УДК 621

ИСПОЛЬЗОВАНИЕ ПРИЛОЖЕНИЯ РАЗРАБОТАННОГО НА ПЛАТФОРМЕ FLUTTER ДЛЯ АВТОМАТИЗАЦИИ ПРОЦЕССА КОНТРОЛЯ ЗАДАЧ СОТРУДНИКОВ ПРОВАЙДЕРОВ Г.КЕМЕРОВО

Паданаев Н.С., Бахаревский А.О, студенты гр. ИТб-201, IV курс
Научный руководитель: Алексеева Г.А.
Кузбасский государственный технический университет
имени Т. Ф. Горбачева
г. Кемерово

Любая организация стремиться развиваться, расширяться и налаживать процессы в эффективное русло. Чем больше сотрудников, тем сложнее контролировать рабочие процессы даже хорошим руководителям. Это грозит ухудшением ситуации, стагнацией и падением как прибыли, так и общей работоспособности организации. Сроки выполнения задач могут растягиваться на дополнительные минуты, часы, дни и даже недели.

Для принятия правильных и эффективных решений хорошему руководителю требуется знать, где и в каком месте работа замедляется, простаивает или не выполняется. По каким причинам это может происходить:

- Сотрудник не получил задачу, в следствии чего не выполнял ее.
- Сотрудник получил задачу, но из-за слабой квалификации или иных причин не смог справится.
- Сотрудник получил задачу, но выполнял ее неоправданно долго.

Каждый сотрудник получает зарплату и в целях любой организации стоит организовать рабочие процессы с минимальными потерями времени специалистов и с максимальной отдачей для выполнения поставленных задач. При этом не стоит давить на работников, поставив высокий порог эффективности выполнения задач, из-за чего копится стресс и специалисты теряют работоспособность.

Еще в 2001 году Дэвид Аллен подсказал путь из водоворота горящих дедлайнов к продуктивности без стресса в книге «Как привести дела в порядок». Алгоритм его методики основан на разгрузке разума, где все важные дела должны быть организованы и записаны. Небольшие и срочные задачи немедленно закрыты, а больше и несрочные отложены на потом. Список задач время от времени нужно пересматривать.

Для отслеживания своей эффективности может помочь приложение с тайм-менеджментом задач. Это приложение позволяет следить за своей продуктивностью как сотруднику, улучшая свои результаты или замечать просадки, что может говорить об утомленности, так и руководителю, который видя время-затраты сможет лучше планировать задачи, перенаправлять сотрудников, помогать «залатывать дыры» в проблемных местах за счет времени, решения локальных проблем или найма дополнительных специалистов.

Приложение позволит автоматизировать процессы слежения за выполнением задач, слежением за временем, которое уходит на задачу каждым сотрудников, быстро получать отчеты.

В данной статье рассмотрим, как можно внедрить в рабочие процессы приложение для контроля выполнения задач сотрудниками, написанное на платформе flutter.

Приложение устанавливается на телефон с операционными системами Android или IOS, написано на платформе Flutter с использованием языка программирования Dart при помощи интегрированной среды разработки Visual Studio Code. Для локального хранилища (базы данных) будет использоваться пакет Hive, в котором будут сохраняться, храниться и загружаться задачи, аккаунты и иные данные.

