DOI 10.25699/SSSB.2019.27.37209

05.13.11

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ОТСЛЕЖИВАНИЯ МЕНТАЛЬНОГО СОСТОЯНИЯ ЧЕЛОВЕКА

С.В. Букунов, П.А. Широбокова

ФГБОУ ВО «Санкт-Петербургский государственный архитектурно-строительный университет», г. Санкт-Петербург

Целью работы является разработка и реализация мобильного приложения, позволяющего пользователю самостоятельно контролировать свое ментальное состояние. Для хранения информации о состоянии и действиях пользователя разработана реляционная база данных. Взаимодействие приложения с базой данных осуществляется с помощью системы управления базами данных SQLite. Для работы с приложением создан многооконный графический интерфейс. В настоящий момент разработанное приложение по некоторым функциональным возможностям не имеет аналогов на рынке.

Ключевые слова: ментальное здоровье, мобильное приложение, база данных, программирование.

ВВЕДЕНИЕ

Развитие технологий в XXI веке помимо прогресса создает и множество новых вызовов, требующих решения. Главным образом выделяют экологические катастрофы, развитие новых болезней и демографический кризис. Но существует еще одна проблема, которая не кажется устрашающей с первого взгляда, однако уже наносит удары по, например, экономике — это ухудшение ментального здоровья людей на фоне изменения мира коммуникаций, распространения интернета и социальных сетей и других перемен в образе жизни человечества. Имеются в виду заболевания по типу неврозов, депрессии, а также повседневный стресс.

Наличие такого рода проблем подтверждается следующими данными:

− из более чем четырех сот пятидесяти миллионов человек, страдающих ментальными расстройствами, лишь одна треть получает помощь [1];

− в 2017 году в России совершили суицид 20 278 человек;

− по прогнозам Всемирной Организации Здравоохранения к 2020 году психические расстройства войдут в первую пятерку болезней человечества;

− на 100 000 человек приходится только один психиатр в более чем половине стран мира;

− снижение производительности труда, связанное с развитием тревожных расстройств, обходится для глобальной экономики в $1 трлн.

Для лечения ментального здоровья можно обратиться к специалистам: психиатрам, психотерапевтам и психологам. Но в России далеко не каждый может позволить себе оплату многократных сеансов, сказывается и отсутствие общей культуры поддержки ментального здоровья. Многие не считают свой недуг серьезной проблемой, не желая выглядеть слабыми и боясь осуждения.

Тем не менее, улучшать свое состояние самостоятельно возможно. Для этого есть медитации, различные физические практики и тренировки осознанности. Однако людям в подавленном состоянии тяжело дается самоконтроль, особенно в век перенасыщения информацией. Необходимы простые и понятные инструменты.

Благодаря широкой распространенности Интернета и мобильных устройств в развивающихся странах (80,9% населения на 2017 год), отслеживание и контроль собственных состояний возможно осуществлять с помощью приложения, установленного на телефоне.

ПОСТАНОВКА ЗАДАЧИ

После проведения исследования рынка мобильных приложений было выяснено, что на российском рынке направление отслеживания настроений не развито. Поэтому цель работы заключается в разработке приложения с многооконным пользовательским интерфейсом, позволяющего пользователю самостоятельно контролировать свое ментальное состояние, в частности:

– оценивать свое настроение;

– записывать выполненные действия;

– добавлять детальное словесное описание в заметки;

– оценивать каждое совершенное действие с помощью количественной характеристики;

– просматривать статистику по своему состоянию;

– просматривать динамику состояния в графическом виде и др.

ИСПОЛЬЗУЕМЫЕ ТЕХНОЛОГИЧЕСКИЕ РЕШЕНИЯ

Разработанное приложение предназначено для работы на мобильных устройствах с операционной системой Android [2]. Для хранения данных пользователя была создана реляционная база данных, расположенная в облачном хранилище Firebase от компании Google. Взаимодействие приложения с базой данных

реализовано с помощью системы управления базами данных SQLite [3]. Для обеспечения более надежного доступа к базе данных при использовании SQLite использовалась библиотека Room [4]. Программный код написан в объектно-ориентированном стиле на языке Java [5]. Для разделения процессов, работающих с базой данных и с интерфейсом, в приложении был реализован принцип многопоточности. Для реализации многопоточности использовался класс асинхронной работы AsyncTask [6]. Разработка приложения осуществлялась в среде Android Studio.

