляхом завантаження файлу .apk і його подальшої установки на апараті [8, 9].

Відмінною особливістю Android є його інтегрованість з сервісами Google – Gmail, Hangouts, Voice Search тошо.

На Android офіційно реалізована підтримка Chrome, що дозволяє синхронізувати вкладки, що відкриваються в браузері на смартфоні з комп'ютерним браузером.

Наприклад, ви можете почати перегляд сторінок з вашого телефону і при бажанні продовжити вивчати інформацію, відкривши цю ж вкладку на комп'ютері, не вдаючись до допомоги повторного пошуку.

«Андроїд» має досить простий і інтуїтивно зрозумілий інтерфейс. Всі потрібні програми розміщуються одночасно на головному екрані і в меню апарату, яке викликається натисканням на центральну сенсорну клавішу або відповідну кнопку на екрані. Всі настройки розташовуються в секції «Налаштування», а кожна дія користувача пояснюється коментарями і підказками при першому запуску апарату.

Операційна система швидко реагує на натискання користувача і виробляє установку і скачування потрібних програм і файлів зі швидкістю, яка не програє іншим сучасним мобільним ОС.

1.4.2 Загальна схема роботи програми Android

Додатки для Android в своїй роботі використовує вікна (аналогічно Windows), проте в даній системі вищевказані вікна носять іншу назву – Activity. Як і в Windows, кожне вікно має свій життєвий цикл і свої особливості (рис. 6).

При створенні нового вікна викликається метод onCreate(), при розробці даний метод перевизначається і в ньому відбувається ініціалізація програми та його компонентів.

Далі викликаються методи onStart() і onResume(). Обидва методи викликаються перед відображенням вікна при його створенні, або

відновленні (при перемиканні з іншої програми, при розгортанні згорнутого додатку тощо).

При згортанні викликаються методи onPause() і onStop().

При закритті програми і вікна викликається onDestory(), в даному методі можна зберегти призначені для користувача дані і параметри. Повний опис та послідовність виклику методів можна знайти на офіційному сайті [10].

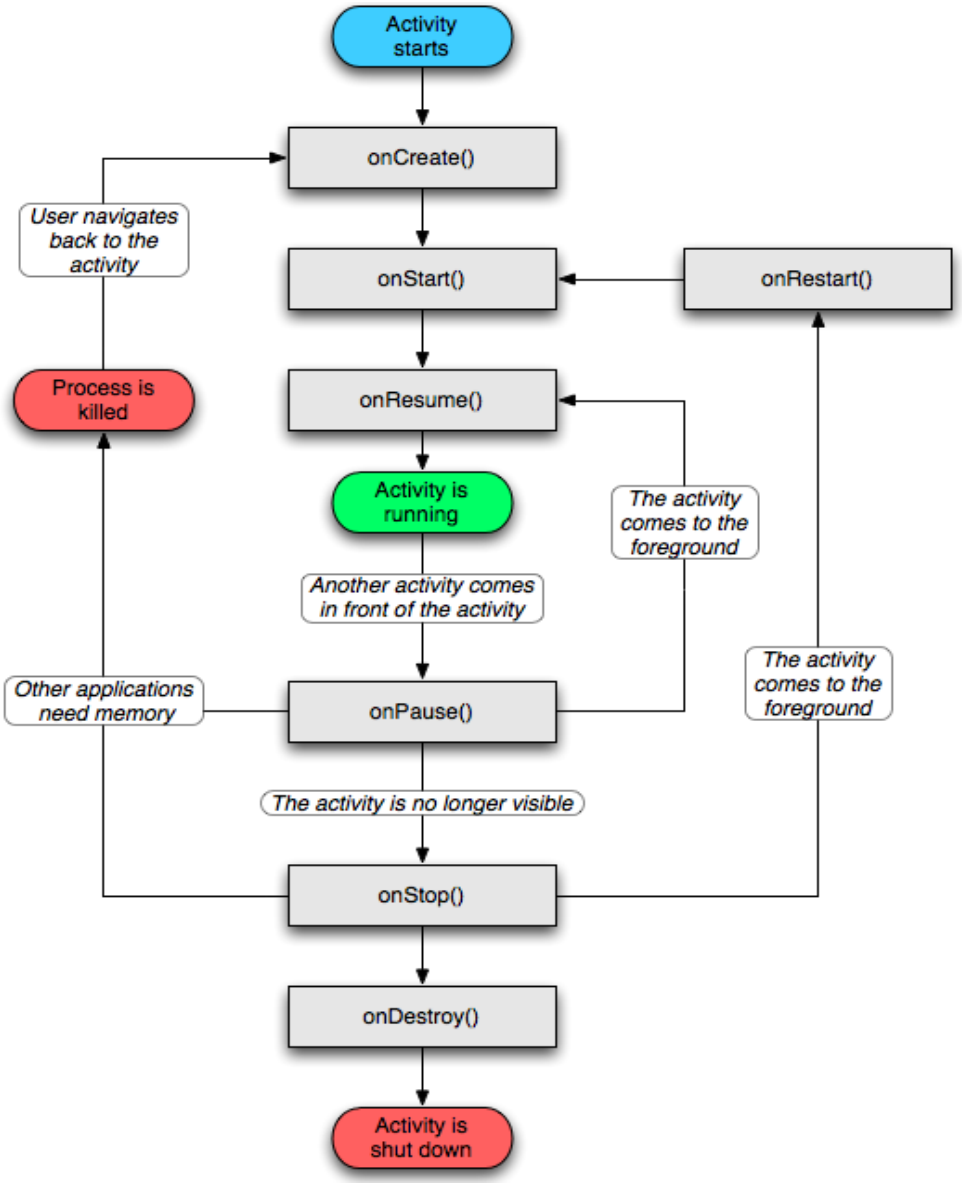


Рисунок 6 – Життєвий цикл додатки для системи під управлінням Android

1.5 Вибір мови програмування та середовища розробки

Стисло розглянемо популярні мови програмування та обґрунтуємо вибір засобу розробки для мобільного застосунку за набором критеріїв, що наведені в табл. 1.

Swift – це нова мова програмування, яка містить в синтаксисі краще від мови Сі. Створювалася та розроблялася виключно компанією Apple для розробки програм під iOS та macOS [11].

Переваги Swift: наявність власної стандартної бібліотеки та шаблонів для розробки; висока швидкість роботи та компіляції кодів; можливість безкоштовного використання. Недоліки Swift:є новим засобом розробки програм, тому існує можливість появи задач, що важко вирішуються в процесі роботи; розроблена виключно для платформи Apple (iOS і macOS).

PHP – мова програмування, що розроблена для генерування і роботи з HTML-сторінками і базами даних на веб-сервері. Є одним з лідерів серед мов, які використовуються для створення динамічних веб-сайтів [12].

Переваги PHP: розповсюджується під ліцензією (PHP license), є вільним ПЗ; підтримка підключення БД; наявність власних бібліотек та розширень; наявність великого вибору навчальних матеріалів; сумісність роботи з великою кількістю платформ та операційних систем: є самою популярною мовою програмування для створення веб-застосувань. Недоліки PHP: орієнтована в основному на веб-розробку; часті можливі протиріччя в коді (код містить в собі декілька мов, в чому і полягає складність); можливі проблеми з безпекою систем, що розроблюється.Мова спрямована на створення як веб-разробок, так і для написання різних програмних модулей та систем [13].

Python – високорівнева мова програмування загального призначення, орієнтована на підвищення продуктивності розробника та чисельність кода. Синтаксис Python не відрізняється громіздкістю та складністю читання. В

той же час стандартна бібліотека включає великий обсяг корисних функцій [14].

Переваги Python: наявність і доступність навчальної інформації; спрямованість на створення як веб-розробок, так і різного рода та складності інформаційних систем; наявність стандартної бібліотеки з необхідним функціоналом; можливість розробок для різних цільових платформ. Недоліки Python: у зв'язку зі складністю функціонала мови, її високим рівнем, швидкість проектів, що розроблюється, може виявитися невисокою відносно інших засобів програмування; наявність платної ліцензії Python Software Foundation License, можливість придбання якої може бути не у кожного бажаючого користувача.

Мова програмування C++ – типізована мова програмування загального призначення, що компілюється статично. Підтримує різні парадигми програмування, але в порівнянні з Сі, найбільша увага приділена підтримці об'єктно-орієнтованого та узагальненого програмування.

Переваги C++: підтримка різних стилів програмування; розробка програм та інформаційних систем для різного виду платформ; висока продуктивність, що пояснюється створенням складних програмних засобів; наявність власної бібліотеки та шаблонів; можливість відкритого доступу до програми для її використання; безліч відкритих джерел для вивчення мови. Недоліки C++: кладність вивчення; у зв'язку зі складністю мови можливе зменшення швидкості роботи компілятора і програмних продуктів.

Kotlin – це мова програмування з відкритим вихідним кодом. Вона відноситься до нових статично типізованих мов, що працює поверх JVM і розробляється компанією JetBrains [15]. Kotlin використовується для створення Android-додатків такими лідерами бізнесу як Uber, Atlassian, Pinterest, Evernote. Kotlin гарний вибір для розробки додатків на Android. Це можна побачити навіть з того факту, що Android Studio поставляється з вбудованою підтримкою Kotlin, як і з підтримкою Java.

Переваги Kotlin: відкритий вихідний код; інтуїтивно зрозумілий лаконічний синтаксис, що значно підвищує продуктивність і швидкість написання коду; гарна сумісність з Java; висока безпека за рахунок зниження кількості збоїв; простота вивчення. Недоліки Kotlin: часто виникають проблеми з компіляцією кода; мова має невелику спільноту і мало ресурсів для вивчення.

C# – мова програмування, яка поєднує об'єктно-орієнтовані і контекстно-орієнтовані концепції. Відноситься до родини мов з C-подібним синтаксисом, який близок до C++ та Java.

