|  |
| --- |
| Министерство науки и высшего образования Российской Федерации |
| федеральное государственное бюджетное образовательное учреждение высшего образования |
| "Российский экономический университет имени Г.В. Плеханова" |
| **МОСКОВСКИЙ ПРИБОРОСТРОИТЕЛЬНЫЙ ТЕХНИКУМ** |

МДК 04.01 Внедрение и поддержка компьютерных систем

ПМ 04 Сопровождение и обслуживание программного обеспечения компьютерных систем

**Специальность: 09.02.07 «Информационные системы и программирование»**

Квалификация: программист

|  |  |
| --- | --- |
| Выполнил студент: | Проверил преподаватель: |
| Группы П50-7-20 | Пахомов Д. А. |
| Огурцов А.А. | «\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2023 года |
| «13» сентября 2023 год | Оценка: \_\_\_\_\_\_\_\_\_ |
|  |  |

# **ПРАКТИЧЕСКАЯ РАБОТА №1**

Тема: язык программирования dart

Цель работы: познакомились с основной написания программы на языке dart.

Создать класс Student с переменными: id, surname, middle, first-name, age, group, grade.

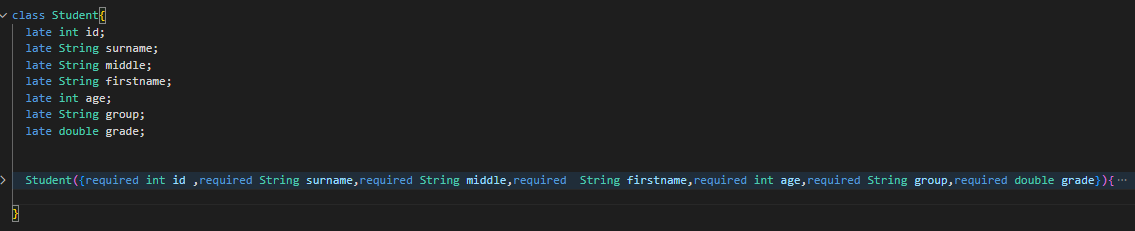


Рисунок 1. Задача 1.

Создаём конструктор с именованными аргументами.

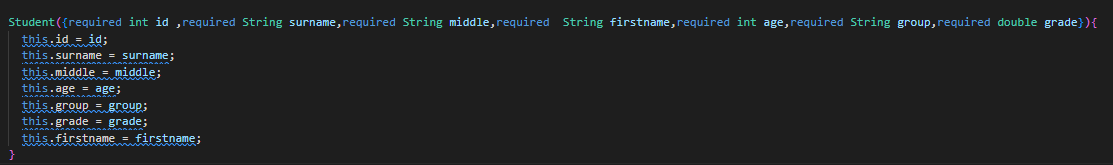


Рисунок 2. Задача 2.

Создаём класс StudentList, который имеет переменную list<Student> равную null.

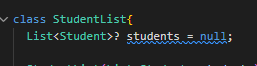


Рисунок 3. Задача 3.

Создаём 20 студентов. Для этого пишем метод, который возвращает список студентов.



Рисунок 4. Задача 4.

В классе StudentList делаем метод addStudent, который принимает объект Student и добавляет его в список students

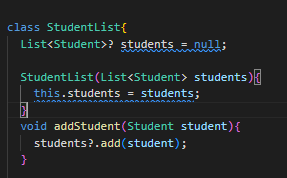


Рисунок 5. Задача 5.

Написан метод getStudentWithHighestGrade, который возвращает объект Student с наивысшим средним или наименьшем, в зависимости от 2 аргумента.

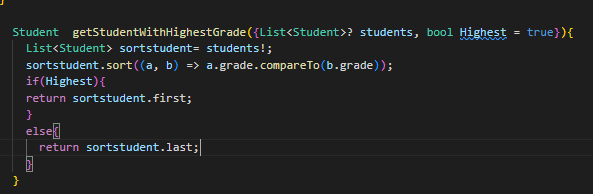


Рисунок 6. Задача 6.

Написан метод getAverageGrade, который возвращает средний балл всех студентов в списке.

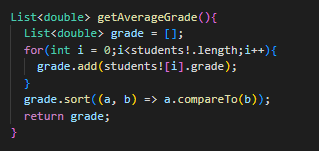


Рисунок 7. Задача 7.

Написан метод который удалят студента по id.

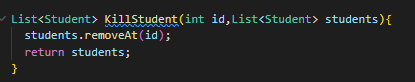


Рисунок 8. Задача 8.

Написан метод который изменяет данные студента.

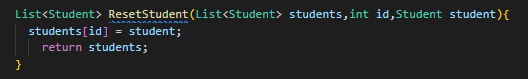


Рисунок 9. Задача 9.

Написан метод который будет выводит всех студентов так же реализовать возможность вывода от фильтрованных данных.

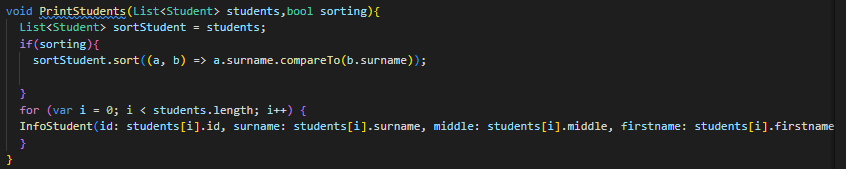


Рисунок 10. Задача 10.

Вывод: мы познакомились с основной написания программы на языке dart.

# **ПРАКТИЧЕСКАЯ РАБОТА №2**

Тема: Использование основных виджетов

Цель работы: С использованием VS Code сделать мобильное приложение.

Ход работы:

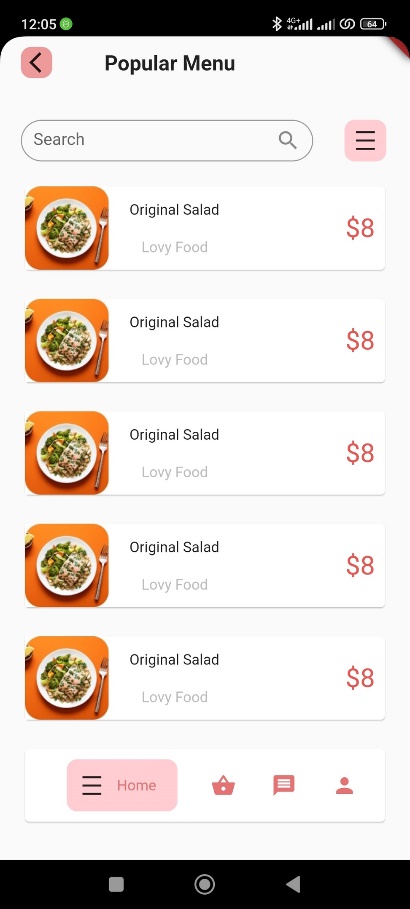


Рисунок 11. Страница 2.