ОБЩАЯ СХЕМА РАБОТЫ ПРИЛОЖЕНИЯ

Для обеспечения удобной работы пользователя в приложении был реализован многооконный графический пользовательский интерфейс. Основные окна (страницы) приложения и их взаимосвязи представлены на рис. 1. Стрелки на схеме демонстрируют возможные переходы пользователя между страницами приложения. Помимо этого, к трем основным окнам (окну выбора настроений, окну с выбором графиков и окну с просмотром существующих записей) можно получить доступ через нижнее навигационное меню приложения.

Рис. 1. Общая схема работы приложения

Ниже дано краткое описание основных страниц.

Страница отметки настроений – главная страница приложения, поскольку именно она отображается при запуске приложения. Предназначена для выбора текущего настроения из пяти возможных. Каждое настроение отображается в виде иконки, загружаемой из таблицы базы данных с соответствующим поясняющим текстом.

После выбора настроения пользователь перенаправляется на страницу выбора дополнительных параметров. Выбранное настроение отображается в самой верхней части окна. Из списка действий, загруженных из базы данных, возможно выбрать последние выполненные действия. Длительное нажатие на иконку действия спровоцирует появление счетчика – это необходимо для указания количественных характеристик. При указании количества иконка действия станет черной, а в целом при выборе действия фон иконки окрасится в синий цвет. Ниже в окне есть поле для добавления заметок, нажатие на которое приведет к отображению соответствующей страницы. Кнопка «сохранить» сохраняет все введенные параметры в базу данных, считывая текущие время и дату.

Страница добавления заметок состоит из текстового поля и кнопки «сохранить», нажатие на которую вернет пользователя на предыдущее окно выбора параметров. Введенная текстовая запись отобразится в окне выбора.

Страница с представлением выбора графиков и статистических показателей содержит четыре иконки, нажатие на которые приведет к открытию соответствующих им окон.

Страница с графиком зависимости настроения от времени содержит этот график с возможностью увеличения его для просмотра статистики на более мелком временном промежутке.

Страница с графиком зависимости настроения от времени по действию содержит переключатель со списком всех действий. Ниже расположена диаграмма. При выборе действия диаграмма обновится.

Страница с отображением средних настроений по всем действиям содержит список, в каждом пункте которого содержится иконка, соответствующая действию, название действия и среднее настроение.

Страница с показателями количественных характеристик для каждого действия содержит список с отображением иконок, названий действий и соответствующим им суммам всех количественных характеристик, введенных пользователем.

На странице с отображением записей по дате есть текстовое поле для показа выбранной даты, кнопка, вызывающая окно с календарем, и список всех записей, показывающий время добавления записи и иконку настроения.

При нажатии на пункт записи пользователь переходит на страницу с подробным отображением данных для выбранной записи. В верхней части окна показаны дата и время записи, ниже указанное настроение, далее выбранные действия и, если были выбраны, количественные характеристики. В нижней части окна отображаются заметки.

В окне выбора даты для отображения записей открывается календарь, нажатие на дату в котором закроет текущее окно и переключит внимание пользователя на предыдущее окно.

БАЗА ДАННЫХ

Для хранения вводимой пользователем информации была создана реляционная база данных. ER-модель (Entity-Relationship model) базы данных представлена на рис. 2.