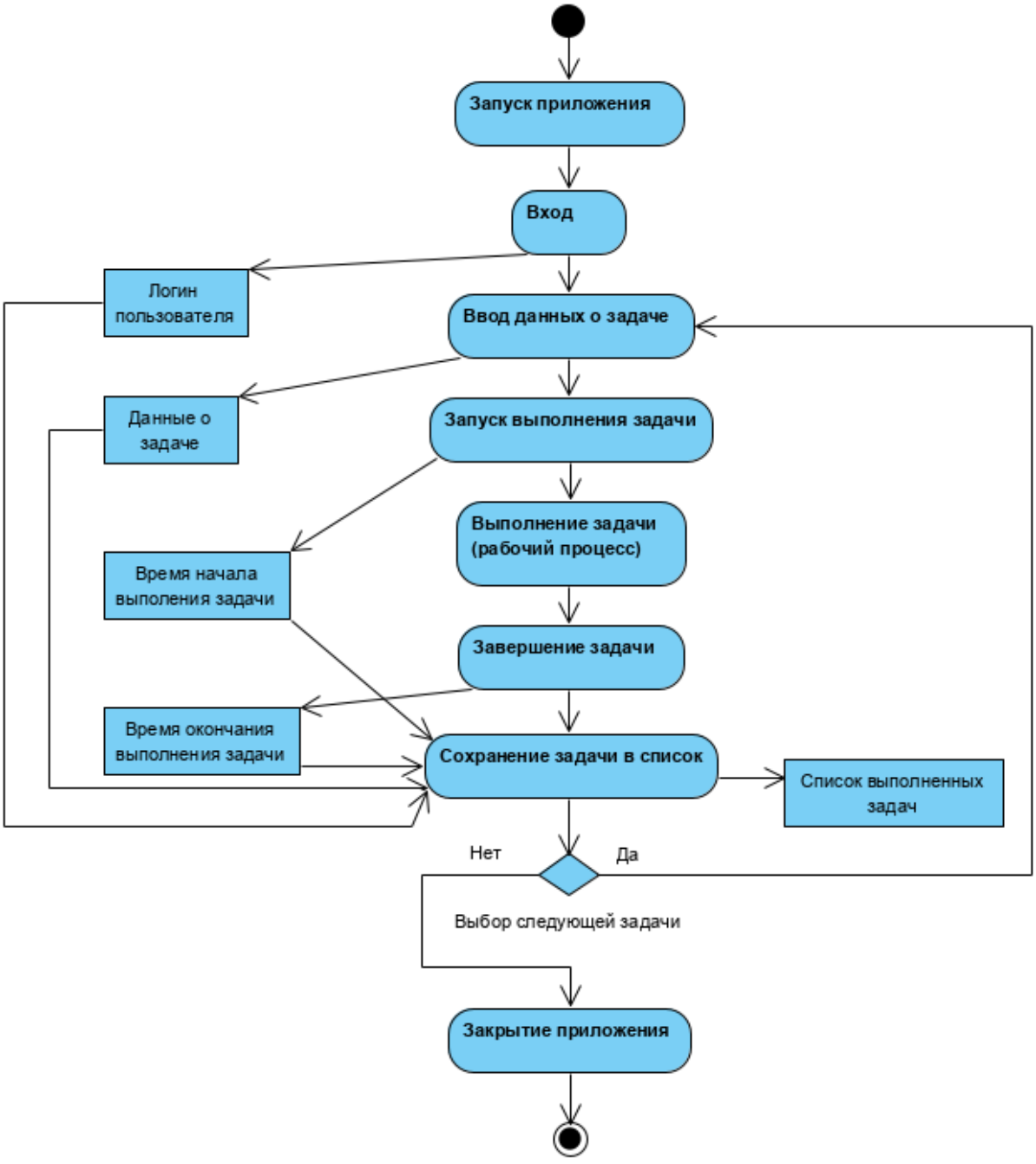

Рисунок 1 – Диаграмма деятельности процесса выполнения задач с использованием приложения

При использовании приложения, специалист вводит задачу в произвольной текстовой форме, после чего нажимает на кнопку «Начать задачу». Программа

запоминает данные и время начала. После выполнения работы, сотрудник может дополнить комментарий с задачей дополнительной информацией и нажимает на кнопку «Завершить задачу». Приложение запоминает время завершения работы, собирает все данные и сохраняет задачу в общий список. Далее сотрудник может начать новую задачу или выйти из приложения, завершив свой рабочий день.

Диаграмма деятельности процесса выполнения задач с использованием приложения показана на рисунке 1.