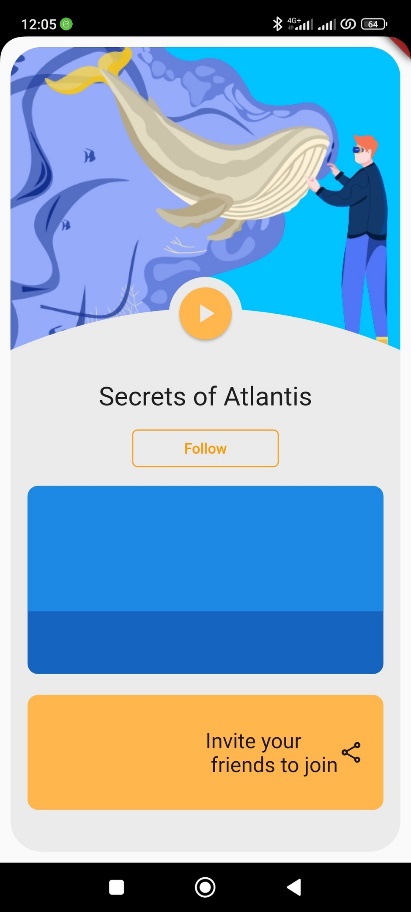


Рисунок 12. Страница 1.

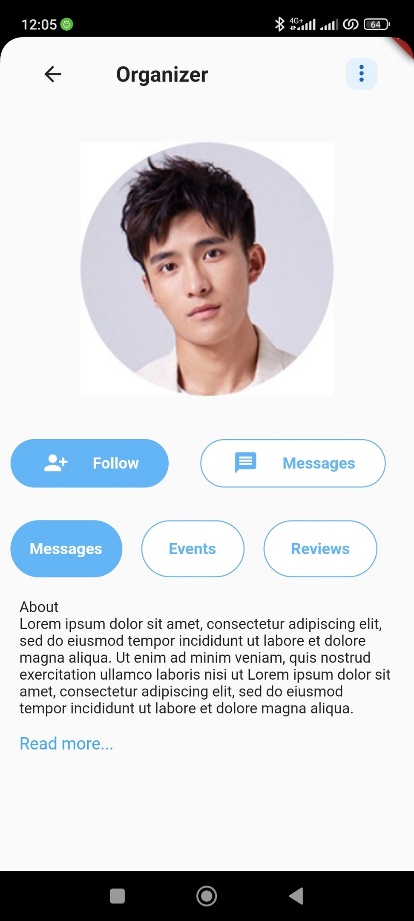


Рисунок 13. Страница 3.

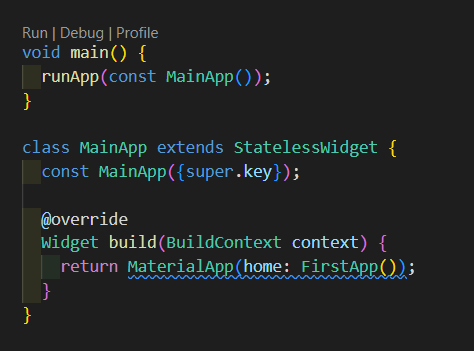


Рисунок 14. Главный запуск.

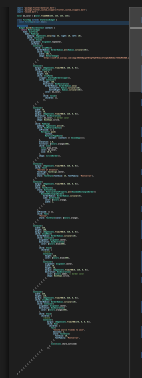


Рисунок 15. Код 1 страницы.



Рисунок 16. Код 2 страницы.



Рисунок 17. Код 3 страницы.

Вывод: была выполнена поставленная цель.

# **ПРАКТИЧЕСКАЯ РАБОТА №3**

Тема: Калькулятор

Цель работы: на 3 создать калькулятор с основными действиями + - / \*   
на оценку 4 создать калькулятор с основными действиями а так же перевод в разные системы счисления.  
на оценку 5 создать калькулятор с выше перечисленными действия но так чтобы подсчет был в одну строчку.

Ход работы:



Рисунок 18. Код. Часть 1



Рисунок 19. Код. Часть 2.

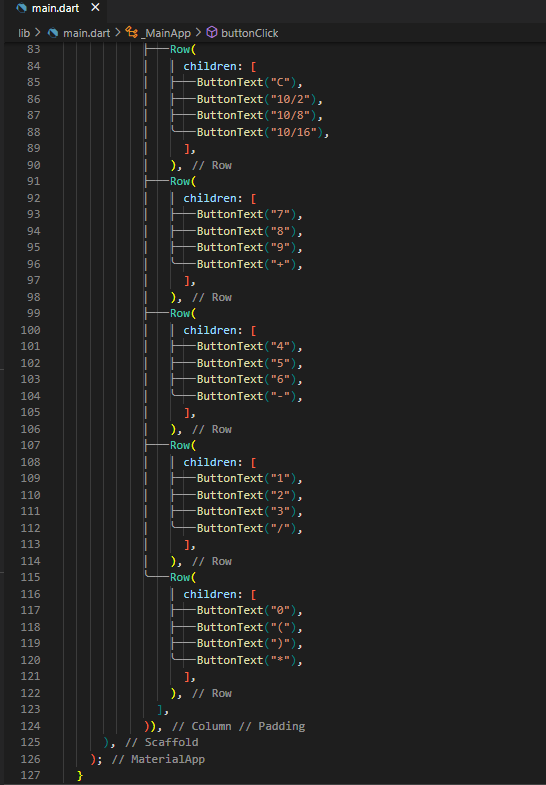


Рисунок 20. Код. Часть 3.

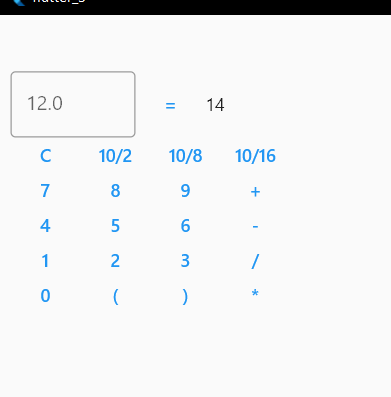


Рисунок 21. Результат.

Вывод: была выполнена поставленная цель.

# **ПРАКТИЧЕСКАЯ РАБОТА №4**

Тема: Навигация

Цель: Реализовать навигацию через следующие функцию onGenerateRoute. Использовать ListView возможность менять позицию элементов как показано на рисунке Использовать Provider для своего проекта. Использовать lazy, value для Provider.

Ход работы:

Создаем onGenerateRoute:

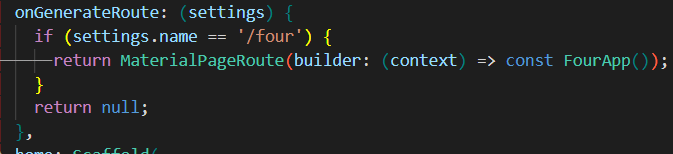


Рисунок 22. OnGenerateRoute.

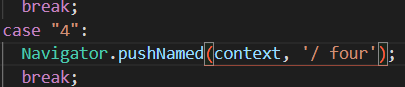


Рисунок 23. Вызов.

Создаём ListView со сменой позиции:



Рисунок 24. ListView обявление и изменение позиций.

