Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мережних технологій факультету інформатики

ПЛАНУВАННЯ АРХІТЕКТУРИ ТА РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ «ОРГАНАЙЗЕР ДЛЯ СТУДЕНТА»

Текстова частина до курсової роботи

за спеціальністю «Інженерія програмного забезпечення»

Керівник курсової роботи: к. ф.-м. н. Пєчкурова О. М.

_____ (*Підпис*)

“___” _____ 2024 року

Виконала студентка: Татарінова Я.В.

_____ (*Підпис*)

“___” _____ 2024 року

Київ 2024

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студентки Татарінової Яни Валеріївни факультету інформатики 3 курсу

Тема: Планування архітектури та розробка мобільного застосунку «Органайзер для студента»

ТЧ до курсової роботи:
Індивідуальне завдання

Календарний план

Зміст

Перелік умовних позначень

Вступ

Розділ 1 Аналіз існуючих методів (алгоритмів) вирішення поставленої задачі

Розділ 2 Вибір принципу дії системи

Розділ 3 Опис реалізації програмного продукту

Висновки

Перелік використаних джерел

Календарний план виконання курсової роботи

Тема: Планування архітектури та розробка мобільного застосунку «Органайзер для студента»

№	Назва етапу	Термін виконання	Примітка
1.	Отримання теми курсової роботи	25.09.2023	
2.	Пошук літератури за темою роботи	12.10.2023	
3.	Ознайомлення та аналіз відповідної літератури	20.10.2023	
4.	Пошук та ознайомлення з існуючими застосунками-аналогами	30.10.23	
5.	Написання основної частини програмного коду	12.02.2024	
6.	Виправлення недоліків, тестування коду на помилки	17.03.2024	
7.	Написання теоретичної частини	19.03.2024	
8.	Завершення написання практичної частини	15.04.2024	
9.	Внесення остаточних змін до теоретичної частини	02.05.2024	
10.	Створення презентації	04.05.2024	
11.	Захист курсової роботи	24.05.2024	

Студент Татарінова Я.В.

Керівник Пєчкурова О.М.

“_____” _____

Зміст

Анотація

Дана курсова робота присвячена розробці мобільного застосунку «Органайзер для студента». Даний застосунок спрямований на полегшення організації завдань та розкладу занять. Легкий та швидкий у використання органайзер, у вигляді мобільного застосунку, який завжди може бути під рукою, та є зручним інструментом для ефективного простежування розкладу, координації завдань студента, та планування власного академічного часу. Метою розробки даного програмного застосунку є надання студентам доступу до легкого інструменту для організації власного навчального процесу.

Вступ

У сучасному світі, враховуючи розвиток сучасних технологій та останні події, коли велика кількість студентів навчається дистанційно, мобільні застосунки для навчання є як ніколи популярними та актуальними [15].

Органайзер є чудовим інструментом для координації власного часу, прослідковування завдань. Завантаживши даний застосунок, студент зможе вдало планувати свій академічний день, не забуваючи про завдання та розклад предметів.

Мета курсової роботи – полегшити життя студентам, забезпечивши якісний та швидкий доступ до розкладу занять, та контролю виконаних і невиконаних завдань. Органайзер має необхідні для координації навчання студента функції, такі як додавання, редагування, видалення завдань та розкладу занять, а також контроль виконання та часу.

Для створення даного мобільного застосунку було використано мову програмування Dart та платформу Flutter.

Основна частина

1. Аналіз існуючих методів (алгоритмів) вирішення поставленої задачі

1.1. Дослідження та обґрунтування вибору теми

Як відомо, у зв’язку з поточною ситуацією з організацією навчання для студентів та школярів, мобільні застосунки-органайзери набули популярності, адже за допомогою них можна організувати власний час, керувати завданнями, навчальним процесом, і покращити власну успішність. Як зазначено у статті [1], завдяки ним можна забути про блокноти та завжди мати під рукою організований план академічного дня. Наша пам'ять не завжди у змозі пригадати всі заплановані предмети та завдання – тут на допомогу і приходять застосунки-органайзери. Вони стануть у пригоді для заповнення розкладу, його редагування у разі зміни, а також прослідковування виконаних і невиконаних завдань, їх редагування та видалення.

На сьогоднішній день, як і у будь-якій іншій сфері, існує багато застосунків-органайзерів для студентів, найбільше мене зацікавили дані програми:

1) Any.Do
2) To Do: Planner, Task Reminders
3) Тижневий планувальник
4) Just reminder with alarm

1.2. Аналіз існуючих аналогів

1.2.1. Any.Do

Даний мобільний застосунок більш схожий на календар, завдяки якому можна керувати своїми завданнями за допомогою подій та завдань у

календарі на одному екрані. Він може стати корисним, як для студентів, так і для звичайних людей, які хочуть організовувати свій час.