Защита данных пользователя, идентификация и аутентификация являются важным приоритетом в подобных приложениях. Аккаунт сотрудника регистрирует администратор организации, выдавая логин и пароль работнику. Чтобы войти в приложение, требуется правильно ввести логин и пароль. При несовпадении пары приложение попросит перепроверить данные и ввести еще раз. При вводе пароля символы скрываются для дополнительной защиты.

Диаграмма деятельности процесса аутентификации пользователя в приложении показана на рисунке 2.

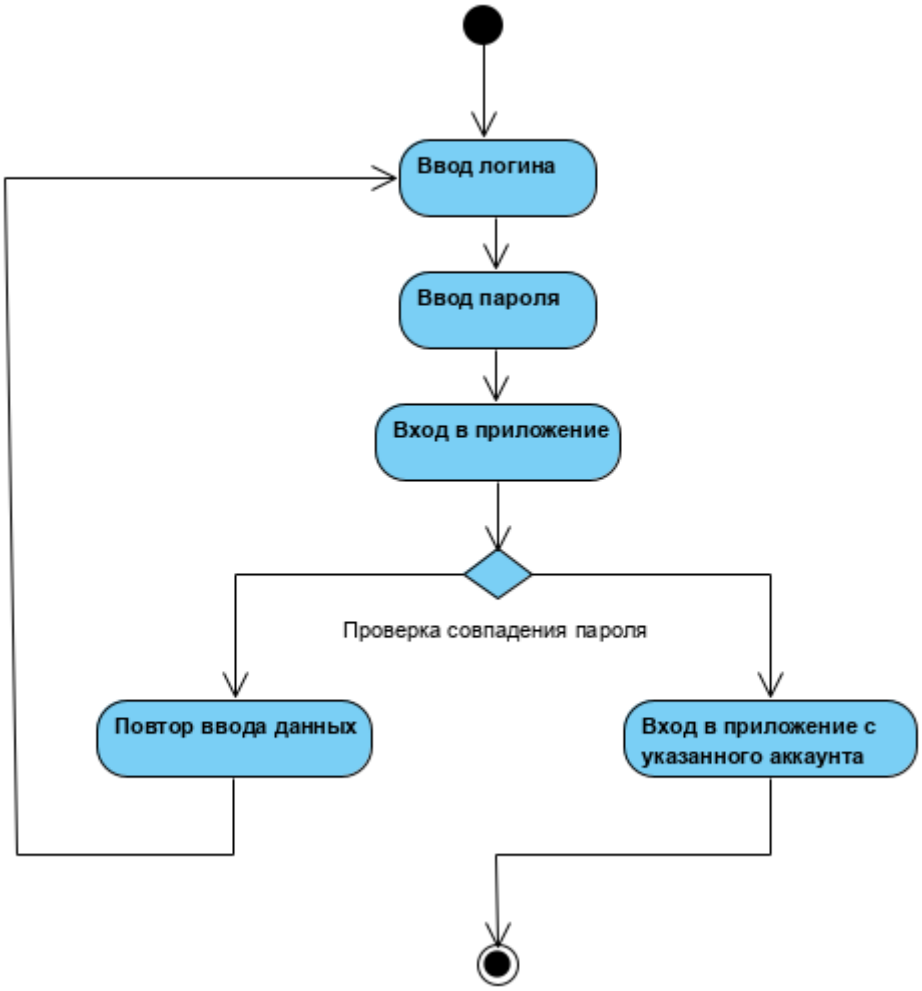

Рисунок 2 – Диаграмма деятельности процесса аутентификации пользователя

Программа написана на платформе flutter, что позволяет создавать и компилировать приложения на разные платформы, такие как Android, Ios, Web,

Windows, Mac, Linux. Flutter использует язык программирования Dart с собственным менеджером пакетов, несколькими компиляторами, транспилерами и синтаксическим анализатором. Основной особенностью Flutter разработки является древовидная структура проекта и кода, состоящего из виджетов. Все виджеты наследуются от класса Widget, имеют разнообразные параметры, такие как child, children, alignment, value и тд. С помощью параметров child и children можно вкладывать виджеты друг в друга, тем самым создавая дерево для построения макета.