Переваги C#: підтримка різних парадигм програмування; наявність власної стандартної бібліотеки, яка спрощує написання коду; висока продуктивність, швидкість і безпека; безкоштовний доступ до середовища розробки програмних продуктів; безліч джерел для вивчення мови та вирішення проблем, що виникають. Недоліки C#: орієнтованість на платформу Windows; наявність безкоштовного використання лише для маленьких компаній та студентів, а придбання ліцензій обійдеться дорого.

Java – об'єктно-орієнтована мова програмування, що поєднує в собі можливості таких мов, як C, C++, і може вирішувати складні задачі в процесі роботи [16]. Java є основою розробок на платформі Android. Може працювати на рызних платформах (Windows, Unix і macOS, а также веб-браузерах).

Переваги Java: мова високого рівня з простим розумілим синтаксисом для вивчення; безліч ресурсів для вивчення; політика безпеки даних; має C-подібний синтаксис; крос-платформний, працює з різними платформами; можливість підключення баз даних; можливість доступа к безлічі бібліотек; широкий спектр варіантів використання мови: як для створення додатків та інформаційних продуктів, так і для створення складних система та програм [17]. Недоліки Java: платне використання для розробок комерційного ПЗ (безкоштовні версії старі).; низька швидкість через великого функціонала і можливостей мови; складність синтаксиса.

Результати порівняльного аналізу мов програмування представлені в табл. 1.

Таблиця 1.2 – Порівняльний аналіз сучасних мов програмування

	Swift	PHP	Python	C++	Kotlin	C#	Java
Читабельність	+	+	+	+	+	+	+
Можливість суміщення роботи на декількох цільових платформах	–	+ (веб-розробки)	+	+	+ (орієнтація на Android)	+	+ (орієнтація на Android)
Гнучкість	+	+	+/–	+	+	+	+
Продуктивність	+	+	–	–	+	+	+
Доступність джерел для вивчення мови	–	+	+	+	+/–	+	+
Безкоштовне використання	+	+	–	+	+	+/–	+/–
Власний набір можливостей	+	+	+	+	+	+	+
Суміщення роботи з файлами та БД	+	+	+	+	+	+	+
Безпека даних	+	+/–	+	+	+	+	+

Таким чином, з врахуванням створення программного продукту для ОС Android, була обрана саме мова програмування Java, яка поряд з іншими перевагами: крос-платформністю, наявності бібліотек і фреймворків, має широкі можливості для створення мобільних додатків для ОС Android.

Таким чином, дана система складається з веб-серверу, написаному на PHP з використанням паттерну MVC, та Android-клієнту, написаному на мові Java з використанням паттерну MVP.

2 АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Основні вимоги до ПЗ – набір вимог щодо властивостей, функцій та якості ПЗ, який розробляється. Вимоги повинні бути визначачені в процесі аналізу та фіксуються в специфікації вимог, діаграмах прецедентів та інших артефактах процесу аналізу та розробки вимог.

2.1 Опис функціональних вимог

Дана система призначена для спрощення взаємодії студента з викладачами і навпаки. Частково система нагадує соціальну мережу для всіх учасників навчального процесу. Крім соціального аспекту, в системі присутній доступ до особистого навчального розкладу.

Аналогів такої системи з такими функціями не так багато. Один з них – «Розклад» (рис. 7). Крім відображеня розкладу, дане ПЗ має можливість надсилати повідомлення від старост або викладачів. Але студент повинен самостійно слідкувати за розкладом та завданнями.

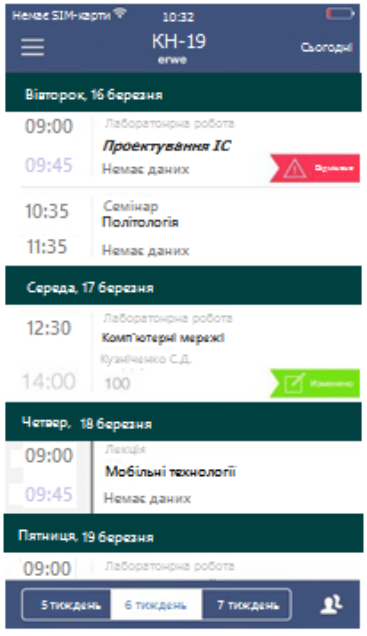


Рисунок 7 – Програма «Розклад»

Велика кількість аналогів в цій області реалізує тільки одну функцію – відображення розкладу, де розклад повинен завантажувати особисто користувач (рис. 8, 9).

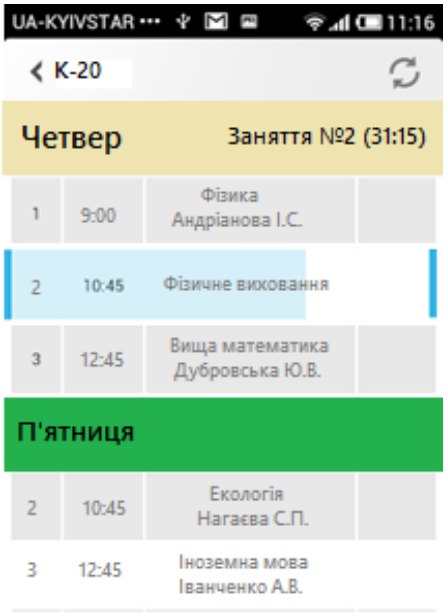


Рисунок 8 – Програма «Schoodle»

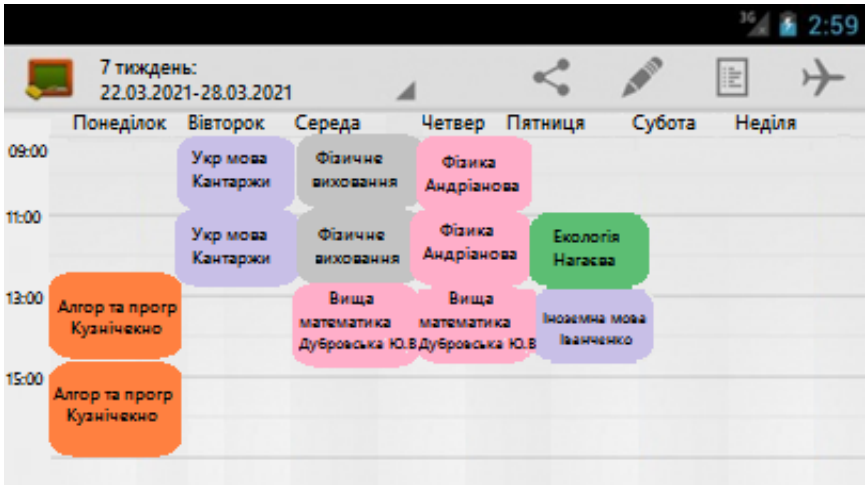


Рисунок 9 – Програма «My Class Schedule»

Розроблена система включає в себе такі функції що вже є в аналогах, так і такі, яких в них немає.

Розроблювана система має реалізувати наступні функції:

1) Відображення розкладу предметів на будь-який день.
2) Відображення списку всіх предметів з короткою інформацією про них (форма контролю, викладач, назва).
3) Відображення докладної інформації про обраний предмет.
4) Відображення і можливість скачування матеріалів для успішного виконання лабораторних, практичних і т.п. робіт.
5) Можливість входу в свій особистий кабінет і редагування інформації в ньому.
6) Можливість завантаження матеріалів, згаданих у пункті 4, викладачем.

В першу чергу були визначені основні користувачі системи. Зокрема, було виявлено три ролі: гість, «студент» і «викладач».

2.2 Формування користувацьких історій

У методології скрам для опису вимог замовника використовуються призначені для користувача історії. Так як при розробка продукта велась у рамках методології Scrum, були сформовані користувацькі історії, на основі яких і велась розробка:

a) Як гість, я можу увійти в систему під виданим мені логіном, для подальшої роботи. Пароль також видається, але може бути змінений пізніше. Приоритет: 1.

Критерії прийняття: при успішному вході користувач потрапляє на поточний розклад.

b) Як користувач, я можу переглянути свій розклад на сьогодні. Розклад зберігається в базі на сервері. Приоритет: 2.

Критерії прийняття: при натисканні на пару переходимо на сторінку з дисципліною.

c) Як користувач, я можу переглянути свій розклад на наступні дні. Приоритет: 3.

Критерії прийняття: за допомогою свайпу на екрані розкладу, вікриваються інші дні розкладу.

d) Як користувач, я можу переглянути свій список навчальних дисциплін. Приоритет: 3.

Критерії прийняття:

1) кнопка списку дисциплін знаходиться в лівому бічному меню, яке відкривається свайпом вправо.
2) в списку відображається назва дисципліни, ім'я викладача і тип дисципліни (іспит/залік).
3) після натискання на дисципліну зі списку відкрити сторінку з дисципліною.

e) Як користувач, я можу переглянути детальну інформацію про дисципліну. На сторінці з дисципліною знаходиться назва дисципліни, ім'я викладача, тип предмета (іспит/залік), кнопка відкриття вікна прикріплених матеріалів, стіна для публікації оголошень від викладача. Приоритет: 4.

Критерії прийняття: після натискання на кнопку прикріплених матеріалів відкривається вікно прикріплених матеріалів.

f) Як користувач, я можу переглянути список і/або скачати прикріплені файли на сторінці з дисципліною. Приоритет: 5.

Критерії прийняття: після натискання на кнопку завантаження файл завантажується на пристрій.

g) Як викладач, я можу завантажити прикріплені файли до дисципліни, яку веду. Кнопка завантаження файлів в систему знаходиться на сторінці прикріплених матеріалів. Приоритет: 5.

Критерії прийняття: після натискання на кнопку завантаження відкривається файловий менеджер, в якому необхідно вибрати файл завантаження.

h) Як викладач, я можу писати оголошення на стіну на сторінці з дисципліною. Текстова область для написання оголошень знаходиться відразу над стіною. Приоритет: 6.

Критерії прийняття: при відправці написаного оголошення відразу публікувати його на стіні.

і) Як викладач, я можу редагувати детальну інформацію про дисципліну, яку викладаю. Поруч з назвою дисципліни, ім'ям викладача і типом предмета знаходиться іконка олівця. Приоритет: 4.