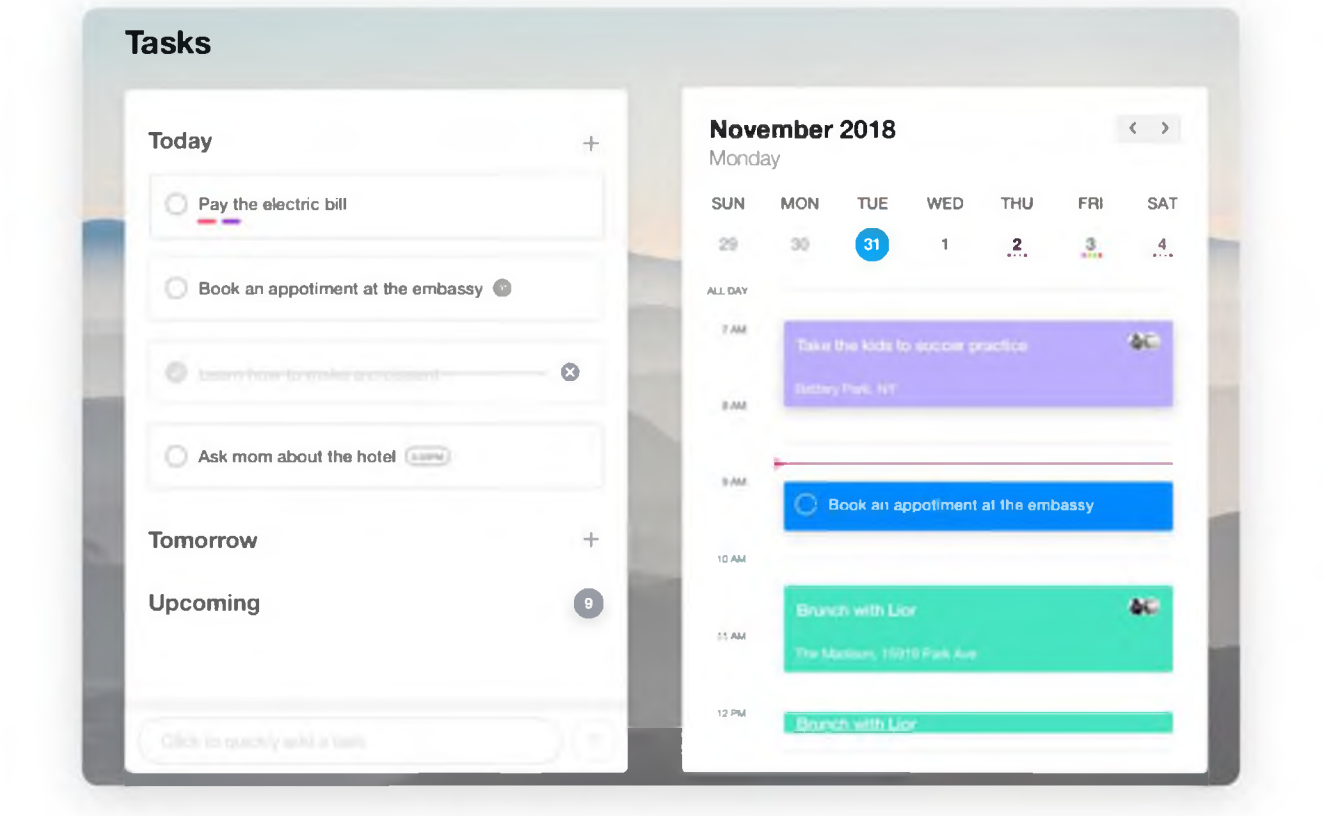

Рисунок 1.1 – Застосунок Any.Do

Функціонал застосунку Any.Do зручний та легкий у використанні. Any.do дозволяє користувачам вести проекти, створювати завдання, планувати подальші завдання у вбудованому календарі. Особливістю даного органайзера є ще те, що завдання можна переносити і в Google календар.

Проаналізувавши також відгуки користувачів, можна примітити, що недоліком є різниця функціоналу на Android та iOS. iOS та Android працюють по-різному, а саме у iOS немає налаштування лише прострочених завдань, на відмінну від Android, тобто маючи гаджети на різних операційних системах користувач вимушений адаптуватися до особливостей.

1.2.2. To Do: Planner, Task Reminders

Даний сервер менш популярний за попередній, адже поступається функціоналом.