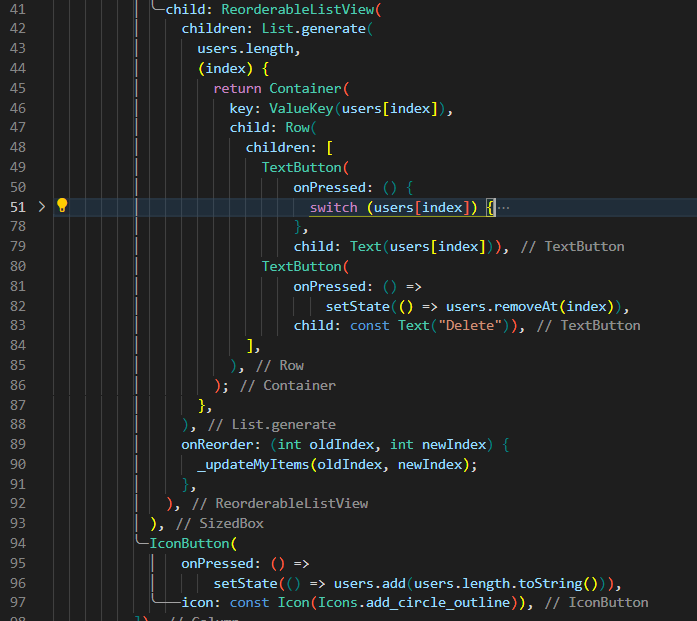


Рисунок 25. ListView.Generate.

Создаём Provider:

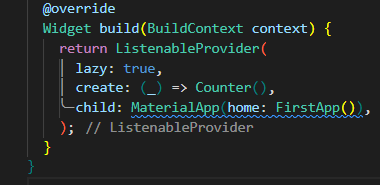


Рисунок 26. Инцилизация.

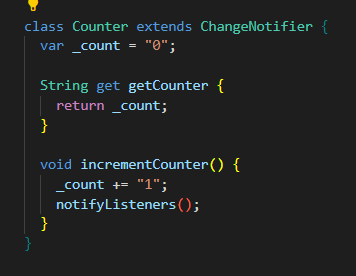


Рисунок 27. Класс.

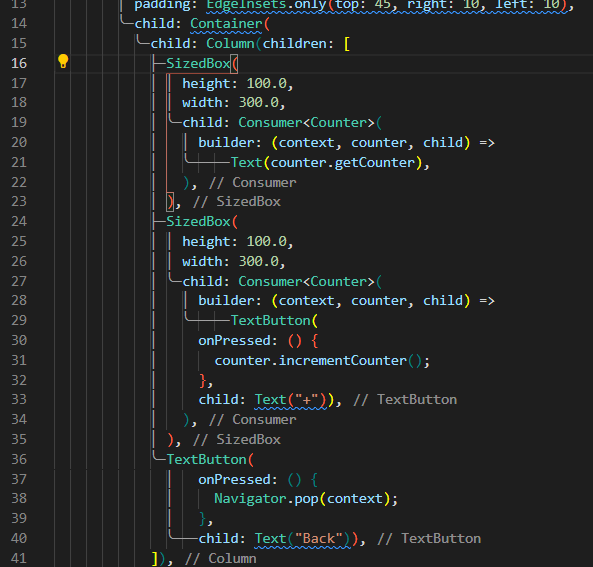


Рисунок 28. Работа с классом через провайдер.

Результат:

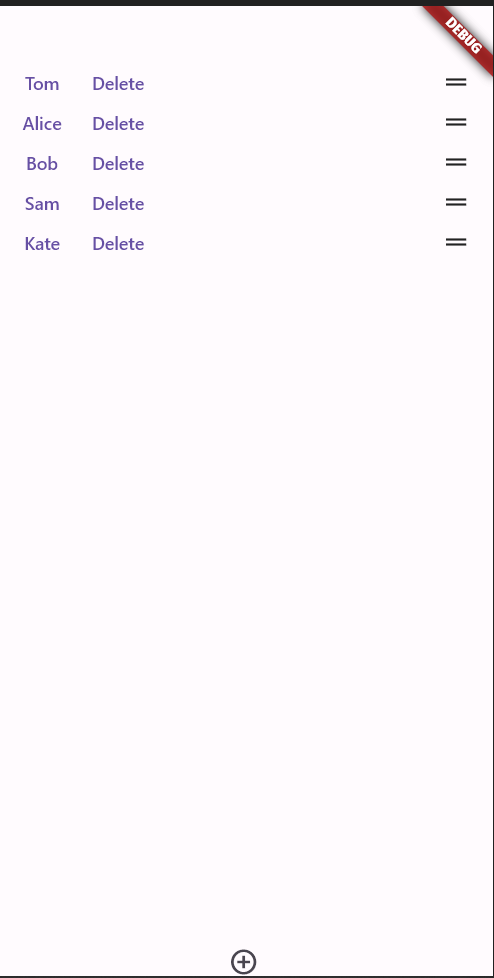


Рисунок 29. Экран 1.



Рисунок 30. Экран 2.



Рисунок 31. Экран 3.

Вывод: была выполнена поставленная цель и сделан отчёт в Word.

# **ПРАКТИЧЕСКАЯ РАБОТА №5**

Тема: Библиотеки.

Цель: Подключите библиотеку Hive. Перепишите четвертую практическую работу так, чтобы все данные хранились в этой библиотеке. Добавьте новое окно "Настройки", где должны быть доступны следующие данные: ФИО, дата рождения, смена темы. Системные настройки должны храниться в SharedPreferences. А также реализуйте возможность входа в программу с использованием кода, который будет храниться в SharedPreferences.

Ход работы:

Библиотеки:

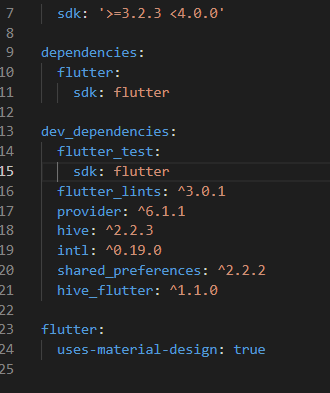


Рисунок 32. Библиотеки.

Работа с Hive:



Рисунок 33. Объявление.

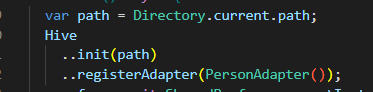


Рисунок 34. Инцилизация.

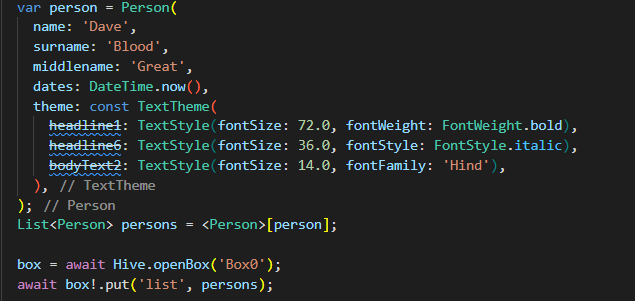


Рисунок 35. Создание каробки.



Рисунок 36. Вытаскивание данных.



Рисунок 37. Добавление.

Работа с Shared Preference:



Рисунок 38. Объявление.



Рисунок 39. Инцилизация.



Рисунок 40. Выгрузка данных.

Вывод: была выполнена поставленная цель и сделан отчёт в Word.