Критерії прийняття: якщо натиснути на піктограму олівця відкривається текстове поле для редагування вибраної інформації.

j) Як користувач/викладач, я можу змінити дані свого профілю. Дані змінюються в налаштуваннях. Налаштування знаходяться в лівому бічному меню (відкривається свайпом вправо). Може бути змінений як логін, так і пароль. Приоритет: 7.

Критерії прийняття:

1) користувач повинен вводити пароль більше 6 символів.
2) користувач повинен мати унікальне ім'я в системі.
3) користувач повинен вводити логін від 6 до 30 символів.

2.3 Формування беклогу

З користувацьких історій, наведений у попередньому пункту, було сформовано беклог проекту (табл. 2).

Таблиця 1 – Беклог проекту

№ за порядком	Короткий опис історії	Пріоритет
1	Як гість, я можу увійти в систему під виданим мені логіном, для подальшої роботи.	1
2	Як користувач, я можу переглянути свій розклад на сьогодні.	2
3	Як користувач, я можу переглянути свій розклад на наступні дні.	3
4	Як користувач, я можу переглянути свій список навчальних дисциплін.	3

Продовження табл. 2

5	Як користувач, я можу переглянути детальну інформацію про дисципліну.	4
9	Як користувач, я можу переглянути список і/або скачати прикріплені файли на сторінці з дисципліною.	4
6	Як викладач, я можу завантажити прикріплені файли до дисципліни, яку веду.	5
7	Як викладач, я можу писати оголошення на стіну на сторінці з дисципліною.	5
8	Як викладач, я можу редагувати детальну інформацію про дисципліну, яку викладаю.	6
10	Як користувач/викладач, я можу змінити дані свого профілю.	7

2.4 Формування спринтів

Наведений вище беклог ділимо на два спринти за спаданням приоритетів. Перший спринт наведено у табл. 3.

Таблиця 2

№ за порядком	Короткий опис історії	Пріоритет
1	Як гість, я можу увійти в систему під виданим мені логіном, для подальшої роботи.	1
2	Як користувач, я можу переглянути свій розклад на сьогодні.	2
3	Як користувач, я можу переглянути свій розклад на наступні дні.	3
4	Як користувач, я можу переглянути свій список навчальних дисциплін.	3
5	Як користувач, я можу переглянути детальну інформацію про дисципліну.	4

Таким чином, вже за результатами першого спринта в нас є робоча демо-версія нашої системи. Але повністю продукт буде реалізований після другого спринта (табл. 4).

Таблиця 3

№ за порядком	Короткий опис історії	Пріоритет
9	Як користувач, я можу переглянути список і/або скачати прикріплені файли на сторінці з дисципліною.	4
6	Як викладач, я можу завантажити прикріплені файли до дисципліни, яку веду.	5
7	Як викладач, я можу писати оголошення на стіну на сторінці з дисципліною.	5
8	Як викладач, я можу редагувати детальну інформацію про дисципліну, яку викладаю.	6
10	Як користувач/викладач, я можу змінити дані свого профілю.	7

2.5 Розробка макетів користувацького інтерфейсу

Перше вікно, який побачить користувач при запуску програми – вікно входу в аккаунт. Його макет створений згідно користувацької історії №1 і зображений на рис. 10.

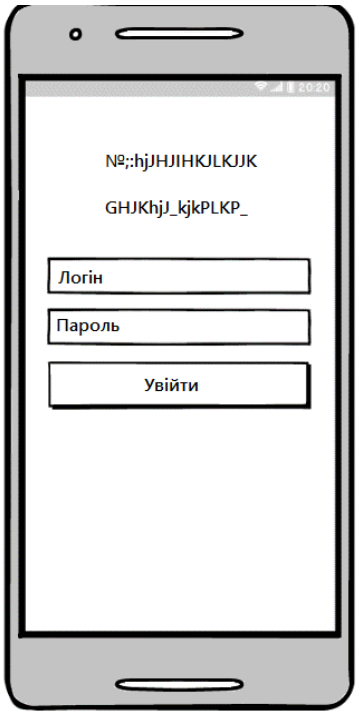


Рисунок 10 – Макет вікна входу в аккаунт

Коли користувач ввів раніше видані йому дані та нажав кнопку входу, він потрапляє на вікно розкладу його групи на сьогоднішній день. В цьому ж вікні, за допомогою свайпу, він може вибрати інший день, на який потрібно відкрити розклад. Макет цього вікна створений згідно користувацьких історій №2, №3 і зображений на рис. 11 а.

Для навигації між функціями в програмі є сайдбар, який з’являється свайпом вправо (макет на рис. 11 б).

Якщо вибрати функцію предметів, переходимо на вікно списку предметів. Макет створений згідно користувацької історії №4 і зображений на рис. 11 в.

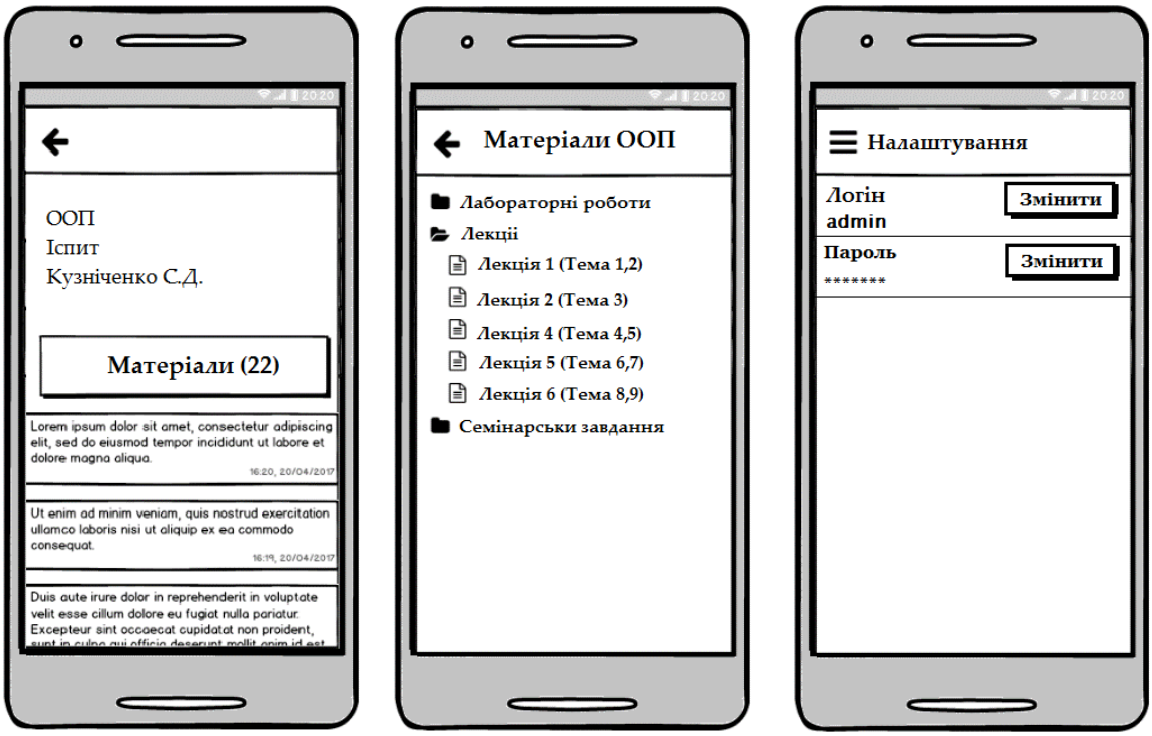
а б в

Рисунок 11 – Макети мобільного застосунку: а) вікно розкладу; б) макет сайдбару; в) вікно сторінки предмета

В цьому вікні ми можемо вибрати будь-який предмет і перейдемо на сторінку предмета. Макет створений згідно користувацьких історій №5, №7, №8 і зображений на рис. 12 а.

Якщо нажати на кнопку матеріалів, перейдемо на сторінку прикріплених матеріалів. Макет створений згідно користувацьких історій №6, №9 і зображений на рис. 12 б.

З сайдбару (рис. 11 б) можна перейти до вікна налаштувань, у якому можна редагувати свої дані. Макет створений згідно користувацької історії №10 і зображений на рис. 12 в.



а б в

Рисунок 12 – Макети мобільного застосунку: а) вікно сторінки предмета; б) вікно прикріплених матеріалів; в) вікно налаштувань

В цьому розділі було розглянуто і проаналізовано предметну область та аналоги програми. Крім цього, були створені користувацькі

історії, які відображают вимоги замовника. Всі ці користувацькі історії були сформовані в беклог, якій після цього був розділений на два спринти, для оптимізації робочого процесу. Згідно сформованих користувацьких історій були спроектовані макеті всіх вікон, що в майбутньому допоможе при розробці програми.

3 ПРОЕКТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

Архітектура програмного забезпечення – уявлення системи програмного забезпечення, яке дає інформацію про компонентах складових систему, про взаємозв'язки між цими компонентами і правила, що регламентують ці взаємозв'язки.

У найпростішому випадку, процес проектування архітектури зводиться до складання статичних і динамічних UML-діаграм.

3.1 Створення CRC-карток

CRC-карта – це метод мозкового штурму, призначений для проектування об'єктно-орієнтованого програмного забезпечення. CRC-карти використовуються, коли необхідно визначити класи та способи їх взаємодії.

Клас Route, який наведено на рис. 13, приймає запит від клієнту, оброблює його та створює потрібний контролер, потім викликає потрібний метод.

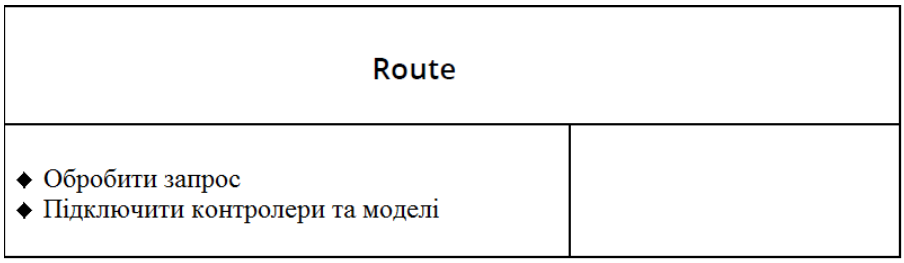


Рисунок 13 – Картка класу Route

Абстрактний клас Controller, який наведено на рис. 14, описує контролери.