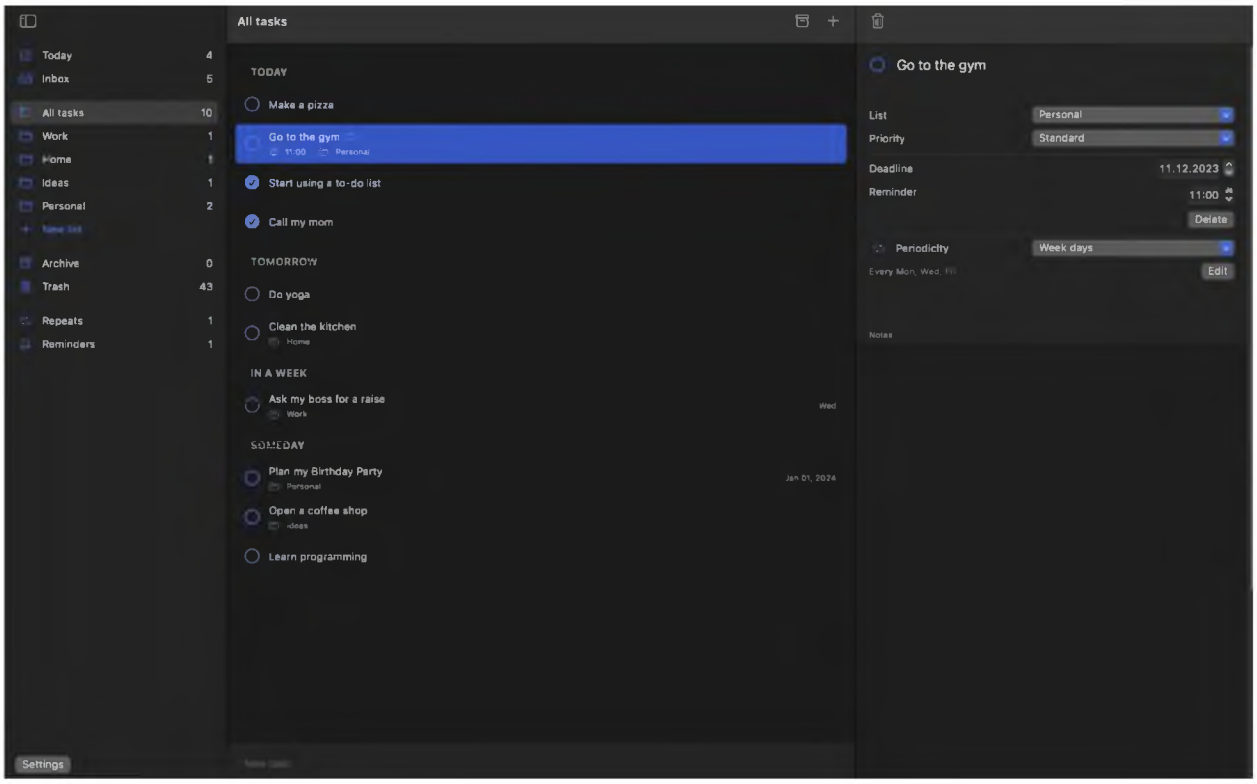

Рисунок 1.2 – Головна сторінка застосунку To Do: Planner, Task Reminders

Це органайзер завдань із нагадуваннями, повторюваними завданнями та віджетом блокування екрана. Він містить функції: додавання завдань, підзавдань, повторюваних завдань, створення папок, до який можна додавати дані завдання і сортувати їх, а також мусор (Trash), де будуть знаходитися видалені завдання.

Проаналізувавши застосунок та відгуки користувачів, можна зробити висновок що у To Do: Planner, Task Reminders не вистачає більш зручного та інтуїтивного зрозумілого новому користувачу інтерфейсу. Застосунок не буде зручним для студентів або школярів, адже не містить функції додавання розкладу.

1.2.3. Тижневий планувальник

Програму Тижневий планувальник можна використовувати як щоденник, блокнот або органайзер денних планів.

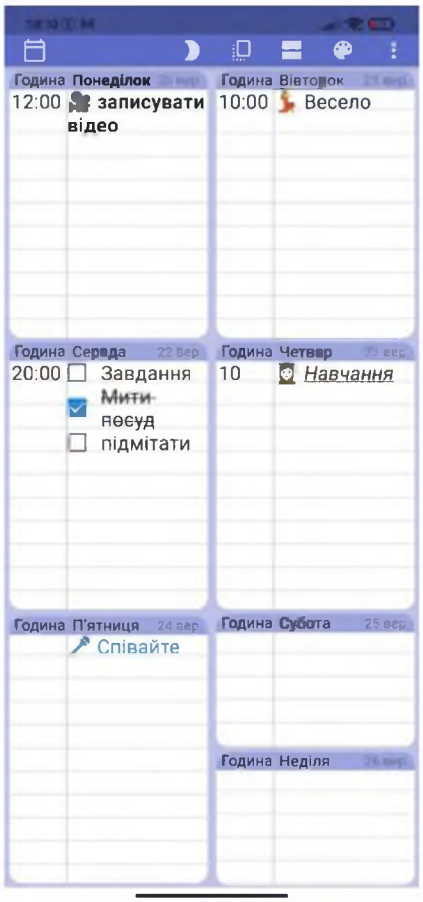

Рисунок 1.3 – Головна сторінка застосунку Тижневий планувальник

Застосунок Тижневий планувальний підійде студентам які хочуть мати так званий щоденник у телефоні, недоліком є те що програма не містить окремого календаря подій для розкладу занять і окремого для виконаних завдань.

1.2.4. Just Reminder with Alarm

Just Reminder with Alarm простий у використанні застосунок, що містить у собі нагадування про заплановані вам події.

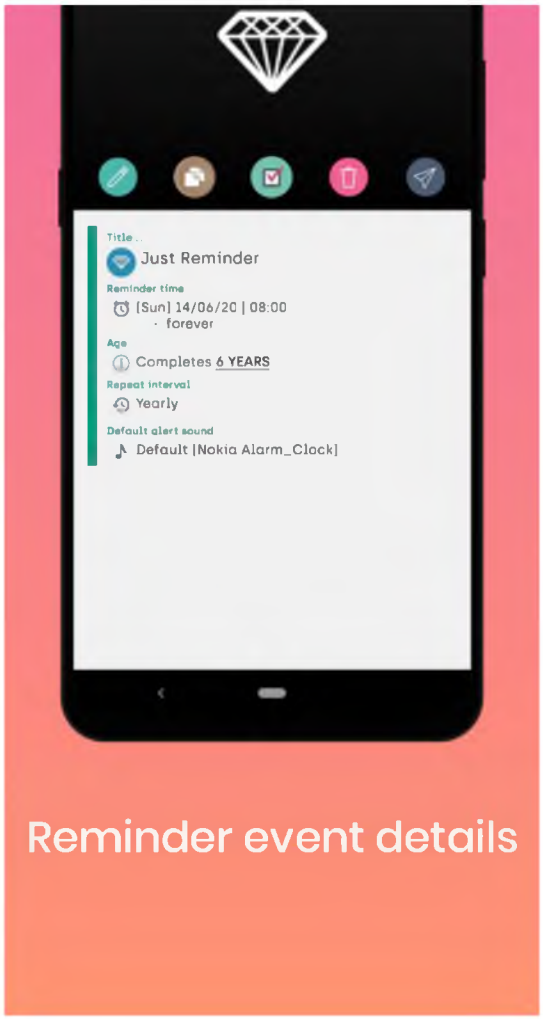

Рисунок 1.4 – Головна сторінка застосунку Just Reminder with Alarm

Для студентів даний застосунок може бути корисним при використанні нагадування про дедлайни, також сюди можна додати й інші події академічного життя. Недоліком додатку є те що, немає можливості додати справи, які треба повторювати з інтервалом.