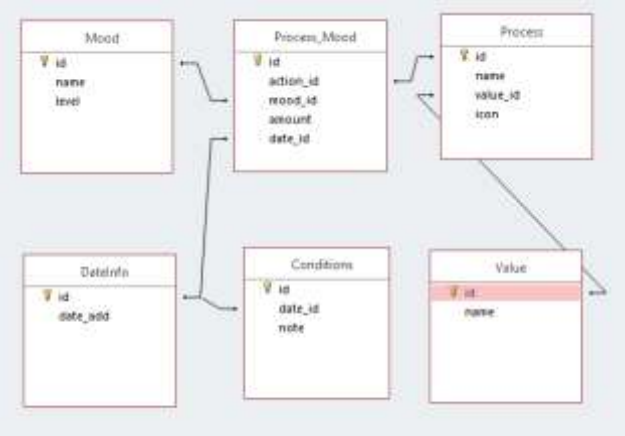

Рис. 2. ER-модель базы данных

Ниже дано краткое описание основных таблиц.

Таблица Mood отвечает за сущность «настроения». Первичным ключом служит поле id, в колонке name указываются названия настроений (ужасное, плохое, среднее, хорошее, отличное), в поле level указывается уровень настроения по шкале от одного до пяти.

Таблица Process представляет сущность «действия». В поле id указывается первичный ключ – идентификационный номер действия, в поле name – название действия (например, спорт, учеба, работа), в поле icon – название иконки для отображения в программе. Иконки загружаются в проект в отдельную папку, после чего извлекаются по имени. Поле value_id необходимо для связи с таблицей Value, чтобы при отображении количественных характеристик указывать значение единицы измерения.

Таблица Process_Mood служит как промежуточная таблица между Process и Mood для реализации связи многие ко многим. После добавления новой записи идентификационные номера обеих сущностей добавляются в эту таблицу вместе со значением количественной характеристикой действия, которая указывается в поле amount. Поле date_id необходимо для связи с таблицей DateInfo, которая предоставляет список всех дат, которые использовались при добавлении записей.

Таблица DateInfo имеет поле id – идентификационный номер и поле date_add – саму дату. Вынесение даты в отдельную таблицу необходимо для более точной работы с датами, ведь при обращении непосредственно к дате на всех этапах запросов пришлось бы постоянно конвертировать строковый формат SQLite в форматы Date и Calendar языка Java.

Таблица Conditions содержит поле id – идентификационный номер, поле date_id, связывающее сущность с таблицей DateInfo, и поле Note, которое отвечает за хранение заметок. Вынесение хранения заметок в отдельную таблицу сделано для того, чтобы при дублировании записей в таблице Process_Mood не происходило дублирования заметок, ведь иногда они могут содержать большое количество символов и занимать больше памяти.

Таблица Value хранит в себе единицы измерения, которые присваиваются каждому действию из таблицы Process. Среди единиц измерения есть, например, часы, километры, штуки, страницы.

РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ

Для обеспечения надежного доступа к базе данных при использовании SQLite в данной работе используется библиотека Room [4].

Room – это фреймворк, устанавливаемый дополнительно в Android Studio в разделе зависимостей Gradle и представляющий собой абстрактную «обертку» над стандартами SQLite.

В основу архитектуры библиотеки Room входят три компонента: Entity, DAO и Database. Однако для большего удобства предлагается использовать также классы ViewModel и Repository [4]. Entity – это специальный класс, описывающий таблицу в базе данных. Программная реализация Entity для таблицы Process_Mood приведена на рис. 3. В классе реализовано представление пяти полей таблицы: id, action_id, mood_id, amount и date_id.

```
@Entity
public class Process_Mood implements Serializable {

    @PrimaryKey(autoGenerate = true)
    public long id;
    public long action_id;
    public long mood_id;
    public long amount;
    public long date_id;

    public long getId() { return id; }

    public void setId(long id) { this.id = id; }

    public long getAction_id() { return action_id; }

    public void setAction_id(long action_id) { this.action_id = action_id; }

    public long getMood_id() { return mood_id; }

    public void setMood_id(long mood_id) { this.mood_id = mood_id; }

    public long getAmount() { return amount; }

    public void setAmount(long amount) { this.amount = amount; }

    public long getDate_id() { return date_id; }

    public void setDate_id(long date_id) { this.date_id = date_id; }
}
```

Рис. 3. Реализация @Entity таблицы Process_Mood

Для реализации самого доступа к данным используется интерфейс DAO, который отвечает за манипулирование данными, хранящимися в таблицах.