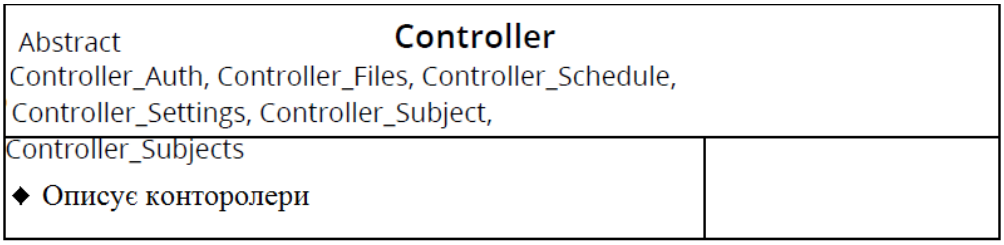


Рисунок 14 – Картка класу Controller

Згідно з історіями 6 та 9: “Як викладач, я можу завантажити прикріплені файли до дисципліни, яку веду”, “Як користувач, я можу переглянути список і/або скачати прикріплені файли на сторінці з дисципліною”, видно, що існує клас Controller_Files, який обролює запити для роботи з файлами. Цей клас наведено на рис. 15.

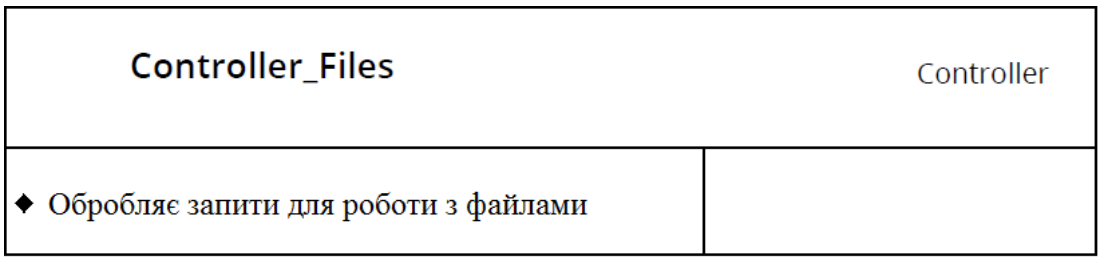


Рисунок 15 – Картка класу Controller_Files

Згідно з історіями 5, 7, 8: “Як користувач, я можу переглянути детальну інформацію про дисципліну”, “Як викладач, я можу писати оголошення на стіну на сторінці з дисципліною”, “Як викладач, я можу редагувати детальну інформацію про дисципліну, яку викладаю”, видно, що існує класс Controller_Subject, який обролює запити для роботи з предметами. Цей клас наведено на рис. 16.

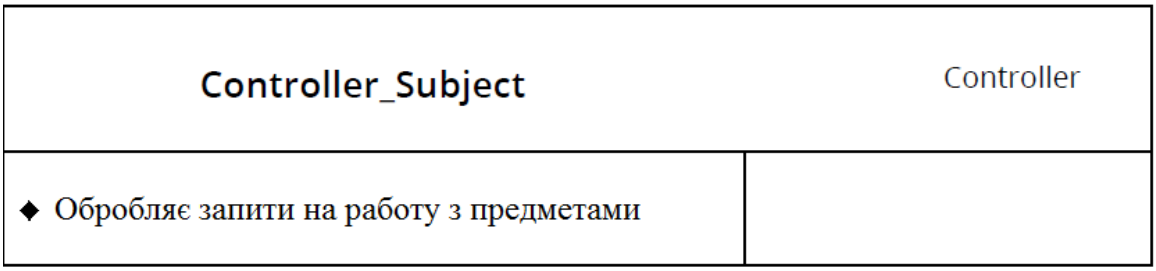


Рисунок 16 – Картка класу Controller_Subject

Абстрактний клас Model, який наведено на рис. 17, описує моделі.

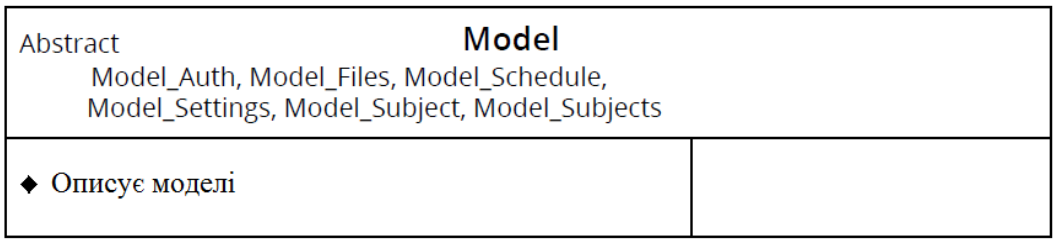


Рисунок 17 – Картка класу Model

Згідно з історіями 6 та 9: “Як викладач, я можу завантажити прикріплені файли до дисципліни, яку веду”, “Як користувач, я можу переглянути список і/або скачати прикріплені файли на сторінці з дисципліною”, видно, що існує клас Model_Files, який виконує роботу з файлами. Цей клас наведено на рисунку 18.

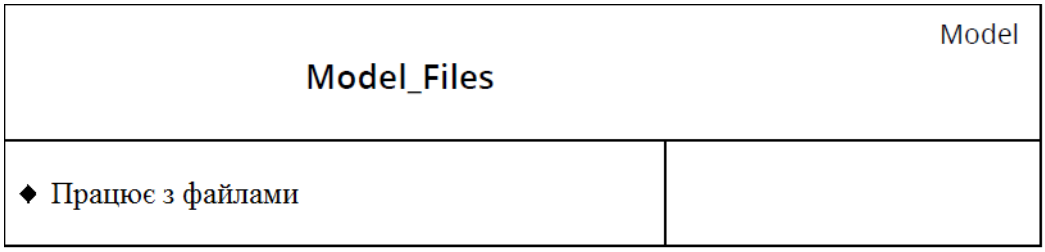


Рисунок 18 – Картка класу Model_Files

Згідно з історіями 5, 7, 8: “Як користувач, я можу переглянути детальну інформацію про дисципліну”, “Як викладач, я можу писати оголошення на стіну на сторінці з дисципліною”, “Як викладач, я можу редагувати детальну інформацію про дисципліну, яку викладаю”, видно, що шснує класс Model_Subject, який виконує роботу з предметами. Цей клас наведено на рис.у 19.

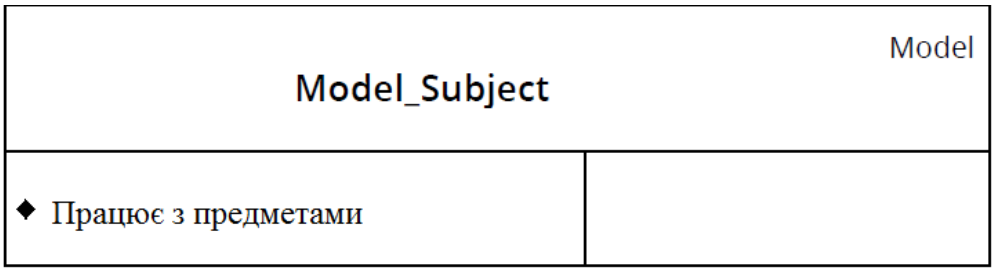


Рисунок 19 – Картка класу Model_Subject

3.2 Побудова діаграми класів

Цей тип діаграми є основним при розробці об’єктно-орієнтованої системи, так як дозволяє наочно зобразити структуру класів системи. Така діаграма корисна як при попередньому проектуванні, так і супроводжені та виправленні помилок (рис. 20).

Є класи <назва>Fragment, які виступають у ролі View. Є клас MainActivity, який зберігає усі класи <назва>Fragment. Усі класи вигляди вхаємодіють з класами подавцями <назва>Presenter.

Класи DataManager, PreferenceHelper, <назва>Service виступають у ролі моделі у паттерні MVP.