2. Вибір принципу дії системи

2.1. Аналіз потреб користувачів

Проаналізувавши вище наведені приклади застосунків-органайзерів, можна зробити висновок, що користувачі хочуть бачити у своєму органайзері зручний для використання інтерфейс [14], можливість розділяти розклад занять та завдання до даних занять, можливість додавання, видалення, редагування занять та завдань, а також застосунок, який буде адаптуватися до різних операційних систем.

2.2. Вибір технологій та інструментів розробки

Відповідно до поставленого завдання для розробки застосунку-органайзера були використані відповідні інструменти. Для середовища розробки було вибрано **Android Studio**, адже вона є зручною у використанні і одною з найкращих при розробці мобільних застосунків, а також включає в собі велику кількість емуляторів-смартфонів та види збірок й генерацію декількох .apk файлів. В основі лежить мова програмування **Dart**, універсальна у використанні мова, яка часто використовується при створенні мобільних застосунків. Dart легкий в опануванні, доступний на всіх системах та емуляторах, а також, допомагає зробити інтерфейс застосунку приємним для користувача, уникаючи смикань при використанні [6]. Використовуючи Dart можна використати лише один великий фреймворк – **Flutter** [16]. З його допомогою ми створюємо елементи інтерфейсу програми з готових віджетів. Код, що написаний мовою Dart, Flutter компілює в обробляючий процесором код ARM [8]. Для бази даних було використано **Firebase**. Firebase – база даних NoSQL, вона дозволяє створити необхідний хостинг, автентифікацію користувачів, базу даних в режимі реального часу. Firebase зручна у використанні та встановленні, пропонує повну інтеграцію з різними

службами Google, такими як Google Analytics, Google Ads і Google Cloud, а також дозволяє писати окремі функції, які відповідають на події, викликані функціями Firebase і запитами HTTPS.

Для головних бібліотек було використано бібліотеку **intl** – використовується для форматування дати (наприклад: DateFormat _dateFormatter = DateFormat('dd MMM, yyyy');); **firebase_core** – для використання служби бази даних firebase; **cloud_firestore** – для підключення фактичного сховища бази даних для операції CRUD; **firebase_auth** – для автентифікації та реєстрації користувача в програмі. Також з основних функціональностей у програмі-органайзері було використано Singleton [9]. **Singleton** використовується для того щоб мати єдиний екземпляр класу, до якого можна отримати доступ із кількох частин програми. Ми створюємо Singleton для керування станом на рівні програми. У файлі singleton_appdata.dart ми додаємо int totalTaskCount = 0 та int totalSubjectCount = 0, які ініціалізуються під час запуску програми, і вони доступні глобально в будь-якій частині будь-якого екрана програми ґ.

3. Розробка програмного застосунку

3.1. Постановка основного завдання

Після дослідження було вирішено створити мобільний застосунок на мові програмування Dart з використанням фреймворку Flutter, який буде мати систему авторизації користувача та три головні сторінки, а саме: для додавання, редагування й видалення предметів у розкладі, для додавання, редагування й видалення завдань у таблиці завдань, та налаштування.

3.2. Опис розробки продукту

3.2.1. Основні елементи

Починаємо ми програму з файлу main.dart і підключаємо базу даних за допомогою файлу google-services.json. У файлі main.dart ми ініціалізуємо підключення до Firebase нашого застосунку (Firebase.initializeApp) , а також описуємо основні функції та віджети. У stateless віджет [10] MaterialApp ми створюємо основний стиль нашого додатку, вказуючи кольори, стиль тексту. Дальше створюється stateful віджет [10] MyHomePage, для головної сторінки застосунку. А також initState, що ініціалізує поточного користувача, його завдання, та розклад занять, для цього використовуються функції getCurrentUser(), getTasksCount() та getSubjectCount(). Для навігації між головними сторінками (за допомогою кнопок, які знаходяться внизу) створюється віджет Scaffold, який ініціалізує основні кнопки (Timetable, Tasks та Settings).

Після цього створюється три файли Timetable.dart, TaskScreens.dart та Settings.dart, вони всі наслідують StatefulWidget.

Файл Timetable використовується для створення головної сторінки застосунку, а саме розкладу (рисунок 3.1).

Рисунок 3.1 – Головна сторінка Timetable

У класі Timetable (розклад) оголошуються основні змінні, такі як _selectedDayIndex, selectedDay, today. А також final List<String> _weekDays , List<Subject> object, а також String можливих кольорів предметів для подальшого використання. Дальше створюється віджет Scaffold, для ініціалізації тексту, контейнера, стилю тексту, детектора натискань (GestureDetector). Також віджет StreamBuilder, для побудови відображення занять і сторінки у тому разі коли заняття ще не додані. Він дозволяє оновлювати вміст інтерфейсу користувача кожного разу, коли дані у потоці будуть змінюватися[11]. Наступним кроком створюється кнопка для додавання на екран нового заняття. (рисунок 3.2), а також метод для виклику на екран розкладу дня, який є сьогодні. (setTodayTimetable()).

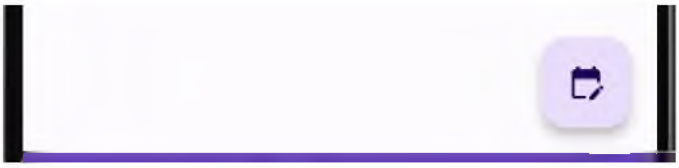

Рисунок 3.2 – Кнопка для додавання нового заняття

Файл TaskScreens використовується для створення сторінки завдань (рисунок 3.3), на якій відображаються виконані і невиконані завдання.