# **ПРАКТИЧЕСКАЯ РАБОТА №6**

Тема: Фотографии.

Цель: Расширить функционал приложения на основе предыдущих практических работ, добавив возможность загрузки фотографий с камеры или галереи при добавлении новых элементов. Также необходимо сохранять геопозицию фотографии и путь к ней в базе данных Hive.

camera: ^0.11.0+1 - для работы с камерой.image\_picker: ^0.8.4+2 - для выбора изображений из галереи.hive: ^2.0.4 - для работы с локальной базой данных.

Реализовать функционал добавления новых элементов с возможностью загрузки фотографий:

Добавить кнопку для открытия камеры и галереи.

При выборе камеры, использовать пакет camera для захвата фотографии.

При выборе галереи, использовать image\_picker для выбора изображения.

Сохранить полученное изображение внутри приложения.

Реализовать сохранение геопозиции фотографии:

Использовать пакет location для получения текущей геопозиции.

Сохранить геопозицию вместе с фотографией.

Интегрировать Hive для хранения пути к фотографиям:

Создать модель данных для фотографии, включая путь и геопозицию.

Сохранить данные в Hive при добавлении нового элемента.

Ход работы:

Библиотеки:

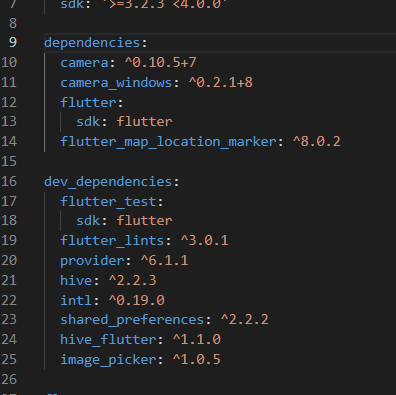


Рисунок 41. Библиотеки.

Результат:

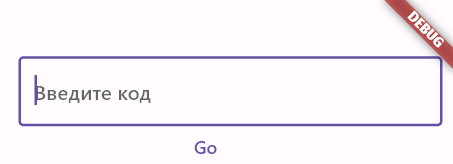


Рисунок 42. Код.

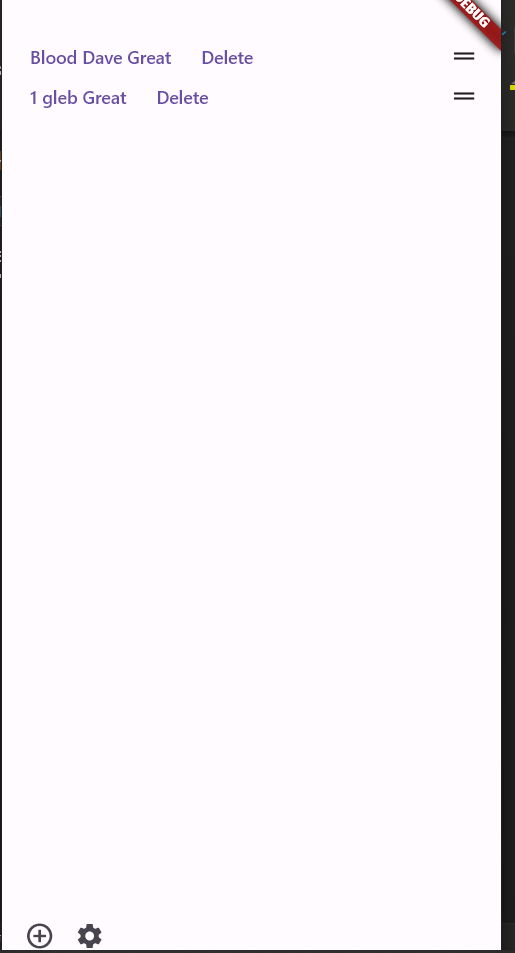


Рисунок 43. Окно 1.

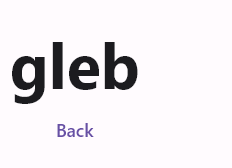


Рисунок 44. Глеб.



Рисунок 45. Настройки.

Изменение класса для сохранения фотографий:

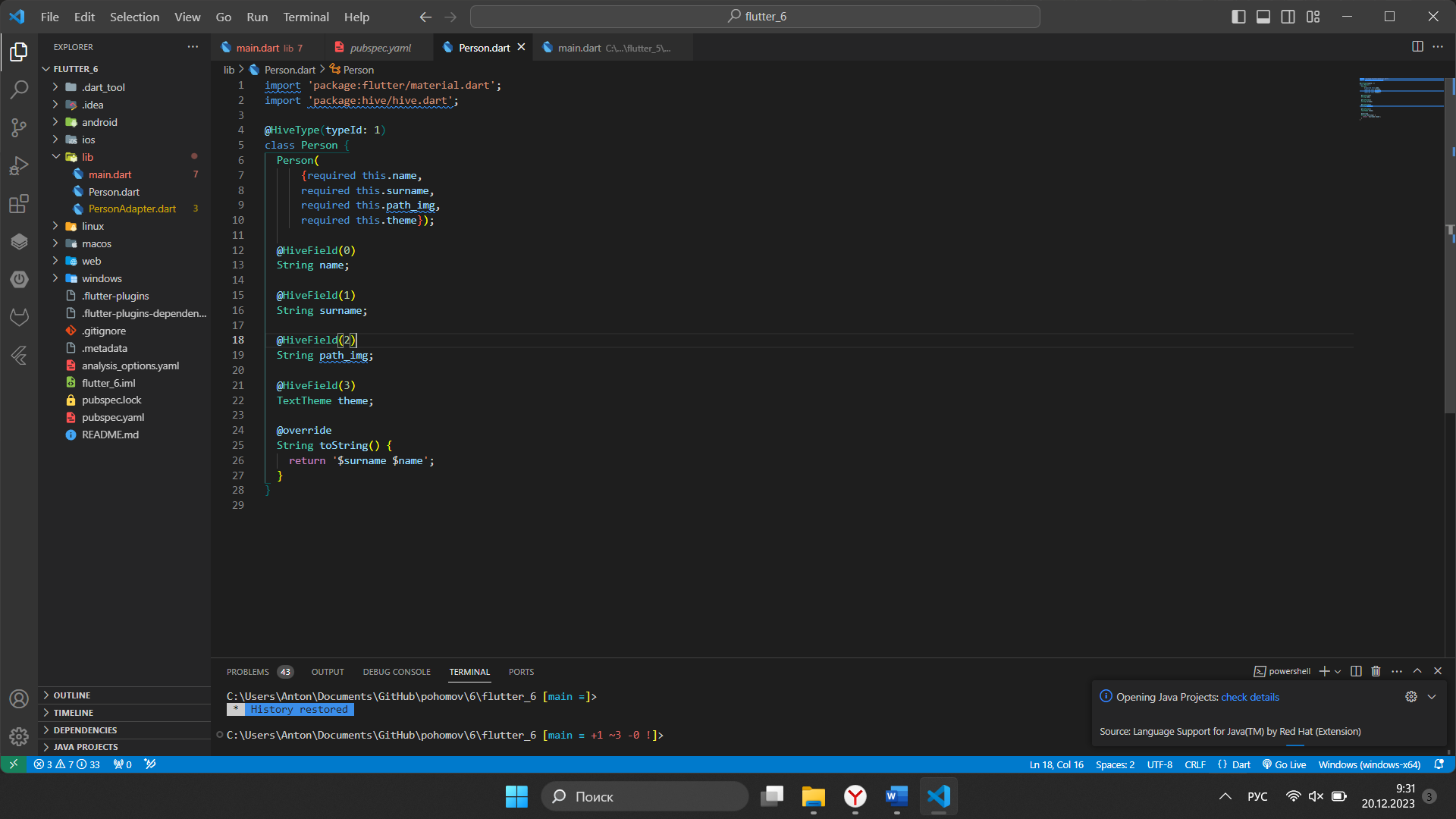


Рисунок 46. Переменная для пути сохранения.