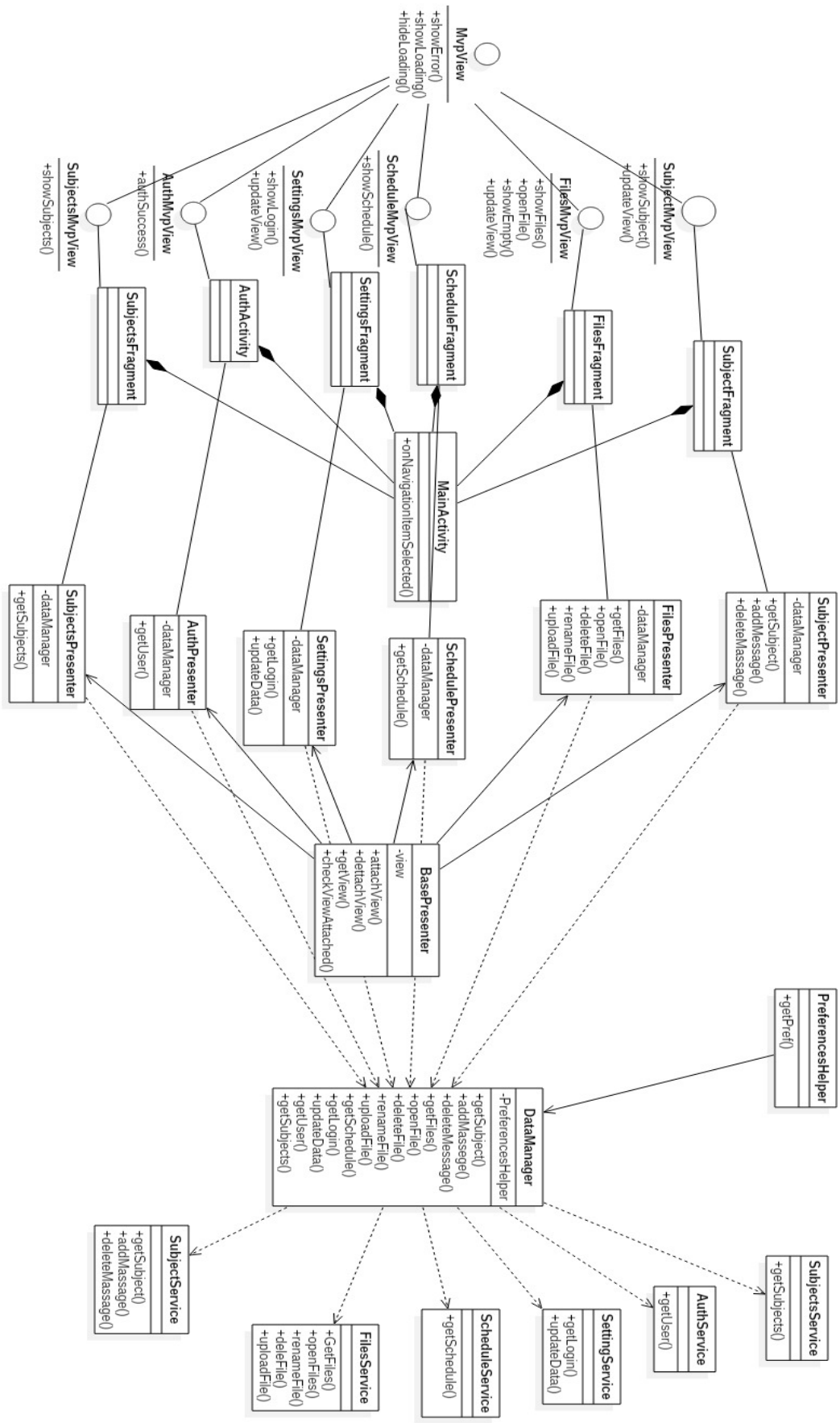


Рисунок 20 – Діаграма класів мобільного додатку

Кожний клас подавець <назва>Presenter взаємодіє з класом моделлю DataManager, викликає потрібний метод, потім клас DataManager викликає відповідний клас <назва>Service, який звертається до серверу, та отримують відповідь з серверу та передають до DataManager, який, в свою чергу, за допомогую патерну проектування Observable передає дані класу <назва>Presenter. Клас <назва>Presenter оновлює відповідний клас вигляд <назва>Fragment.

3.3 Побудова діаграми послідовностей

На діаграмі послідовностей (рис.21) для деякого набору об'єктів на єдиній тимчасової осі показаний життєвий цикл будь-якого певного об'єкта (створення-діяльність-знищення якоїсь сутності) і взаємодія акторів (дійових осіб) системи в рамках певної користувацької історії (відправка запитів і отримання відповідей).

На основі першої користувацької історії: “Як гість, я можу увійти в систему під виданим мені логіном, для подальшої роботи”. Створено діаграму послідовностей відносно гостя, яку приведено на рис. 21.

Згідно з рис. 22 можна побачити, що гість заповнює форму на екрані авторизації, ці дані відправляються до класу AuthActivity, який викликає метод getUser у класу AuthPresenter. В свою чергу, клас AuthPresenter викликає метод getUser у класу DataManager, який передає ці дані до AuthApi. AuthApi відпраляє дані на сервер. Потім сервер передає клас AuthModel у зворотньму порядку він передається до класу AuthPresenter де відбувається перевірка даних. У залежності від результатів перевірки користувачу буде показано повідомлення про помилку, або він буде перенаправлений до головної сторінки.

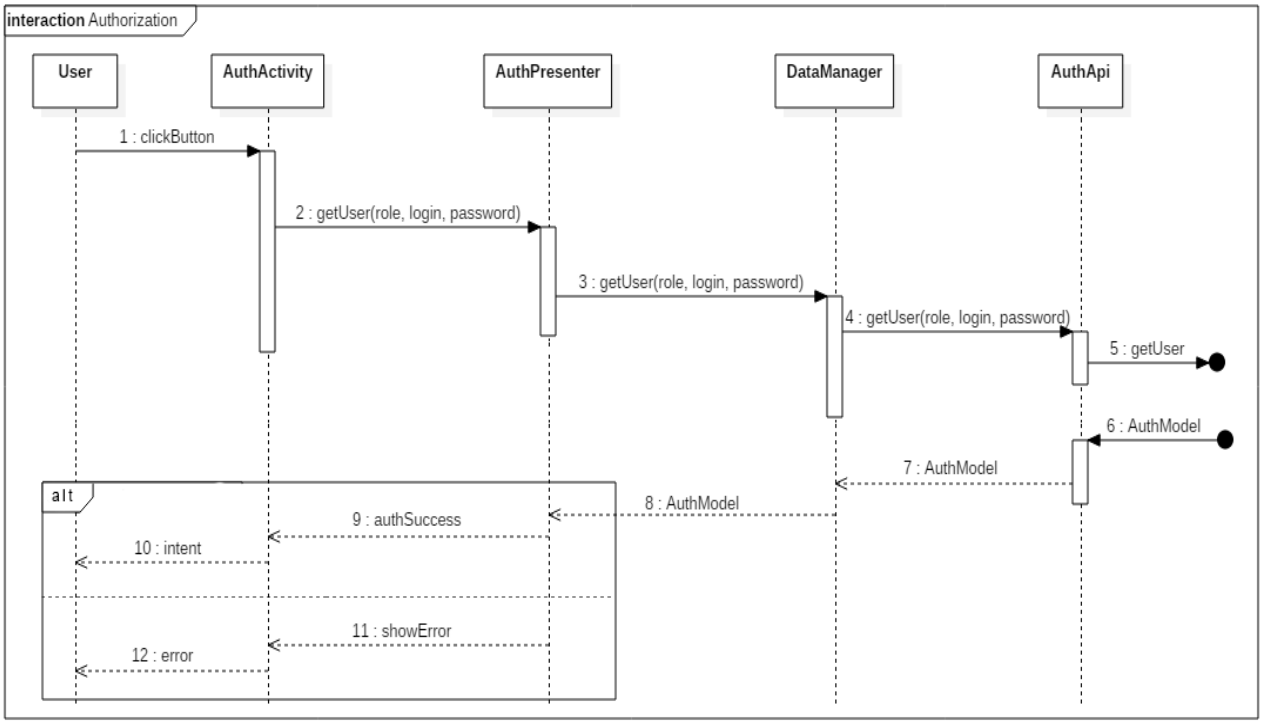


Рисунок 21 – Діаграма послідовностей користувацької історії №1

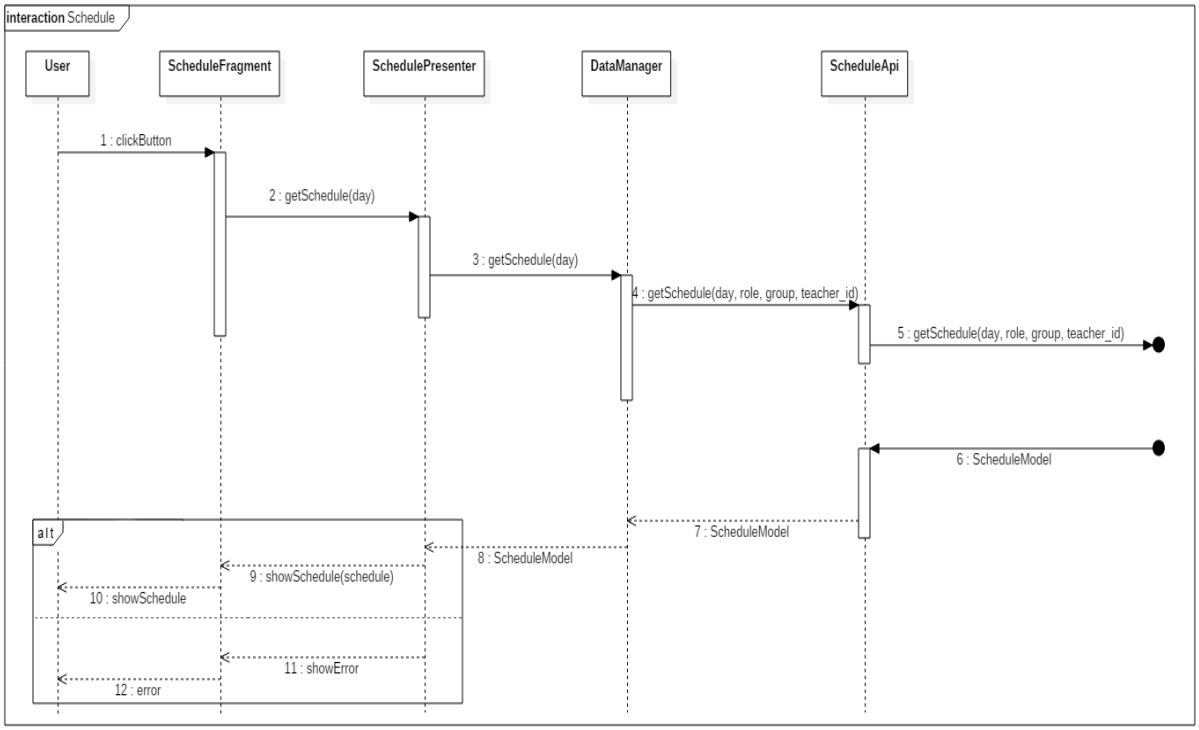


Рисунок 22 – Діаграма послідовностей користувацьких історій №2 та №3

На основі другої та третьої користувацьких історій: “Як користувач, я можу переглянути свій розклад на сьогодні” та “Як користувач, я можу переглянути свій розклад на наступні дні”. Створено діаграму послідовностей відносно Користувача, яку приведено на рис 23.