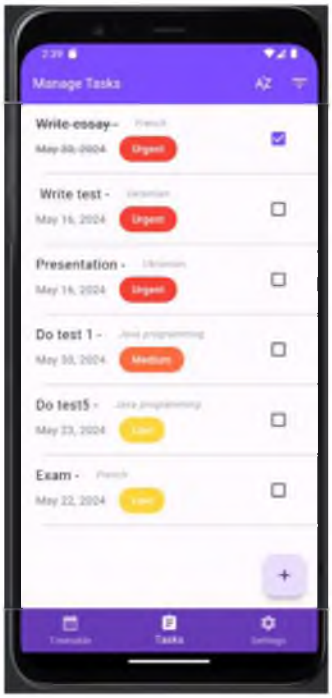

Рисунок 3.3 – Сторінка завдань

Всередині TaskScreens оголошуються основні змінні, такі як _dateFormatter, selectedFilter, isDescending, що використовуються пізніше.

Створюється віджет _buildTask, для створення тексту, його стиля, розміщення завдань на сторінці, ScaffoldMessenger, для спливаючого повідомлення коли завдання відредаговано або виконано, MaterialPageRoute [12] для зручного переходу між сторінками «завдання» та «додати завдання», Divider для розміщення лінії-роздільника на сторінці.

Після цього створюється віджет Scaffold для ініціалізації кнопок та фільтрування(за статусом, пріоритетом та датою), для фільтрування також використовується метод filter(status).

Файл Settings використовується для створення сторінки налаштувань, де ви можете продивитись власний профіль або вийти з нього.

Він використовує основний віджет buildSettingsList() для ініціалізації тексту, розташування всіх елементів, а метод signOut() для користувача виходу зі свого акаунту.

Щоб додавати, редагувати або видаляти заняття та завдання необхідно, щоб у вікні при натисканні відповідних кнопок, що знаходяться внизу екрана відкривались нові сторінки, тому для цього було створено файли AddTask.dart та AddSubject.dart, за допомогою який вони реалізуються.

AddSubject.dart використовується для побудови віджету для сторінки додавання нового предмету у розклад (рисунок 3.4).

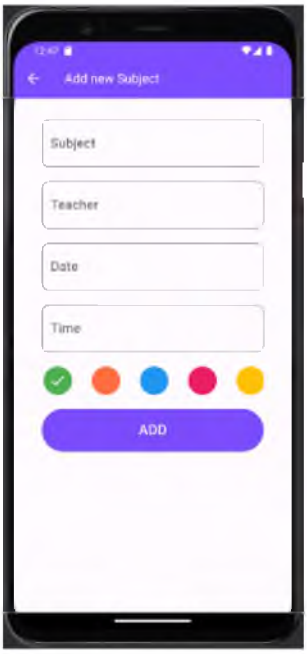

Рисунок 3.4 – Додавання нового предмету

Він містить необхідні поля ініціалізації, методи для вибору дати (_handleDatePicker()) та часу (timePick()), метод dispose(),який викликається для звільнення пам'яті, яка була використана для кожного контролера, а також для припинення слідкування за змінами даних, які можуть виникнути в цих контролерах.

Також у цьому ж файлі створюється віджет build, для створення тексту, його стиля, розміщення елементів на сторінці, і методи _submit(), populateField(), для вибору кольору, яким буде відображатися предмет на сторінці, showDeleteDialog(), для відкриття діалогового вікна при видаленні предмету, _delete() для спливаючого вікна, що повідомляє користувачу, що предмет успішно видалено.

Наступним важливим елементом програмного коду є файл AddTask.dart, що використовується для побудови віджету для сторінки додавання нового завдання до списку завдань (рисунок 3.5).

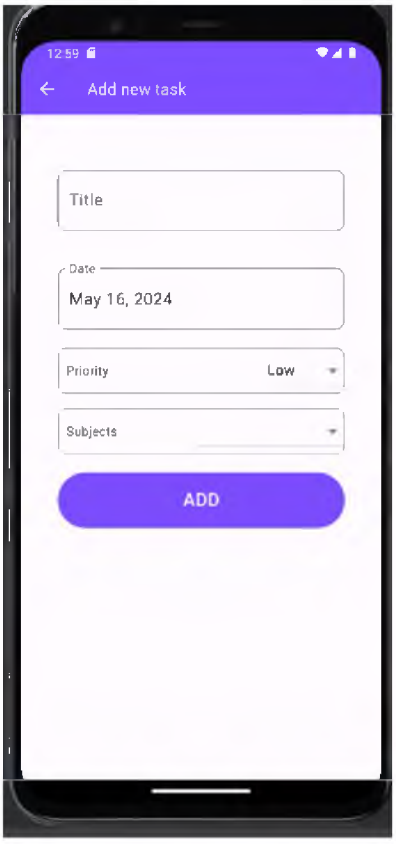

Рисунок 3.5 – Додавання нового завдання

AddTask містить необхідні поля ініціалізації, методи _handleDatePicker(), для вибору дати, аналогічний віджет build, для створення тексту, його стиля, розміщення елементів на сторінці. А також методи _submit(), для додавання нового завдання з усіма необхідними полями, showDeleteDialog(), для спливаючого діалогового вікна щоб упевнитись що користувач дійсно хоче видалити завдання, та _delete(), для спливаючого вікна, що повідомляє користувачу, що предмет успішно видалено.