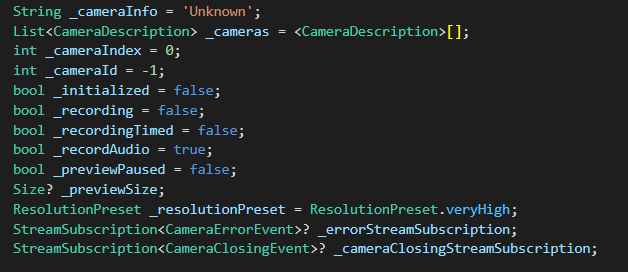


Рисунок 47. Объявление.



Рисунок 48. Классы для работы.



Рисунок 49. Инцилизация камеры.

Работа с местоположением:

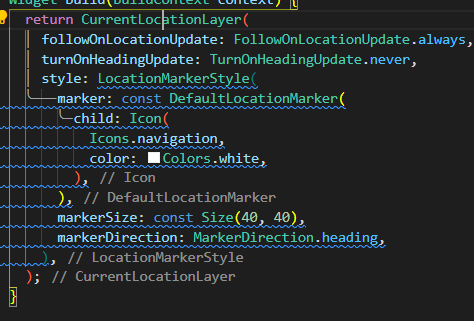


Рисунок 50. Работа с локацией.

Вывод: Поставленная цель оказалась сложно выполнимой на windows и не была реализована.

# **ПРАКТИЧЕСКАЯ РАБОТА №7**

Тема: Навигация.

Цель: На оценку 4:

Реализовать bottomNavigationBar с переходом между страницами: +1 балл (если нет навигации - 0,5 балла).

Реализовать Drawer с минимум 3 элементами и навигацией: +1 балл (-0,25 балла, если элементов не хватает, и также, если нет навигации).

Реализовать AppBar с минимум 2 элементами в action, а также использовать PopupButtonMenu (+0,3 балла), AlertDialog (+0,3 балла) и SnackBar (+1 балл) (-0,5 балла, если элементов не хватает).

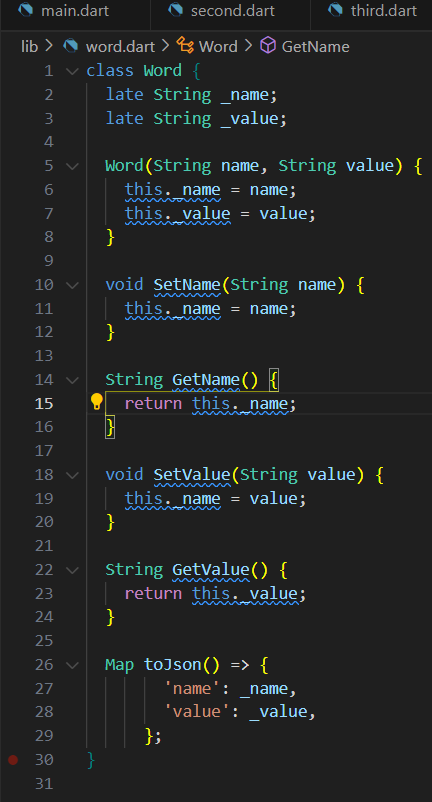


Рисунок 51. Класс Слово.

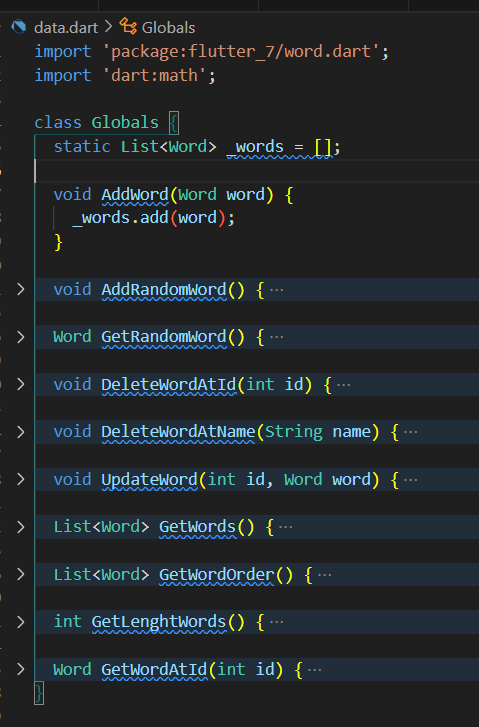


Рисунок 52. Работа с классов слова. Круд действия..

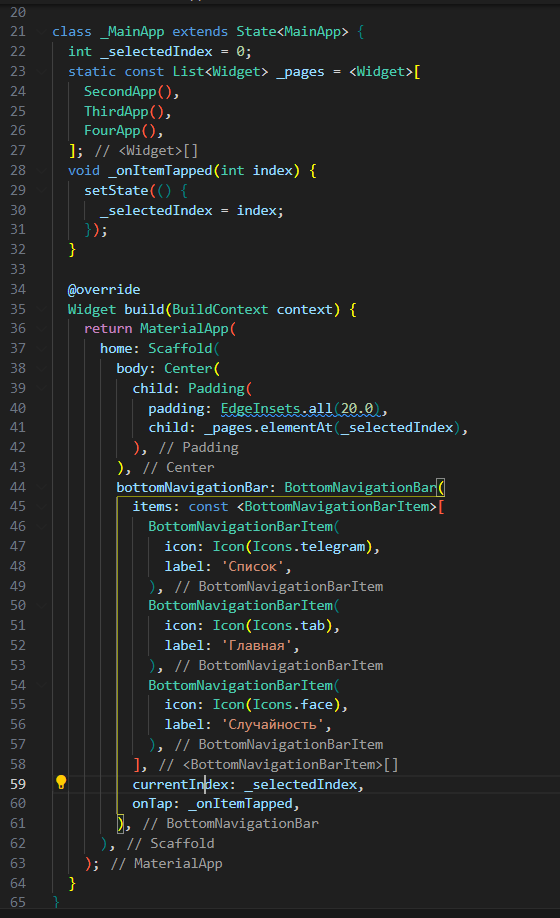


Рисунок 53. Главный класс.