В интерфейсе DAO описываются SQL запросы, которые будут вызываться в дальнейшем коде программы. Для вставки и удаления данных в Room реализованы готовые аннотации @Insert и @Delete. Для выборки и в целом выполнения любых запросов создана аннотация @Query, после котором записывается нужный SQL запрос.

На рис. 4 приведен пример реализации DAO таблицы Process_Mood.

```
@Dao
public interface Process_MoodDao {
    @Query("SELECT * FROM process_mood")
    List<Process_Mood> getAll();

    @Query("SELECT * FROM process_mood WHERE action_id=:act_id")
    List<Process_Mood> getAllByProId(long act_id);

    @Query("SELECT * FROM process_mood WHERE date_id=:date_id")
    List<Process_Mood> getAllDate(long date_id);

    @Query("SELECT * FROM process_mood GROUP BY date_id")
    List<Process_Mood> getDateMood();

    @Query("SELECT * FROM process_mood")
    Process_Mood[] getAllArray();

    @Insert
    void insert(Process_Mood process_mood);

    @Insert
    void insert(List<Process_Mood> process_moods);
}
```

Рис. 4. Реализация @DAO таблицы Process_Mood

В данном коде описываются основные запросы, используемые при работе с таблицей Process_Mood. Как можно заметить, в интерфейсе присутствует разнообразная выборка данных через аннотацию @Query: запросы меняются в зависимости от принимаемых аргументов и типов возвращаемых значений.

Класс Database служит для создания и обновления версий базы данных. Класс объявляется аннотацией @Database, должен быть абстрактным и наследовать класс RoomDatabase. В параметрах аннотации указываются все сущности, участвующие в базе данных и версия базы данных. Также в этом методе нужно описать абстрактные методы для работы с объектами DAO.

На рис. 5 представлена стандартная реализация абстрактного класса Database, в котором единожды создается база данных, и при последующем к ней обращении сущностей будет осуществляться проверка, при которой статическая переменная INSTANCE будет индикатором загруженности или не загруженности базы данных.

Рис. 5. Реализация абстрактного класса Database

РЕАЛИЗАЦИЯ МНОГОПОТОЧНОСТИ

В архитектуре Android приложений есть важная особенность, которую необходимо учитывать при манипулировании с данными. Ни в коем случае нельзя работать с базой данных из главного потока – потока интерфейса. Это взаимодействие может повлечь за собой длительную задержку из-за большого объема данных, что приведет к зависанию приложения. Если обращаться к базе данных из главного потока, приложение не запустится и выдаст ошибку.

Для обхода этой особенности в языке Java существует многопоточность. Многопоточность позволяет перенаправлять более трудоемкие и долгие задачи на отдельные потоки и выполнять несколько задач сразу.

В Android есть несколько возможностей для реализации многопоточности. В данной работе для работы с базой данных был выбран такой класс асинхронной работы, как AsyncTask [6].

ОСНОВНЫЕ РЕЗУЛЬТАТЫ

Внешний вид главного окна разработанного приложения представлен на рис. 6. На экране данной страницы отображаются пять иконок, соответствующих пяти настроениям с подписями. Внизу отображается навигационная панель.

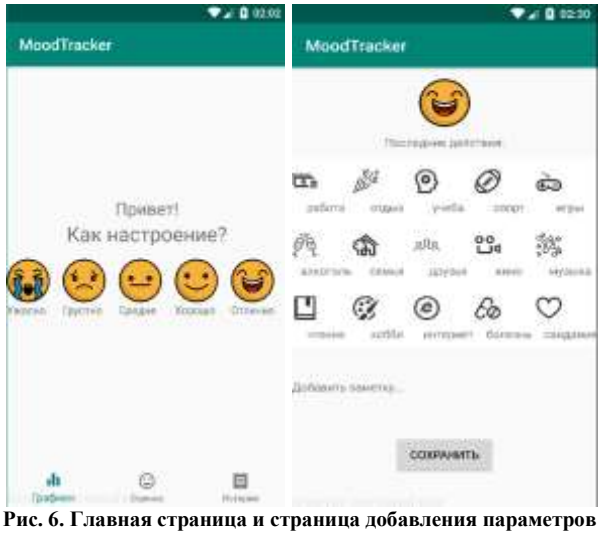

Рис. 6. Главная страница и страница добавления параметров

На странице добавления параметров, внешний вид которой представлен на рис. 6, реализовано принятие данных с предыдущей страницы. В результате в верхней части окна отображается сделанный ранее выбор настроения в виде соответствующей иконки.