3.2.2. Авторизація користувача

Наступним було створено для файли для ініціалізації користувача: Login.dart та Signup.dart. Вони є основними для авторизації вже існуючого користувача та реєстрації нового користувача (рисунок 3.6)

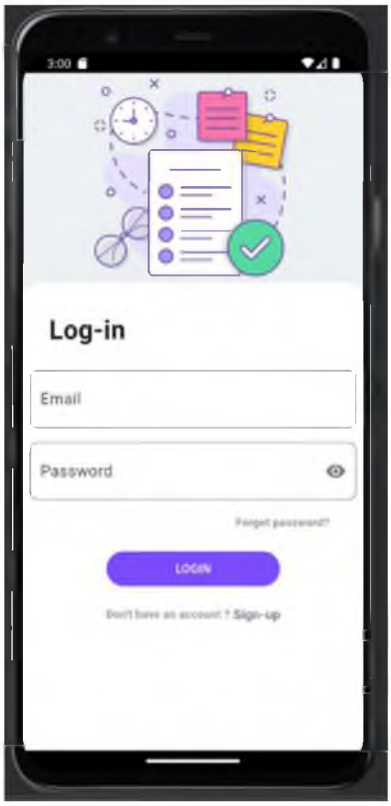

Рисунок 3.6 – Вхід користувача в застосунок

Login.dart містить у собі такі основні змінні, як _userEmail, _userPswd, _loginFormKey, _auth, що використовуються для авторизації користувача.

Головний метод login() перевіряє чи є вже зареєстрованим, звіряючи введений пароль та логін по базі даних. Також використовується віджет build для побудови сторінки (рисунок 3.6), ініціалізації тексту, кольору, розташування всіх елементів, валідатору, який провіряє чи введено пароль, додавання функції «Forget password» та «Don't have an account » та кнопки Sign up.

Signup.dart містить у собі такі основні змінні _userEmail, _userPswd, _userName, _auth.

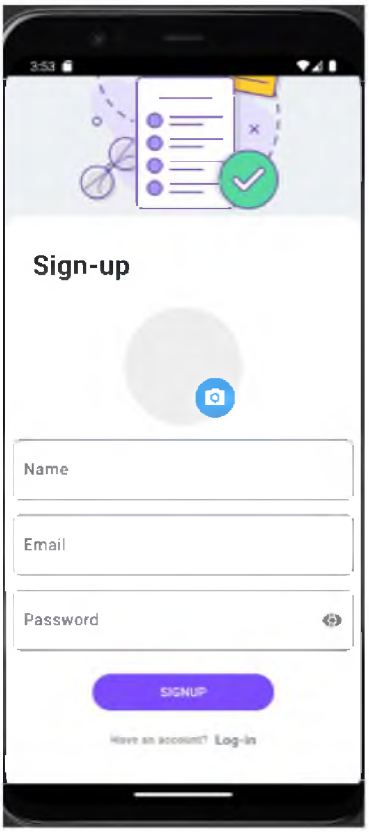

Рисунок 3.7 – Реєстрація користувача

Метод _signupUser() додає нового користувача до бази даних, після введення ім'я, пароля та логіну(рисунок 3.7), створює спливаюче вікно при успішній реєстрації. Також використовується віджет build для ініціалізації тексту, кольору, розташування всіх елементів, валідатору, що перевіряє чи введено логін та пароль, додавання функції «Have an account?» та кнопки Log-in.

3.2.3. Допоміжні елементи

Наступним кроком було створено папку components та common. В яких створені відвідно файли SubjectsDropdown.dart , CustomDropDown.dart , ColorsPicker.dart, Card.dart, PageHeading.dart та PageHeader.dart. Дані файли є допоміжними елементами в застосунку. SubjectsDropdown.dart та CustomDropDown.dart створюють список-вибірку предметів та пріоритетів відповідно на сторінці додавання нового завдання (рисунок 3.8).

Рисунок 3.8 – Списки-вибірки на сторінці додавання нового завдання

Card.dart за допомогою віджету ListView.builder створює список-вибірку кольорів на сторінці додавання нового предмету (рисунок 3.9).

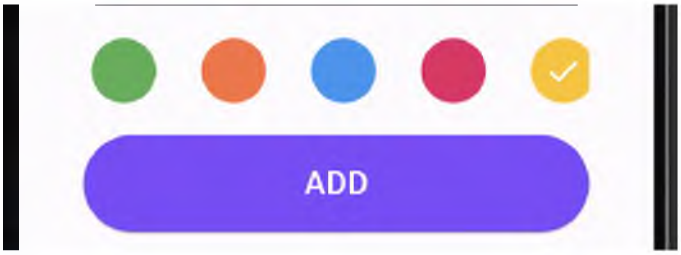

Рисунок 3.9 – Список-вибірки на додавання нового предмету

PageHeading.dart та PageHeader.dart у свою чергу створюються для створення початковий екран, при запуску застосунку, додавання головної іконки.

Також створюємо singleton_appdata.dart, який використовується для зберігання даних в середині коду, які ініціалізуються під час запуску програми, і доступні в будь-якій частині будь-якого екрана програми [13]. FirebaseServices.dart допоміжний файл для аутентифікації користувачів за id.

Ще одними допоміжними елементами у створенні програмного коду є файли subject_model.dart та task_model.dart, за допомогою них ми створюємо класи для завдань (Task) та занять (Subject), визначаючи їхні основні змінні.

3.3. Користувацький інтерфейс

Метою було виконати інтерфейс у зручній, легкій та інтуїтивно-зрозумілій для користувача манері.