Згідно з рис. 23 можна побачити, що користувач потрапляє на сторінку розкладу(ScheduleFragment), ScheduleFragment визначає день, для якого потрібно завантажити розгляд. Ці дані передаются у клас SchedulePresenter за допомогю методу getSchedule. Далі цей метод виклакється у класі DataManager, який передає ці дані до ScheduleApi. ScheduleApi відпраляє дані на сервер. Потім сервер передає клас ScheduleModel у зворотньму порядку він передається до класу SchedulePresenter де відбувається перевірка даних. У залежності від результатів перевірки користувачу буде показано повідомлення про помилку, або роклад на потрібний день.

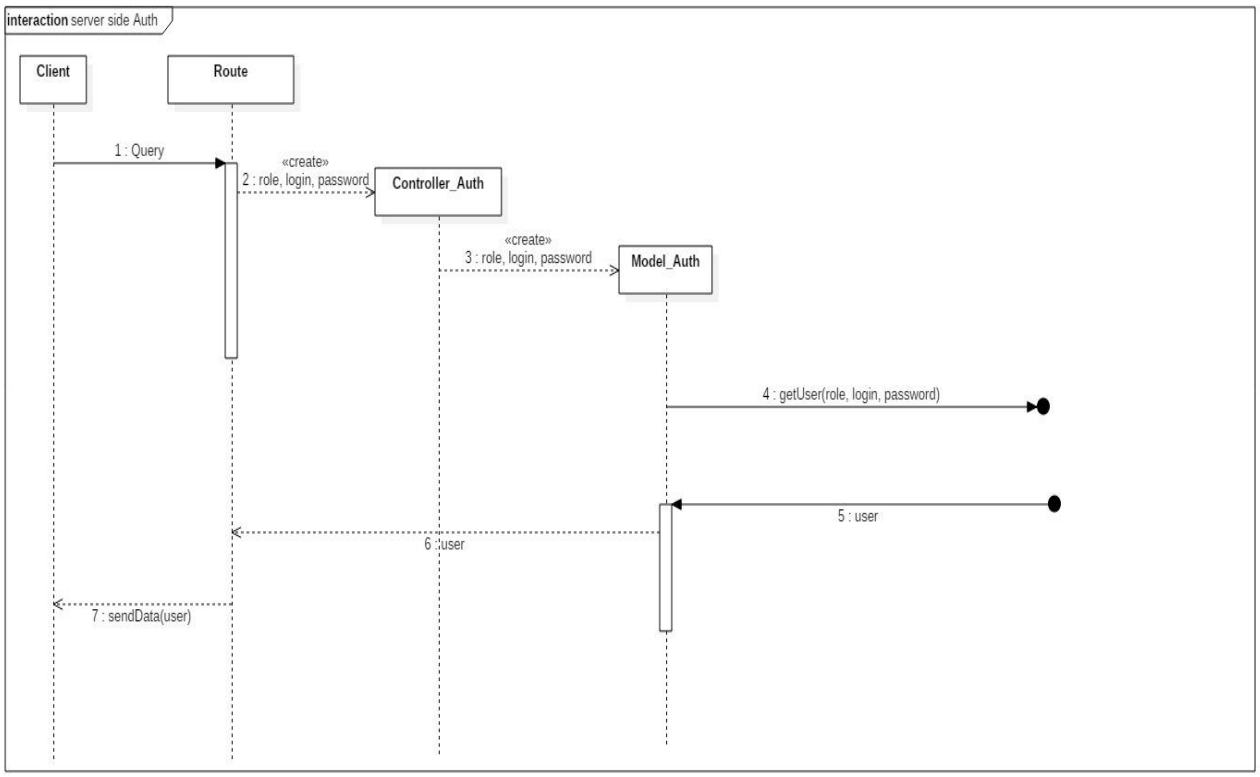


Рисунок 23 – Діаграма послідовностей користувацької історії №1

На основі першої користувацької історії: “Як гість, я можу увійти в систему під виданим мені логіном, для подальшої роботи”. Створено діаграму послідовностей відносно Гостя, яку приведено на рис. 24.

Клієнт передає до класу Route запит. Клас Route створює клас Controller_Auth та передає йому параметри. Клас Controller_Auth створює клас Model_Auth та передає йому параметри. Потім виконується запит до бази даних. Після отримання відповіді дані передаються до класу Model_Auth. Звідти вони передаются до клієнту за допомогою класу Route.

На основі другої та третьої користувацьких історій: “Як користувач, я можу переглянути свій розклад на сьогодні” та “Як користувач, я можу переглянути свій розклад на наступні дні”. Створено діаграму послідовностей відносно Клієнту, яку приведено на рис. 25.

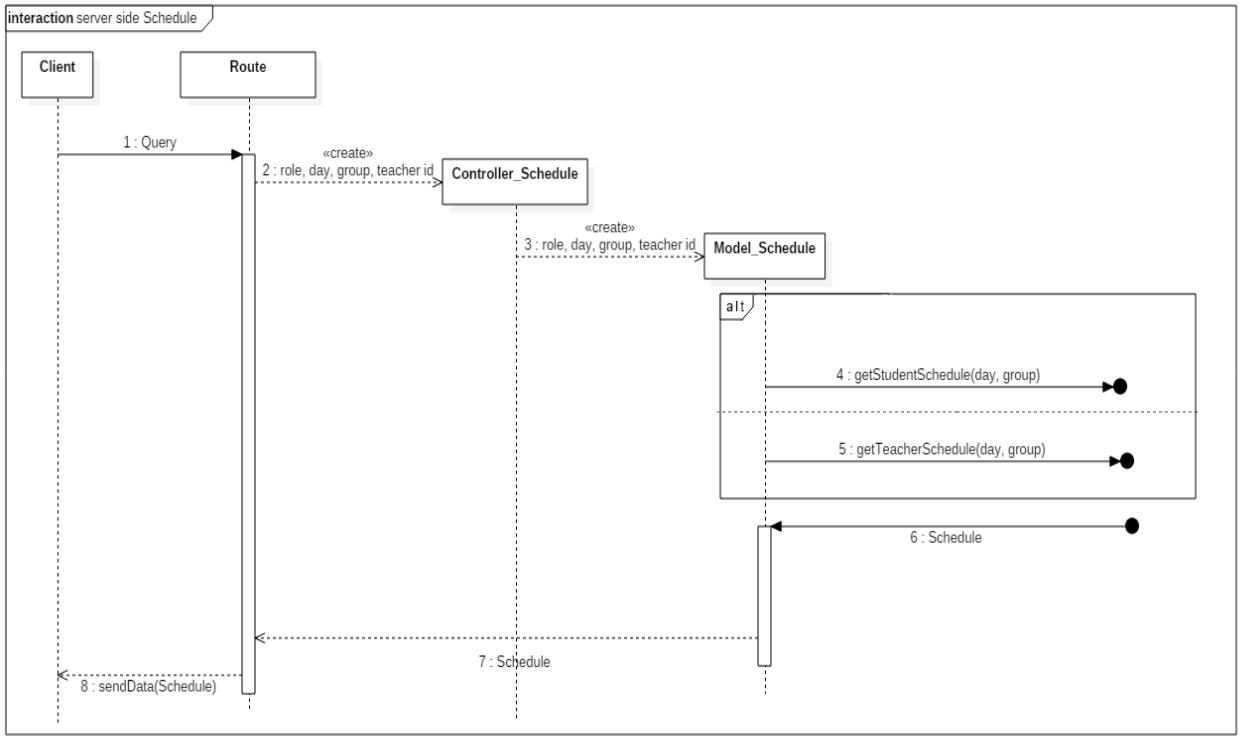


Рисунок 24 – Діаграма послідовностей користувацьких історій №2 та №3

Клієнт передає до класу Route запит. Клас Route створює клас Controller_Schedule та передає йому параметри. Клас Controller_Schedule

створює клас Model_Schedule та передає йому параметри. Потім виконується запит до бази даних, відповідно до ролі, яка була передана. Після отримання відповіді дані передаються до класу Model_ Schedule. Звідти вони передаються до клієнту за допомогою класу Route.

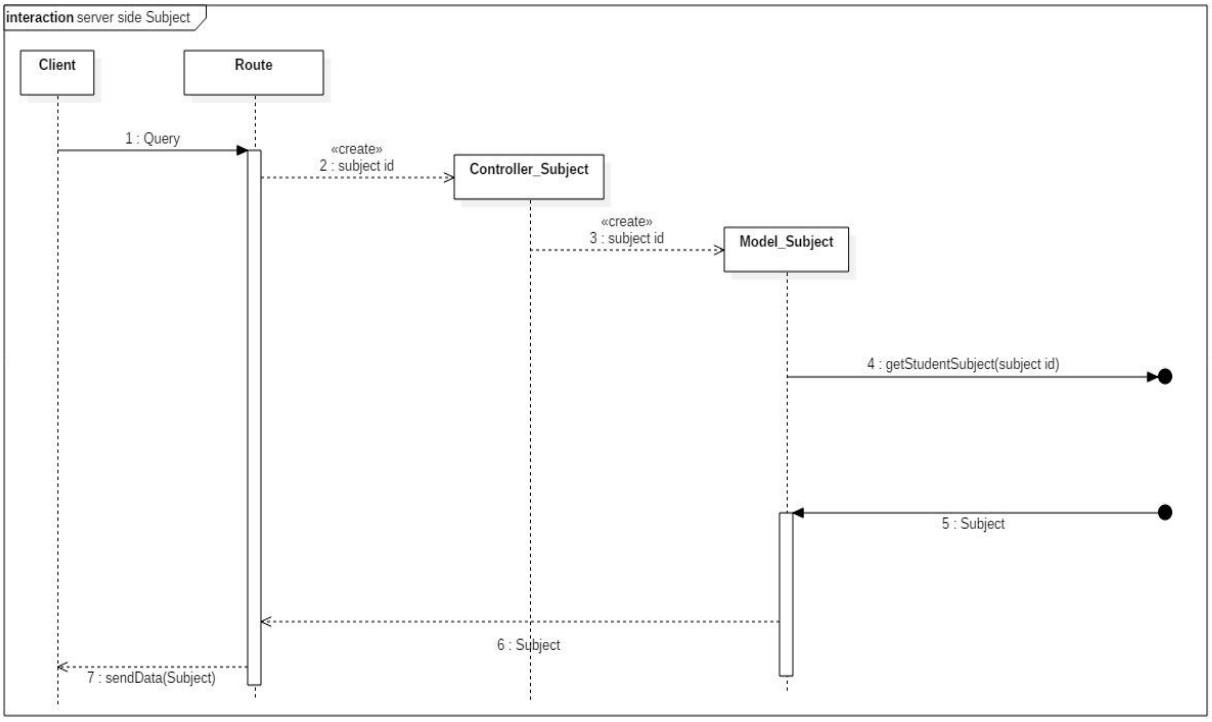


Рисунок 25 – Діаграма послідовностей користувацьких історій №5 та №7 та №8

На основі п’ятої, сьомої та восьмої користувацьких історій створено діаграму послідовностей відносно Клієнту.