В центральной части страницы расположен список возможных действий, ниже которого находится текстовое поле для добавления заметки, при нажатии на которое откроется соответствующая страница. После нажатия на кнопку «сохранить» программа устанавливает текущие время и дату и добавляет введенные пользователем данные в соответствующие таблицы базы данных.

На рис. 7 показан внешний вид двух окон, отображающих графики: окна с графиком зависимости настроения от времени и окна с диаграммой зависимости настроения от времени по выбранному фильтру действия.

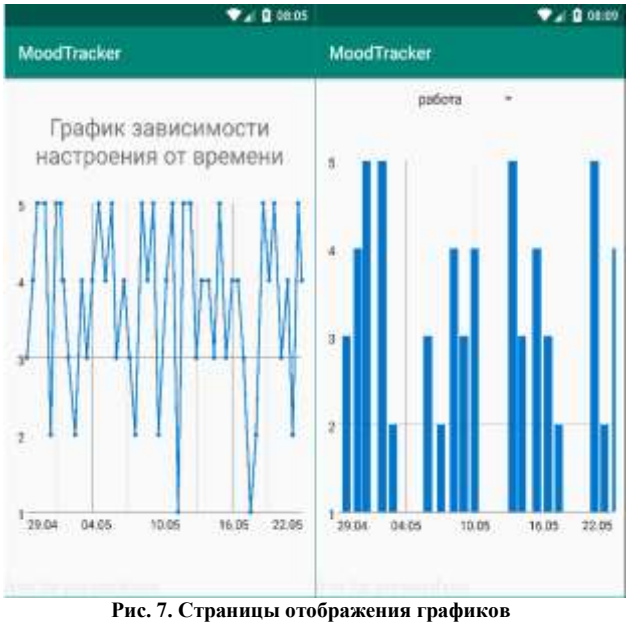

Рис. 7. Страницы отображения графиков

Для отображения суммы всех количественных характеристик по действиям и средних показателей настроений по действиям были созданы две страницы со списками. Внешний вид окон показан на рис. 8.

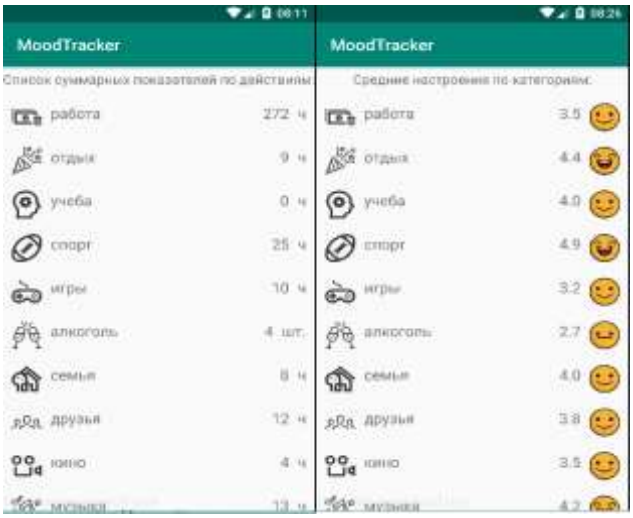

Рис. 8. Страницы отображения количественных характеристик

ЗАКЛЮЧЕНИЕ

В настоящее время активному пользователю смартфона и прочих гаджетов бывает довольно сложно сконцентрировать свое внимание на важных задачах. Помимо этого, гаджеты и распространенность Интернета меняют человеческую психику, навязывая ложные ценности и вводя человека в подавленное состояние.