Коли користувач відкриває мобільний застосунок він бачить перед собою наступне вікно (рисунок 3.10)

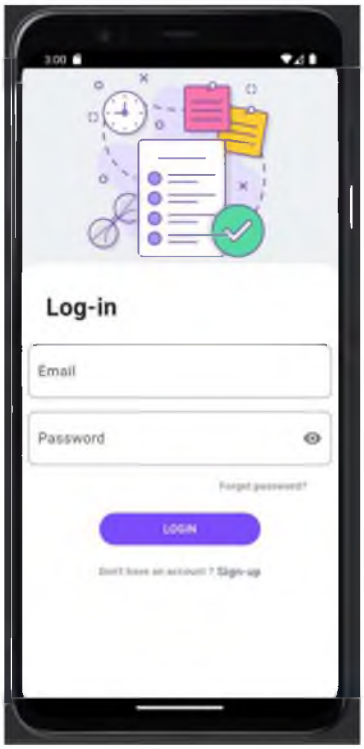

Рисунок 3.10 – Вхід до органайзера

Якщо користувач уже зареєстрований він вводить свій логін та пароль, після чого входить у систему, якщо пароль та логін збігаються, у іншому разі його буде повідомлено про неправильно введені дані і дано змогу ввести їх ще раз.

Якщо ж користувач не зареєстрований, то він переходить на сторінку реєстрації, натиснувши кнопку Sign-up(рисунок 3.7)

Успішно увійшовши або зареєструвавшись, користувач потрапляє до сторінки розкладу занять(рисунок 11)

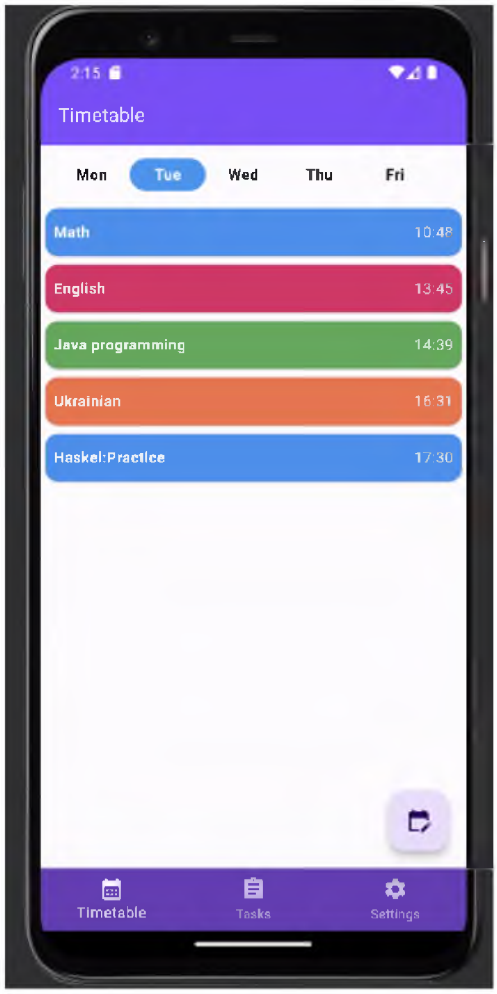

Рисунок 3.11 – Сторінка розкладу занять

Тут користувач має змогу редагувати, видаляти, додавати предмети, за допомогою кнопки в правому нижньому куті екрана. Натиснувши на неї користувач перейде до сторінки додавання нового заняття (рисунок 3.12), вона ж буде відображатися і при натисканні на вже існуючий предмет, але з уже заповненими раніше полями, що можна редагувати.

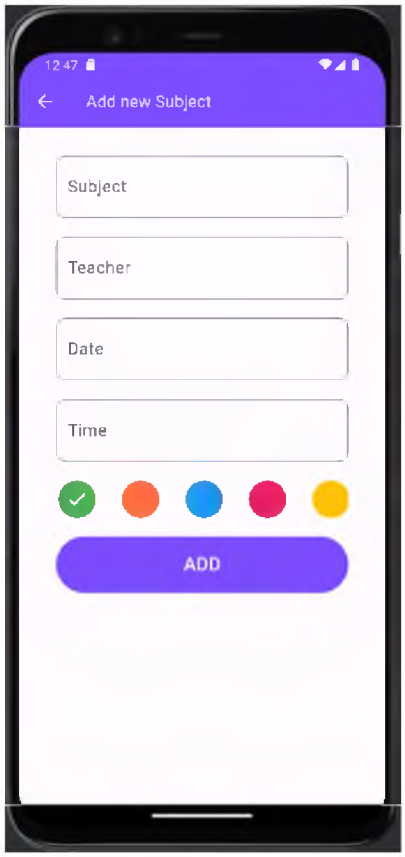

Рисунок 3.12 – Додавання нового предмету

Щоб додати новий предмет користувачу потрібно натиснути кнопку «Add», а щоб переходити по дням тижня використовувати список тижнів вверху та натискати на потрібний.

Щоб перейти до сторінки таблиці завдань, користувач повинен скористатися нижньою панеллю, після чого він опиниться на потрібній сторінці(рисунок 3.13), тут користувач може додавати, редагувати та видаляти завдання, які йому потрібно виконати. Також він може відсортувати всі наявні завдання, натиснувши кнопку в правому верхньому куті сторінки. Щоб відмітити завдання виконаним користувачу потрібно натиснути в квадрат, який розташований біля кожного предмету, після цього предмет буде перекреслено.