Простые примеры виджетов:

- Align – указывает на позиционирование вложенного элемента относительно определенной стороны контейнера.
- Padding – указывает на отступы вложенного элемента от сторон контейнера.
- ConstrainedBox – задает параметры прямоугольной области, в которой размещается вложенный виджет.
- Container – виджет, объединяющий свойства многих виджетов, таких как Align, Padding, ConstrainedBox.
- Column и Row – располагают вложенные элементы вертикально или горизонтально.

Виджеты в основном бывают 2 типов:

1) StatelessWidget – неизменяемые виджеты с постоянным состоянием. Зависят только от параметром от родительских виджетов.
2) StatefulWidget – виджеты с изменяемым внутренним состоянием, имеющие собственное, независимое от родителей, поведение.

Пример простого StatelessWidget, служащего для запуска приложения, показан на рисунке 3:

```
9   class MainApp extends StatelessWidget {
10    const MainApp({super.key});
11
12    @override
13    Widget build(BuildContext context) {
14      return const MaterialApp(
15        home: LaunchScreen(),
16      ); // MaterialApp
17    }
18  }
```

Рисунок 3 – MainApp виджет для запуска приложения

StatefulWidget в сравнении со StatelessWidget имеет отдельное состояние State. Описание StatefulWidget из приложения на рисунке 4-5. В состоянии, кроме функции build, могут описываться методы:

- initState() – для инициализации состояния (первого создания виджета).
- didUpdateWidget() – служит для обновления виджета после изменения виджета, а именно параметров, приходящих от родителей.
- dispose() – удаление и освобождение слушателей, контроллеров и остальных объектов, требующих вызова метода dispose().

```
class LaunchScreen extends StatefulWidget {
  const LaunchScreen({
    super.key,
  });

  @override
  State<LaunchScreen> createState() => _LaunchScreen();
}
```

Рисунок 4 – описание StatefulWidget с созданием State

```
class _LaunchScreen extends State<LaunchScreen> with TickerProviderStateMixin {
  @override
  void initState() {
    super.initState();
    GlobalStorage.boxInit().then((_){
        Navigator.of(context).push(MaterialPageRoute(builder: (context) => const LoginScreen()),);
      }
    );
  }
  @override
  void dispose() async {
    await GlobalStorage.boxDispose();
    super.dispose();
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Padding(
        padding: const EdgeInsets.all(20.0),
        child: Center(
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: <Widget>[
              Text('Loading...',
                style: Theme.of(context).textTheme.titleLarge,
              ), // Text
              const SizedBox(height: 20,),
              const CircularProgressIndicator(),
            ], // <Widget>[]
          ), // Column
        ), // Center
      ), // Padding
    ); // Scaffold
  }
}
```

Рисунок 5 – описание State с использованием initState() и dispose()

Протестируем приложение на рисунках 6-7, войдя в свой аккаунт. Укажем задачу, завершим ее, после чего повторим данные действия еще с несколькими задачами, получив в итоге список задач.