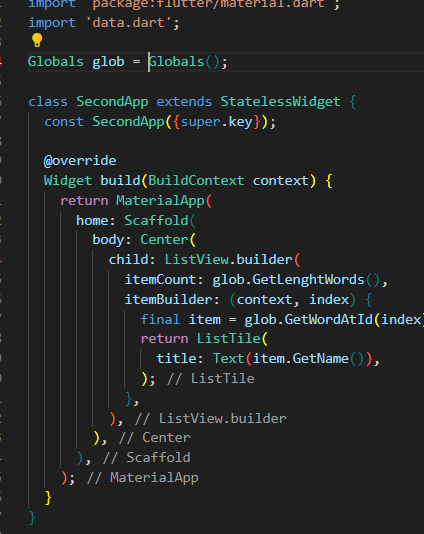


Рисунок 54. Второй класс.



Рисунок 55. Третий класс.

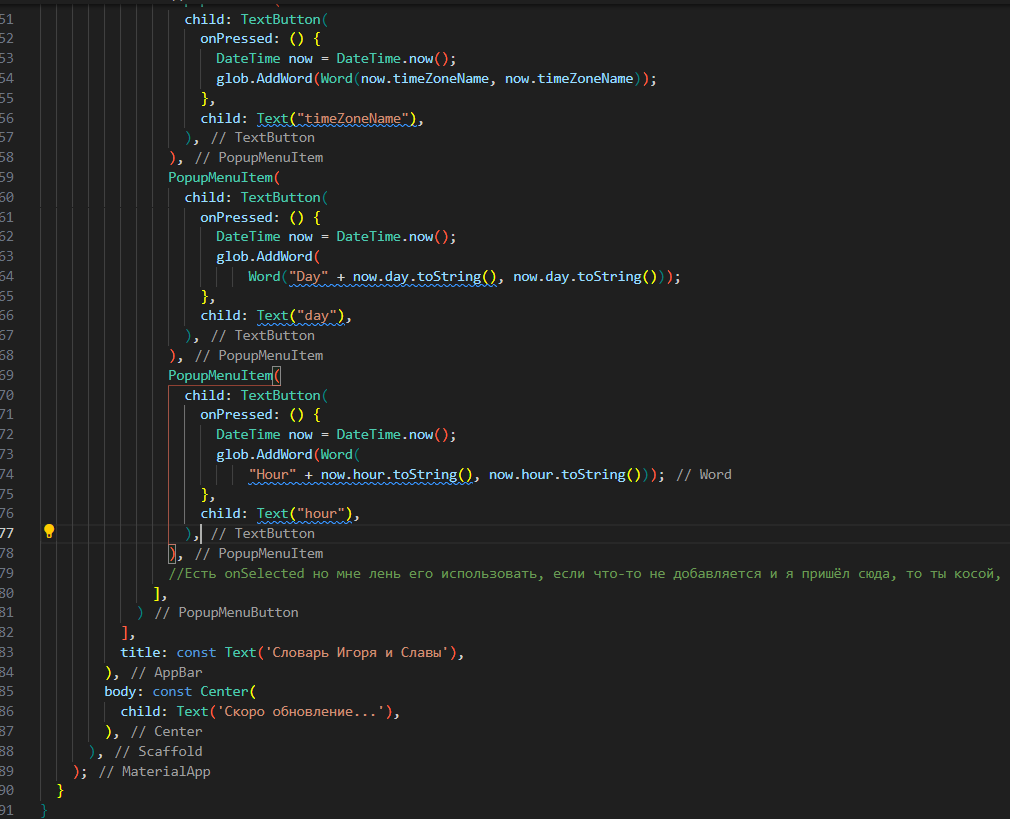


Рисунок 56. Третий класс продолжение.



Рисунок 57. Четвёртый класс.

Вывод: Поставленная цель оказалась сложно выполнимой на windows и не была реализована.

# **ПРАКТИЧЕСКАЯ РАБОТА №8**

Тема: Чистая архитектура.

Для оценки "5" минимум следует реализовать четыре экрана, а общее количество функций в репозитории должно составлять 5.

Ход работы:

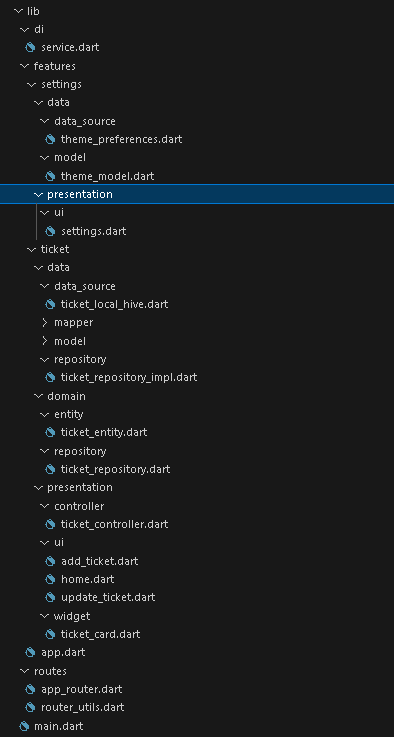


Рисунок 49 – Архитектура

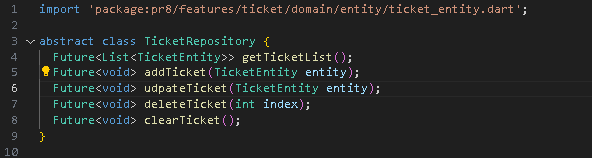


Рисунок 50 – Функции

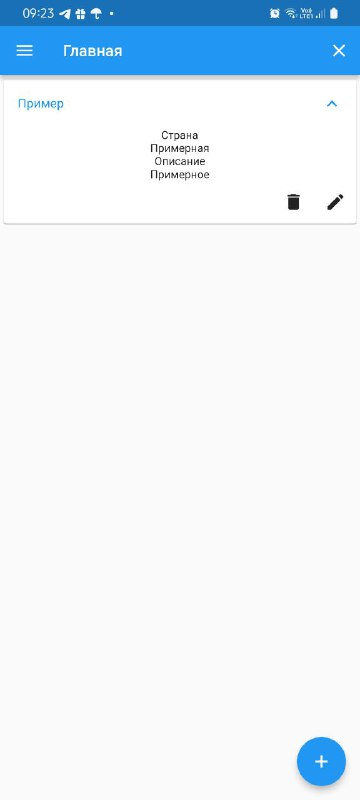


Рисунок 51 – Главная

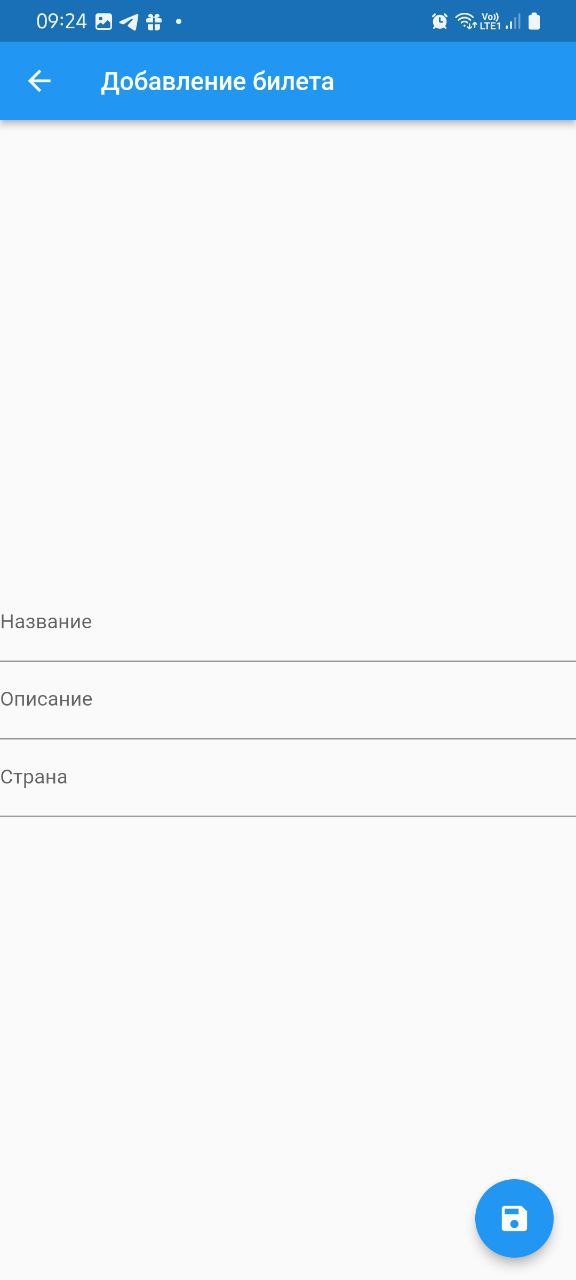


Рисунок 52 – Добавление билета

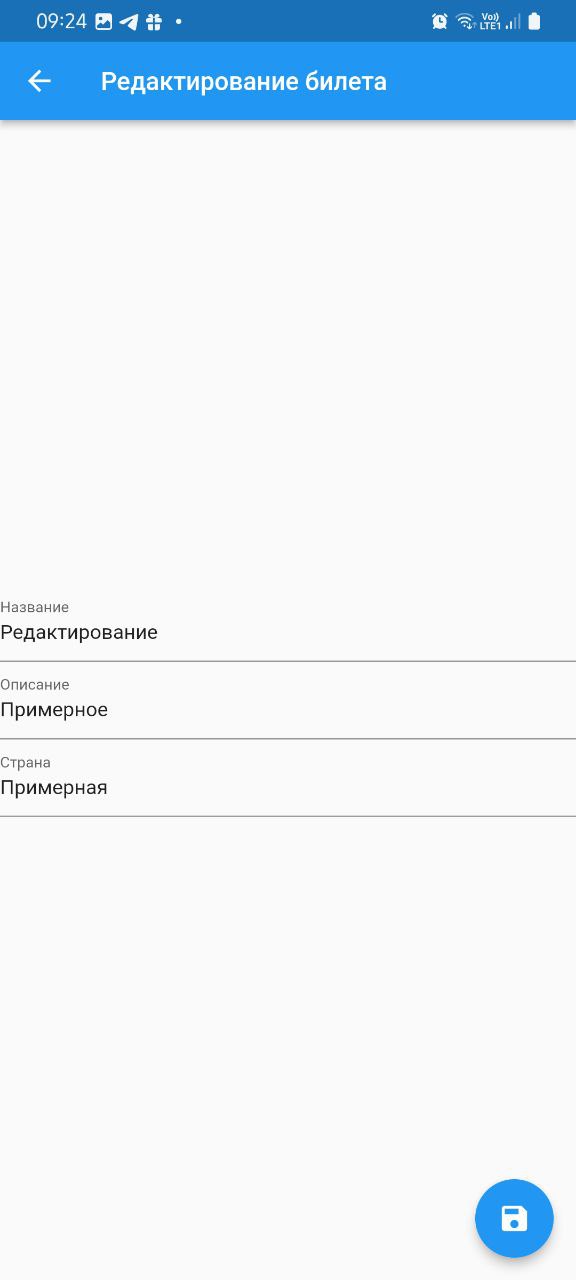


Рисунок 53 – Редактирование



Рисунок 54 – Смена темы

Вывод: создано приложение с чистой архитектурой, реализовано 4 экрана и 5 функций в репозитории.

# **ПРАКТИЧЕСКАЯ РАБОТА №9**

Тема: Работа с сетью.

На оценку 5:

Разработать приложение, следуя принципам чистой архитектуры. Для работы с сетью использовать пакет Dio. Создать класс interceptor, ответственный за добавление токена в заголовок, а также использовать interceptor от пакета pretty\_dio\_logger. Для управления зависимостями (DI) использовать пакет get\_it.

Ход работы:

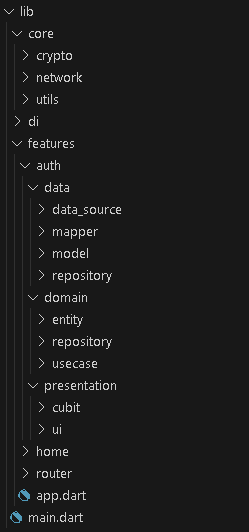


Рисунок 55 – Архитектура

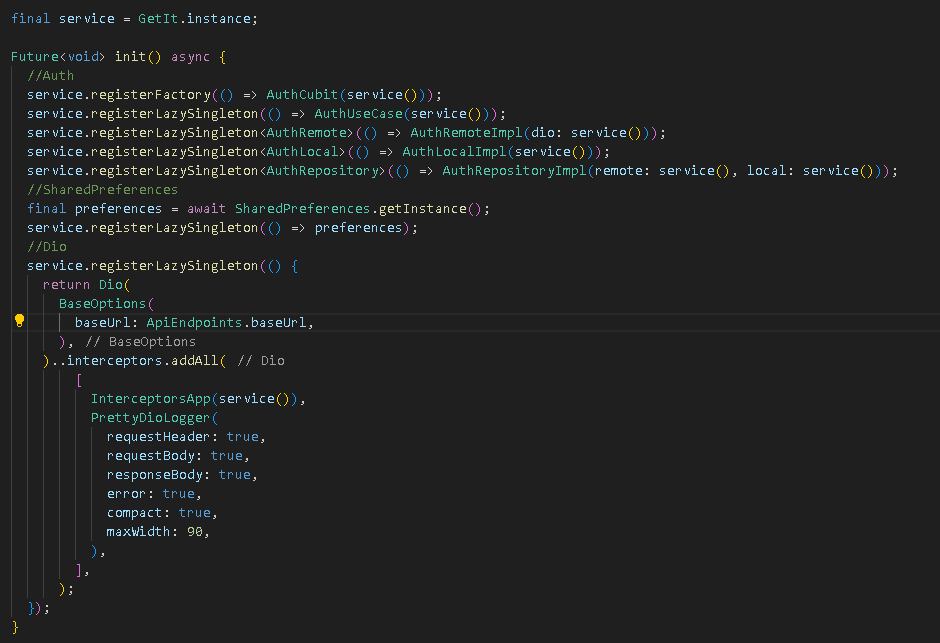


Рисунок 56 – GetIt

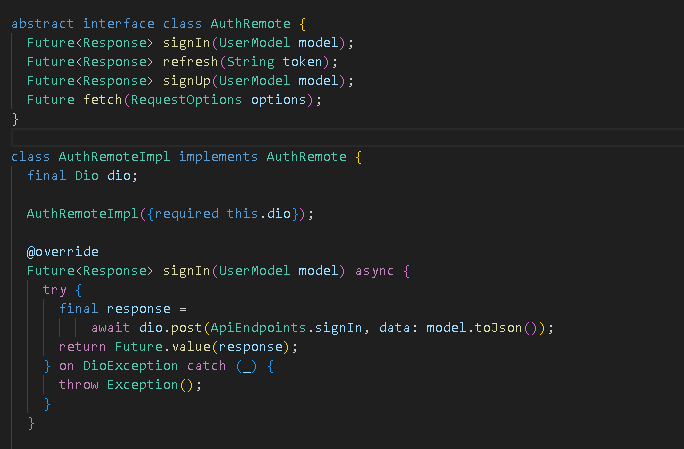


Рисунок 57 – Использование dio

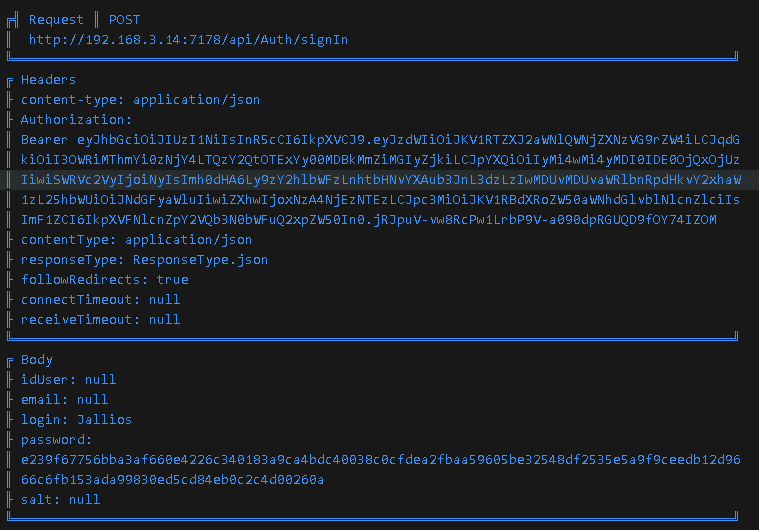


Рисунок 58 – DioLogger

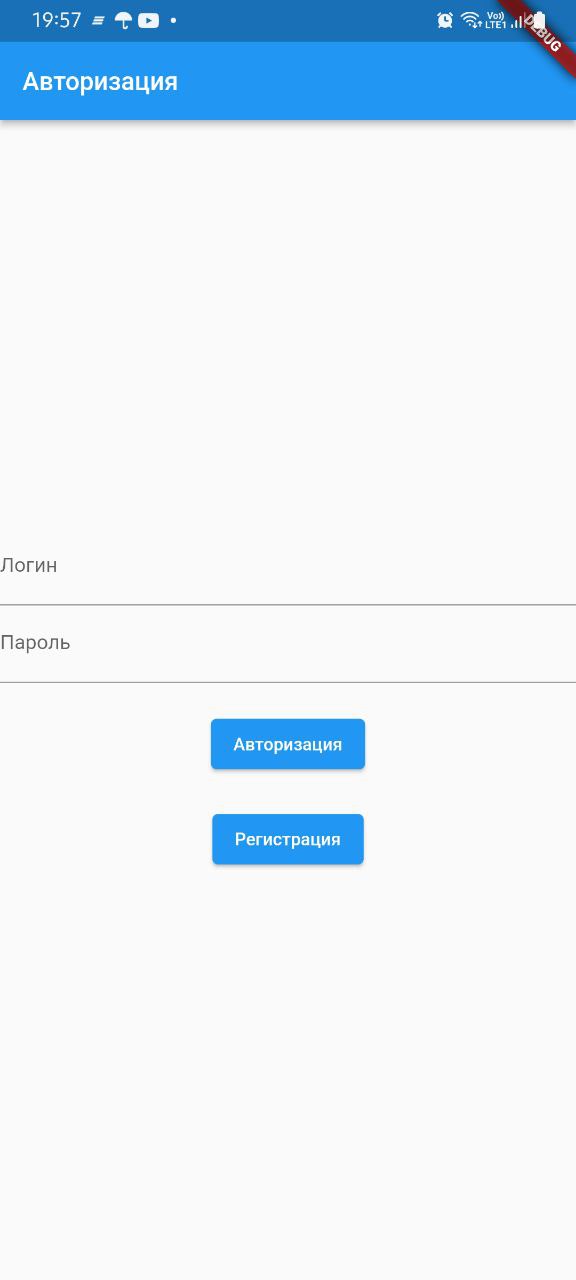


Рисунок 59 – Авторизация

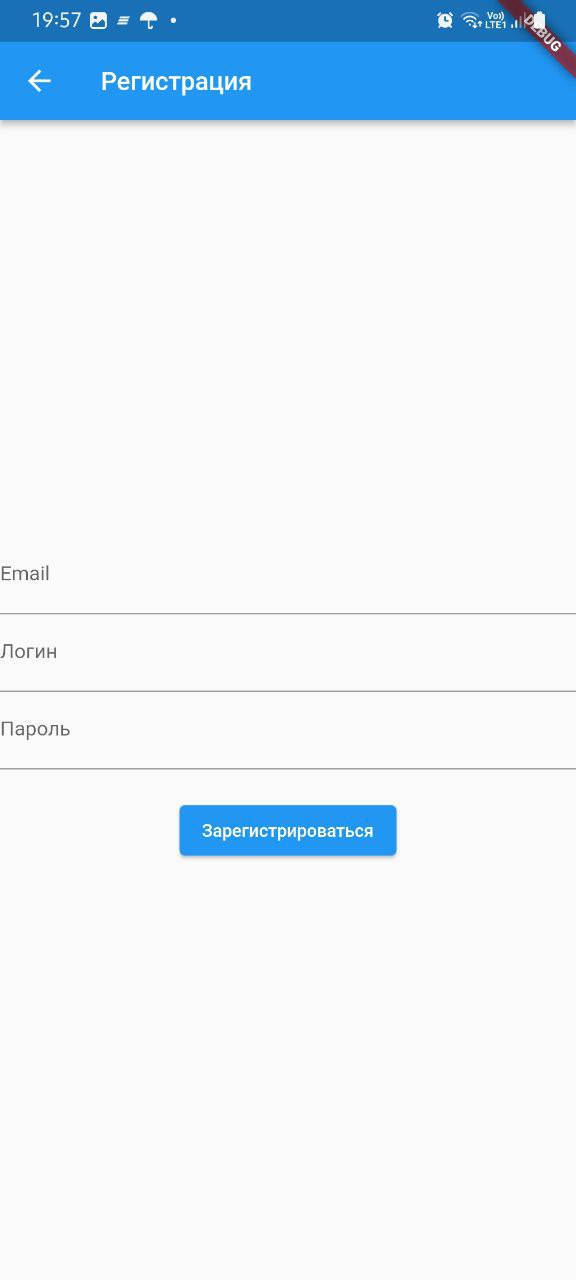


Рисунок 60 – Регистрация

Вывод: сделано приложение с чистой архитектурой и использованием dio.