Клієнт передає до класу Route запит. Клас Route створює клас Controller_Subject та передає йому параметри. Клас Controller_Subject створює клас Model_Subject та передає йому параметри. Потім виконується запит до бази даних. Після отримання відповіді дані передаються до класу Model_Subject. Звідти вони передаються до клієнту за допомогою класу Route.

У даному розділі було спроектовано систему даної курсової роботи, створено CRC-картки, а, отже виявлено основні класи програми, створено діаграму класі, тобто виділені основні відносини між класами, розроблено діаграми послідовностей для уявлення роботи програми.

Отже, система була майже повністю спроектована, що допоможе не тільки мати правильне уявлення майбутньої програми, а й розробити систему правильно, маючи на увазі всі вимоги, зв’язки, та схеми.

4 ПРОГРАМНА РЕАЛІЗАЦІЇ МОБІЛЬНОГО ЗАСТОСУНКУ «ПОМІЧНИК СТУДЕНТА»

Відкривши програму, користувач побачить вікно входу в аккаунт (рис. 26).

Рисунок 26 – Вікно входу в аккаунт

Ввівши раніше видані користувачу дані та нажав кнопку входу, він потрапляє на вікно розкладу його групи на сьогоднішній день (рис. 27).

З цього ж вікна, за допомогою свайпу, користувач може вибрати день, на який потрібно відкрити розклад. Відкриється таке ж вікно розкладу, як на рис. 27.

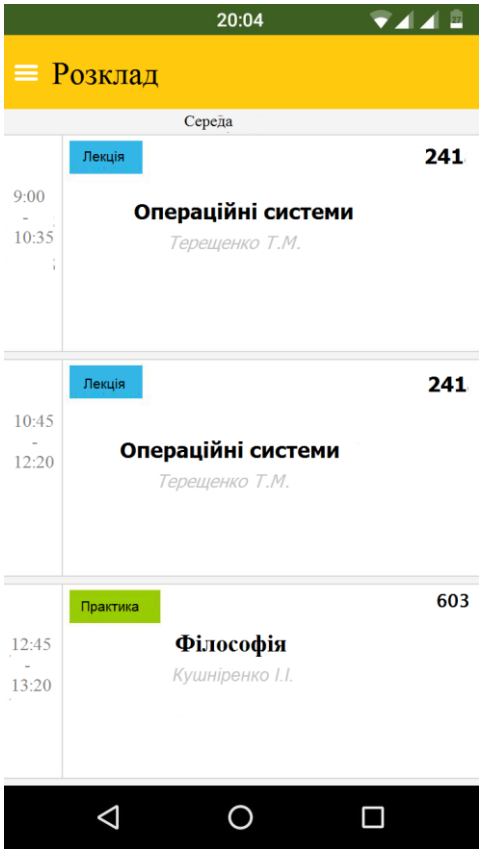


Рисунок 27 – Вікно розкладу

Для навигації між функціями в програмі є сайдбар, який з’являється свайпом вправо (рис. 28).

Якщо вибрати функцію предметів, переходимо на вікно списку предметів (рис. 29).

В цьому вікні ми можемо вибрати будь-який предмет і перейдемо на сторінку предмета (рис. 30).

Якщо нажати на кнопку матеріалів, перейдемо на сторінку прикріплених матеріалів, на якій можна завантажити необхідні матеріали (рис. 31).

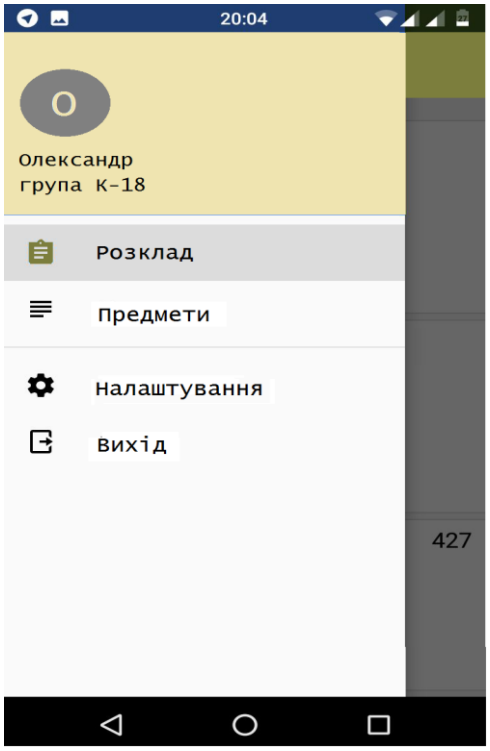


Рисунок 28 – Сайдбар

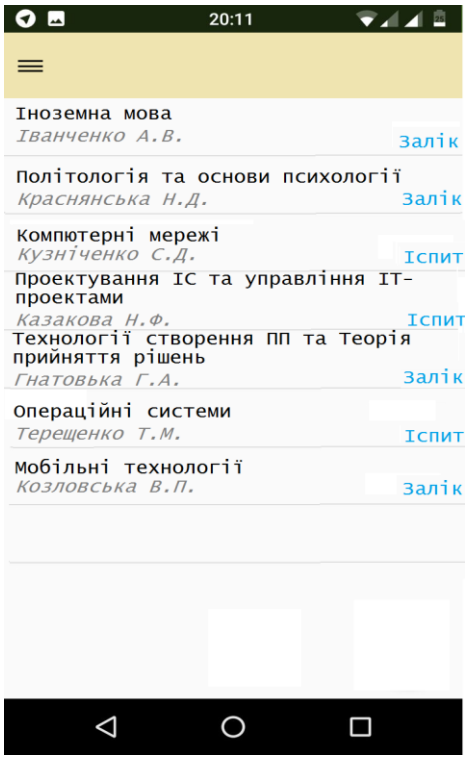


Рисунок 29 – Вікно списку предметів

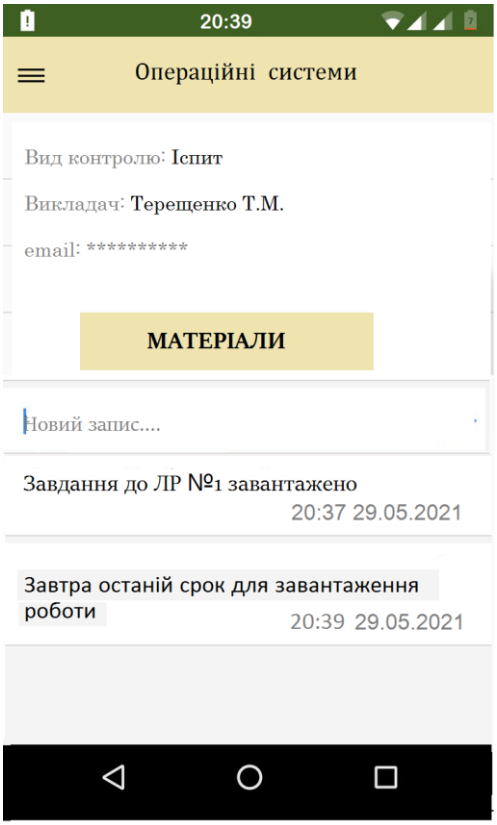


Рисунок 30 – Вікно сторінки предмета

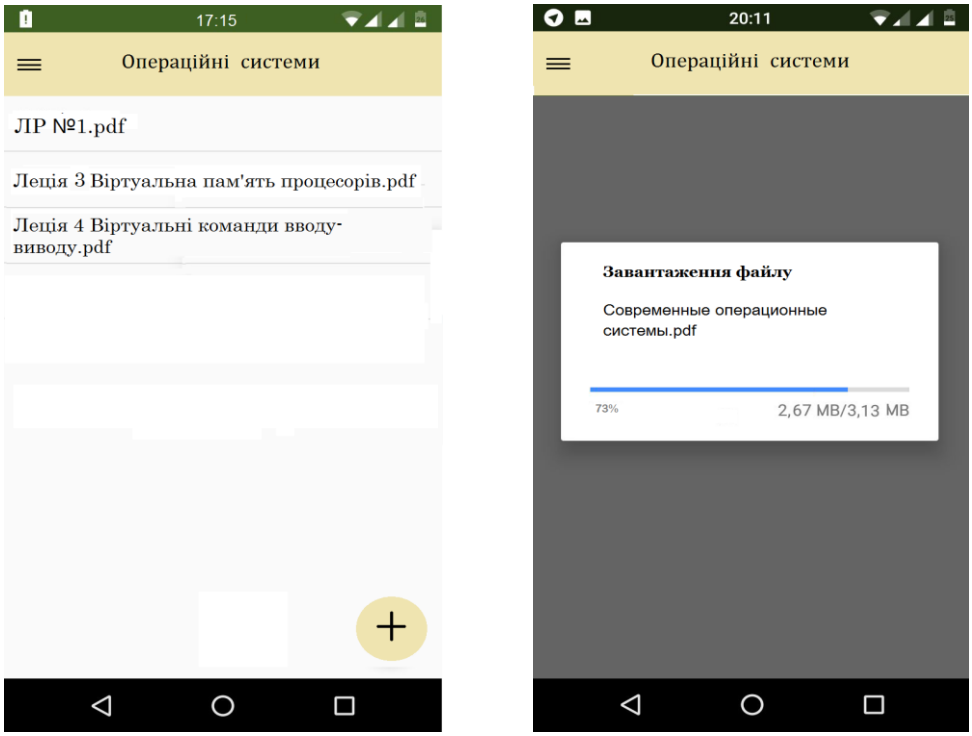


Рисунок 31 – Вікно прикріплених матеріалів

З сайдбару (рис. 28) можна перейти до вікна налаштувань, у якому можна редагувати свої дані (рис. 33).