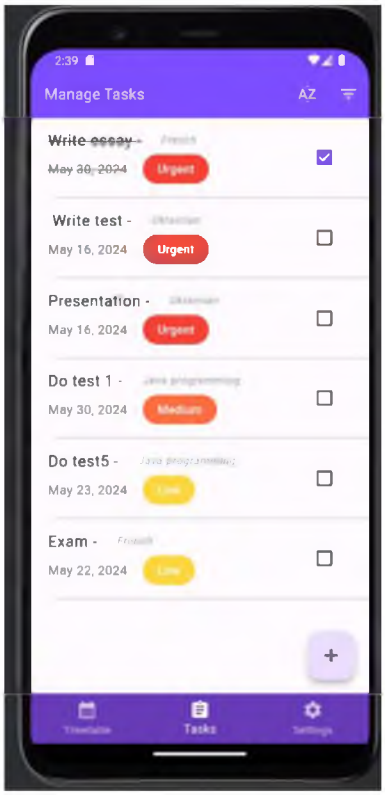

Рисунок 3.13 – Сторінка таблиці завдань

Щоб додати нове завдання користувачу потрібно натиснути на кнопку, що розташована в правому нижньому куті сторінки, тоді перед ним з’явиться вікно додавання завдання. (рисунок 3.15)

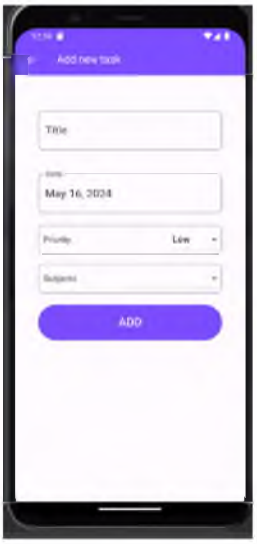

Рисунок 3.15 – Сторінка додавання нового завдання

Користувач повинен заповнити всі необхідні форми, вибрати назву, дату, пріоритет та предмет, після чого натиснути кнопку «Add», після цього нове завдання з’явиться у списку завдань.

Користувач може переглядати, редагувати, видаляти завдання та предмети, для редагування та видалення потрібно просто натиснути на необхідний предмет або завдання, після чого відкриється вікно в якому можна буде провести потрібні дії, також користувач може вийти з застосунку, перейшовши в меню налаштування, скориставшись нижньою панеллю.

Висновки

У результаті дослідження багатьох ресурсів і проектування архітектури, було створено мобільний застосунок-органайзер, що стане в пригоді студентам, які навчаються та хочуть правильно й якісно організовувати власний академічний час.

Органайзер має всі необхідні функції для прослідковування та редагування розкладу предметів, а також для редагування, видалення та додавання завдань, які потрібно виконати. У зв’язку з розвитком сучасних технологій та переходу багатьох студентів на дистанційну форму навчання даний застосунок-органайзер може стати чудовим помічником кожному студенту для моніторингу, організування та контролю його академічної діяльності.

Список використаної літератури

1. Мобільні додатки-органайзери, які допоможуть встигнути все. [Електронний ресурс].

   Режим доступу:

https://zhyvyaktyvno.org/news/mobln-dodatki-organajzeri-yak-dopomozhut-vstignuti-vse

2. Органайзер Any.do. [Електронний ресурс].
   Режим доступу:
   https://www.any.do
3. To Do: Planner, Task Reminders [Електронний ресурс].
   Режим доступу:
   https://apps.apple.com/ie/app/to-do-planner-task-reminders/id6449979760
4. Тижневий планувальник [Електронний ресурс].
   Режим доступу:
   https://play.google.com/store/apps/details?id=com.jonasbernardo.developer.planejamento_semanal&hl=uk&gl=US
5. Just remind me with alarm [Електронний ресурс].
   Режим доступу:
   https://play.google.com/store/apps/details?id=in.smsoft.justremind&hl=uk&gl=US
6. Мова програмування Dart та Flutter [Електронний ресурс].
   Режим доступу:
   https://highload.today/uk/blogs/za-shho-my-lyubymo-dart-golovnu-ta-yedynu-movu-stvorennya-dodatkiv-na-flutter/
7. Flutter.Основні відомості [Електронний ресурс].
   Режим доступу:
   https://avada-media.ua/ua/services/flutter/
8. Процесор ARM [Електронний ресурс].
   Режим доступу:

https://evo.net.ua/shcho-take-protsesor-arm-vse-shcho-vam-potribno-znati/

9. Singleton in Flutter [Електронний ресурс].
   Режим доступу:
   https://medium.com/@ahmedtahaelelemy/singleton-in-flutter-7747bfda6d20

10. Основні типи віджетів у Flutter [Електронний ресурс].
    Режим доступу:
    https://drukarnia.com.ua/articles/flutter-widgets-stateless-stateful-inherited-bQoC9

11. StreamBuilder<T> class [Електронний ресурс].
    Режим доступу:
    https://api.flutter.dev/flutter/widgets/StreamBuilder-class.html

12. MaterialPageRoute <T> class [Електронний ресурс].
    Режим доступу:
    https://api.flutter.dev/flutter/material/MaterialPageRoute-class.html

13. Singleton – шаблон проектування в Dart [Електронний ресурс].
    Режим доступу:
    https://medium-com.translate.goog/@swe.jamirulinfo/singleton-is-a-design-pattern-in-dart-98dd947c6dd1?_x_tr_sl=en&_x_tr_tl=uk&_x_tr_hl=uk&_x_tr_pto=sc&_x_tr_hist=true

14. Головне про зручність в інтерфейсах [Електронний ресурс].
    Режим доступу:
    https://cases.media/en/article/golovne-pro-zruchnist-v-interfeisakh

15. Революціонізація Навчання та Розвитку [Електронний ресурс].

Режим доступу:
https://cluelabs.com/blog/революціонізація-навчання-та-розвит/

16. Чому флаттер найкращий вибір для мобільного застосунку [Електронний ресурс].
Режим доступу:
https://spacelab.ua/articles/chomu-flutter-najkrashij-vibir-dlya-zapusku-vashogo-mobilnogo-dodatku/
17. Створення застосунків на Flutter [Електронний ресурс].
Режим доступу:
https://wezom.com.ua/ua/flutter
18. Тестування додатків на Flutter: процес та труднощі [Електронний ресурс].
Режим доступу:
https://wezom.com.ua/ua/blog/testuvannya-dodatkiv-na-flutter-protses-ta-trudnoschi