В качестве одного из методов решения вышеперечисленных проблем психологи рекомендуют проводить анализ своих дней, поведения и настроений для рефлексии и повышения осознанности, что влечет улучшение качества жизни.

Разработанное Android приложение позволяет в любой момент времени сделать отметку о своем состоянии с указанием подробных показателей, а впоследствии на графиках и в списках увидеть всю динамику настроений. Поиск в истории записей по дате позволяет увидеть полный отчет о каждом дне и каждой записи.

Разработка приложения на операционной системе Android позволяет распространить приложение для большего количества людей из-за доминирующего положения на рынке данной операционной системы.

Использование библиотеки Room исключило возможные ошибки при работе с базой данных в Android. Многопоточный доступ к данным, благодаря AsyncTask, позволяет приложению работать гибко и не зависеть от ожидания ответа базы данных.

На момент создания приложения его аналогов по некоторым функциям найдено не было.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. World health report // World Health Organization [Электронный ресурс]. – Режим доступа: https://www.who.int/whr/2001/media_centre/press_release/en/.

2. Build anything on Android // Android Development Documentation [Электронный ресурс]. – Режим доступа: https://developer.android.com/.

3. SQLite // SQLite Development Documentation [Электронный ресурс]. – Режим доступа: https://www.sqlite.org/docs.html.

4. Android Room with a View – Java // Android Room Documentation [Электронный ресурс]. – Режим доступа: https://codelabs.developers.google.com/codelabs/android-room-with-a-view/index.html.

5. Шилдт, Г. Java 8: руководство для начинающих [Текст] / Г. Шилдт. 6-е изд. − М.: Вильямс, 2015. − 712 с.

6. Климов, А. Android: AsyncTask / А. Климов [Электронный ресурс]. – Режим доступа: http://developer.alexanderklimov.ru/android/theory/asynctask.php.

Букунов Сергей Витальевич – кандидат технических наук, доцент кафедры информационных технологий, Санкт-Петербургский государственный архитектурно-строительный университет, тел. 8(921)3213162, e-mail: sergeybukunov@yandex.ru.

Широбокова Полина Андреевна – студент 4 курса, Санкт-Петербургский государственный архитектурно-строительный университет, тел. 8(900)6523831, e-mail: izhevsk07@gmail.com.

DEVELOPMENT OF MOBILE APPLICATION FOR TRACKING OF THE HUMAN MENTAL CONDITION

S.V. Bukunov, P.A. Schirobokova

Saint-Petersburg State University of Architecture and Civil Engineering, Saint-Petersburg

Abstract – The aim of the work is the development and implementation of a mobile application that allows the user to control their mental state themselves. A relational database to store user status and actions has developed. The interaction of the application with the database carried out using the SQLite database management system. A multi-window graphic interface for working with the application has developed. The developed application has no analogues for some functionality in the present time.

Index terms: mental health, mobile application, database, programming.

REFERENCES

1. World health report // World Health Organization. – The access mode: https://www.who.int/whr/2001/media_centre/press_release/en/.
2. Build anything on Android // Android Development Documentation. – The access mode: https://developer.android.com/.
3. SQLite // SQLite Development Documentation. – The access mode: https://www.sqlite.org/docs.html.
4. Android Room with a View – Java // Android Room Documentation. – The access mode: https://codelabs.developers.google.com/codelabs/android-room-with-a-view/index.html.
5. Schildt, G. Java 8: руководство для начинающих [Text] / G. Schildt. 6th ed. − M.: Williams, 2015. − 712 p.
6. Klimov, A. Android: AsyncTask / A. Klimov. – The access mode: http://developer.alexanderklimov.ru/android/theory/asynctask.php.

Bukunov Sergey Vitalievich – candidate of technical sciences, associate professor of the chair informatics technologies, Saint-Petersburg State University of Architecture and Civil Engineering, (921)3213162, e-mail: sergeybukunov@yandex.ru.

Schirobokova Polina Andreevna – student of 4 course, Saint-Petersburg State University of Architecture and Civil Engineering, (900)6523831, e-mail: izhevsk07@gmail.com.