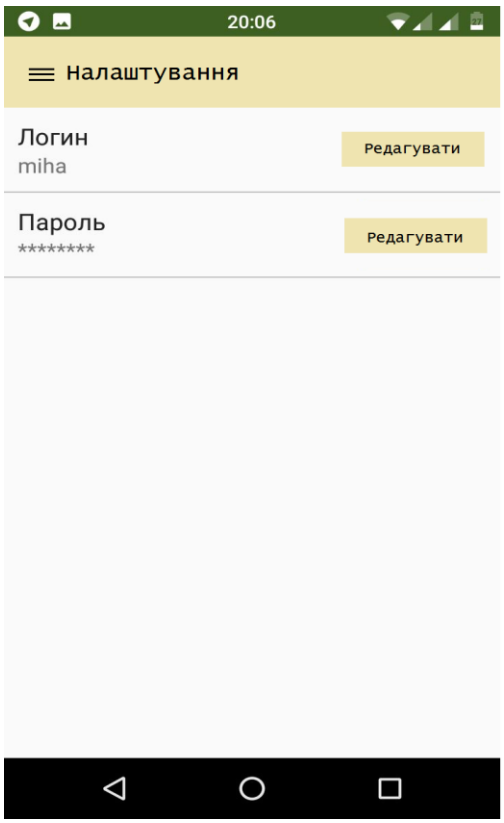


Рисунок 32 – Вікно налаштувань

Для викладача актуальна більша частина інструкції користувача-студента, окрім деяких моментів, які будут описані нижче.

На сторінці детальної інформації про предмет, викладач може опублікувати будь-яке повідомлення (рис. 33).

Крім цього, викладач має можливість редагувати назви матеріалів та видаляти матеріали свого предмету, за допомогою довгого натискання на потрібний матеріал (рис. 34).

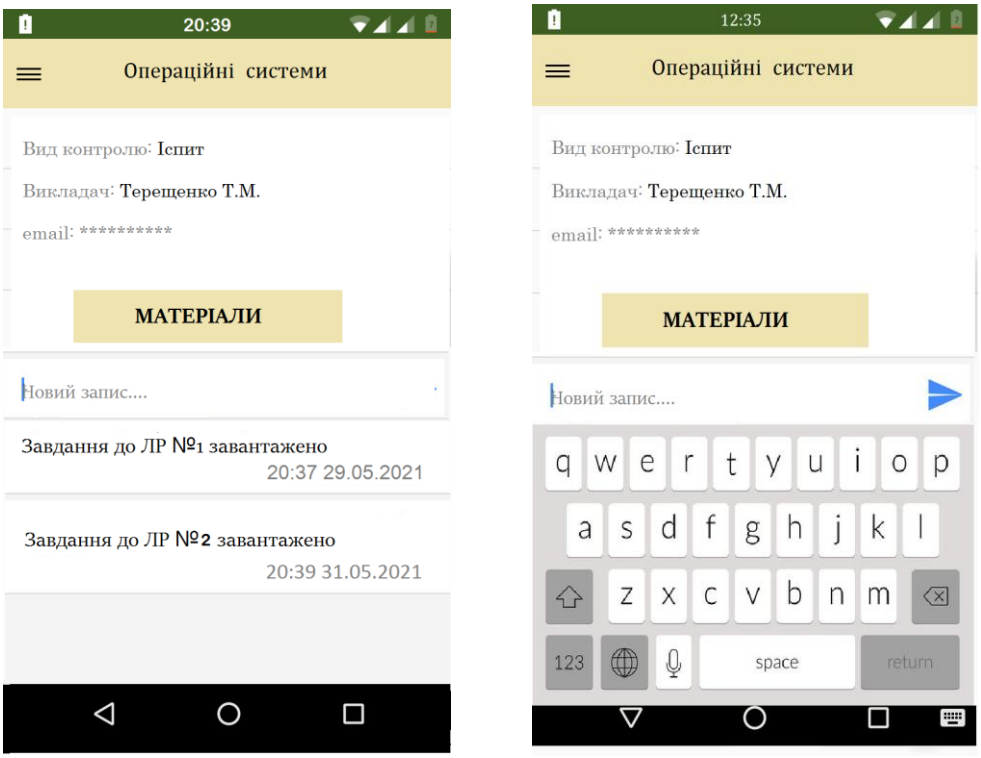


Рисунок 33 – Публікація повідомлення на сторінці предмета

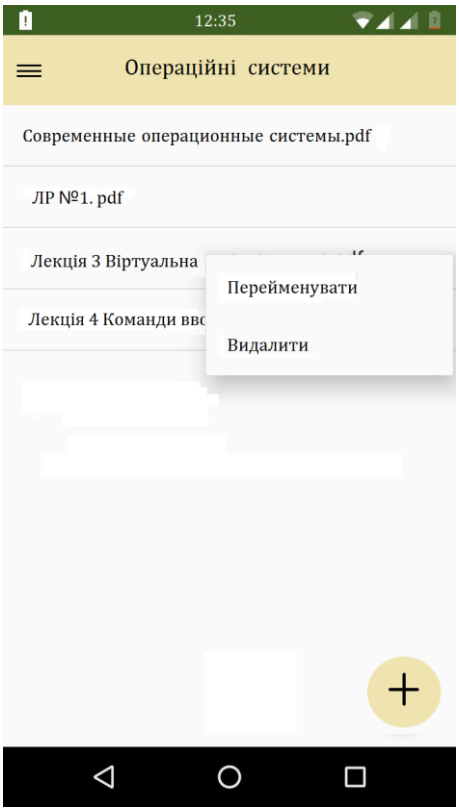


Рисунок 34 – Редагування назви чи видалення матеріалу

ВИСНОВКИ

Попит на мобільні додатки в сфері освіти підвищується. Це обумовлено тим, що такі проекти:

- економлять час і кошти на вивчення предмета (вся необхідна інформація і методики її вивчення під рукою);
- дають можливість навчатися незалежно від місця знаходження (отримувати нові знання можна в будь-який час і в будь-якому місці за допомогою мобільного телефону);
- оперативно оцінюють успіхи користувача (додатки зберігають особисту статистику, рівень засвоєння матеріалу, прогрес в навчанні).

В кваліфікаційній роботі розглянуто і проаналізовано предметну область та аналоги програми. Крім цього, були створені користувацькі історії, які відображають вимоги замовника. Всі ці користувацькі історії були сформовані в беклог, якій після цього був розділений на два спринти, для оптимізації робочого процесу. Згідно сформованих користувацьких історій були спроектовані макеті всіх вікон, що в майбутньому допоможе при розробці програми.

Система складається з веб-серверу, написаному на PHP з використанням паттерну MVC, та Android-клієнту, написаному на мові Java з використанням паттерну MVP.

На етапі проектування створено CRC-картки, і відносно виявлено основні класи програми, зроблено діаграму класів, а саме виділені основні відносини між класами, розроблено діаграми послідовностей для уявлення роботи програми.

У роботі наведено детальну інструкцію як для користувача-викладача, так і для користувача-студента. Наочність цих інструкцій забезпечується великою кількістю рисунків, які демонструют всі наявні функціональні можливості програми.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Timetable – один из лучших ежедневников студента. URL: https://android4all.ru (Дата звернення 20.04.2021)

2. Photomath – обзор, плюсы и минусы программы, отзывы. URL: https://8d9.ru › (Дата звернення 22.04.2021)

3. Microsoft Math Solver – приложение для автоматического решения математических задач URL: https://www.iguides.ru. (Дата звернення 23.04.2021)

4. Brainly – обзор, плюсы и минусы программы, отзывы. URL: https://8d9.ru/program/brainly (Дата звернення 23.04.2021)

5. Басс Лен, Клементс Пол, Кацман Рик. Архитектура программного обеспечения на практике. 2-е издание. – СПб.: Питер, 2006. 575 с.

6. Анатольев А.Г. Учебно-методический комплекс по дисциплине «Сетевые технологии». URL: http://www.4stud.info/networking/ (дата звернення 19.02.2021)

7. Коматинени С. Android 4 для профессионалов. Создание приложений для планшетних компьютеров и смартфонов: Научно-популярное издание. Санкт-Петербург: ТОВ «И.Д. Вильямс », 2012. 880 с.

8. Дон Гриффитс, Девид Гриффитс Программирование для Android. СПб.: Питер, 2018. 912 с.

9. Билл Филлипс, К. Стюарт, Кристин Марсикано Android. Программирование для профессмоналов. СПб.: Питер, 2017. 688 с.

10. Ян Клифтон Проектирование пользовательского интерфейса на Android.Санкт-Петербург: ДМК Прес, 2017. 452 с.

11. Swift – Офіційний сайт. URL: https://swift.org/ (дата звернення 19.02.2021)

12. PHP – Вікіпедія. (загол. з екрана). URL: https://uk.wikipedia.org/wiki/PHP (дата звернення 19.02.2021)

13. Кузнецов М.В, Объектно-ориентированное программирование на PHP. СПб.: «БХВ-Петербург», 2007. 608 с.

14. Python – Вікіпедія. (загол. з екрана). URL: https://uk.wikipedia.org/wiki/Python (дата звернення 19.02.2021)

15. Kotlin – Вікіпедія. (загол. з екрана). URL: https://uk.wikipedia.org/wiki/Kotlin (дата звернення 19.02.2021)

16. Java – Вікіпедія. (загол. з екрана). URL: https://uk.wikipedia.org/wiki/Java (дата звернення 19.02.2021)

17. Блинов, И.Н., Романчик, В. С. Java. Методы программирования : уч.-мет. Пособие. Минск : издательство «Четыре четверти», 2013. 896 с.