Рисунок 6 – ввод логина и пароля

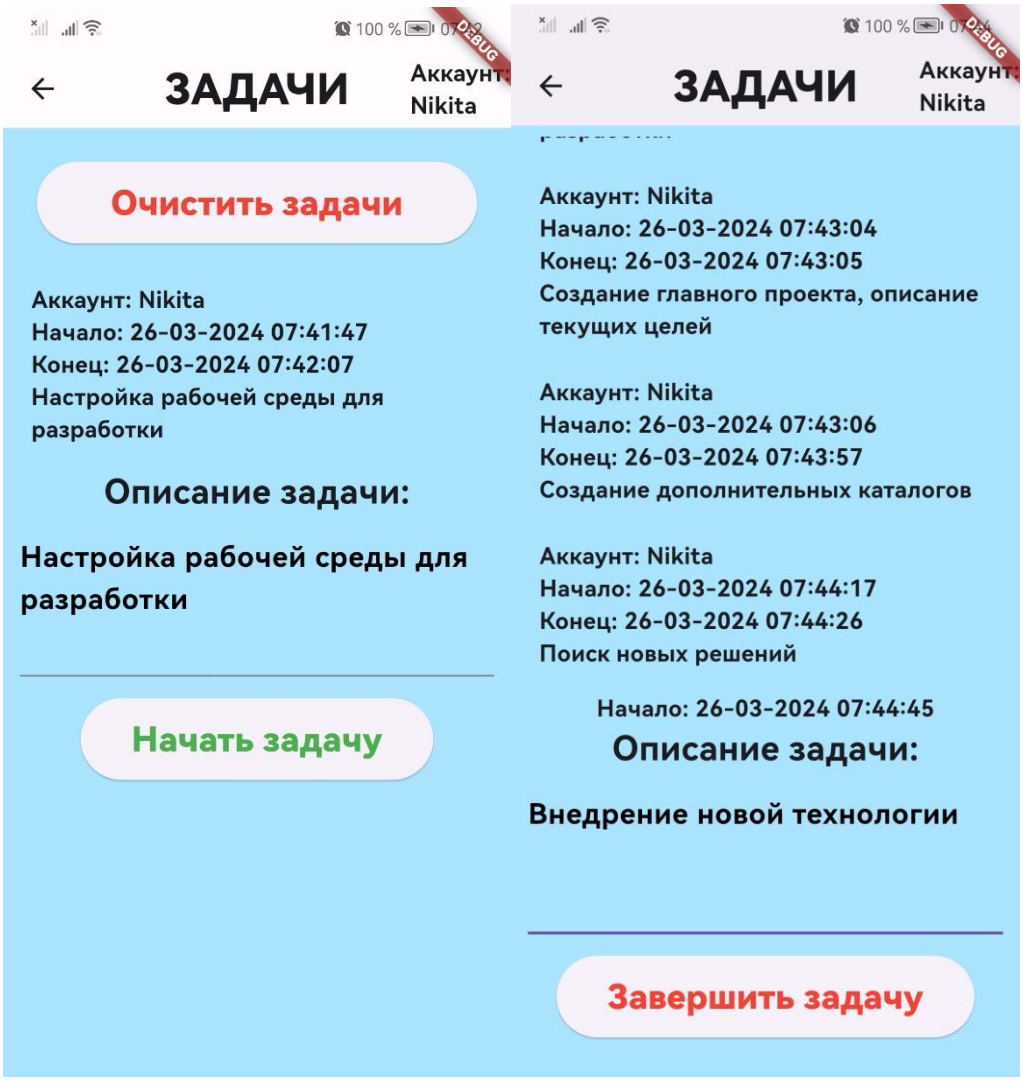

Рисунок 7 – создание задач и получение итогового списка

Благодаря внедрению приложения контроля задач в рабочие процессы сотрудники и руководители смогут следить за временем выполнения задач, отслеживая продуктивность и вовремя реагировать на возникающие статистические задержки, предпринимая соответствующие меры для повышения эффективности предприятия.

Список литературы:

1. Документация Flutter [Электронный ресурс] URL: https://flutter.dev/ (дата обращения: 25.03.2024)
2. Документация Flutter на русском: metanit.com [Электронный ресурс] URL: https://metanit.com/dart/flutter/1.1.php (дата обращения: 25.03.2024)
3. Документация Dart: metanit.com [Электронный ресурс] URL: https://metanit.com/dart/tutorial/1.1.php (дата обращения: 25.03.2024)
4. Моделирование на языке UML в среде Visual Paradigm 14. [Электронный ресурс] URL: http://sp.cs.msu.ru/ooap/exer2017.html#exer34 (дата обращения: 27.03.2024)