FRI-1.414-1-MIP-05

MOBILE APP MOUNTAIN TRACK [5]

Prop. Assistant Valentin Velikov, PhD
Department of Informatics and Information Technologies,
University of Ruse "Angel Kanchev"
Tel.: +359 886 011 544
E-mail: vvelikov@ami.un-ruse.bg

Bac. Emil Voykov
Department of Informatics and Information Technologies,
University of Ruse "Angel Kanchev"
Tel.: +359 888 702 200
E-mail: emil.voykov.th@gmail.com

Abstract: *The article discusses some Android-related technologies for working with the Internet, GPS, local database (MySQL), Google Maps API, threads, dynamic structures, etc. The purpose of the created practical application is to develop, master and test the relevant technologies.*

It was necessary for these techniques and programming skills to be researched and studied in connection with the creation and development of a lecture course.

"Mountain Track" - the idea of the developed practical system is to assist tourists in planning (by loading maps and ready-made routes), navigating and documenting (by creating tracks, photos and markers) routes in mountainous areas.

Key words: *Mountain track, Android, GPS navigation, Software Engineering, Information systems, Tourism.*

INTRODUCTION

In the last decade, the world has become increasingly mobile and technologically advanced, and the need for digital solutions in tourism is growing rapidly. Mobile applications, combining cartographic information and navigation capabilities, are becoming a standard in serving tourists - both for individual trips and organized excursions.

In this context, travel agencies and hotels, striving for a modern, competitive service offering, are looking for ways to provide innovative and practical solutions for their clients. This is where the development of a mobile application comes in, allowing tourists not only to navigate their surroundings, but also to interact with them in a new, more personal and technologically enriched way.

The goal of the project is to develop a mobile application for Android that assists tourists in planning, navigating and documenting routes in mountainous areas using GPS, Google Maps and a local database. The development is based on Java, Android SDK, Google Maps API and SQLite database.

EXPOSITION

The task was to create an application with the following capabilities:

1. Interaction with maps:
   - creating a map of the region from a photo (jpg-map) or Google maps integration;
   - loading a map (of a selected region or country);

[5] Докладът е представен на конференция на Русенския университет на 24 октомври 2025 г. в секция „Математика, информатика и физика“ и отразява резултати от работата по проект № 25-ФПНО-04, финансиран от фонд „Научни изследвания“ на Русенския университет.

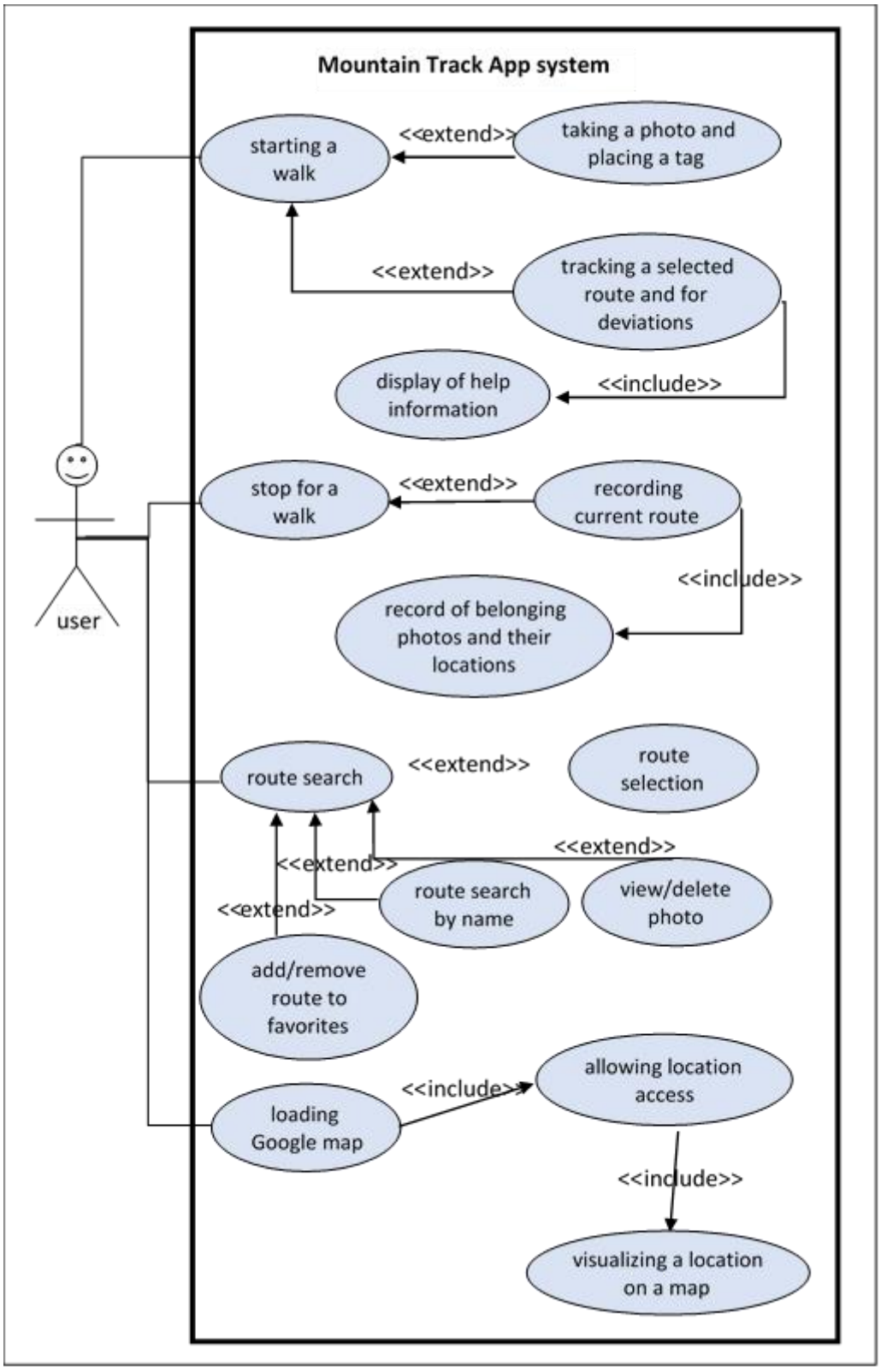

Fig. 1. – Use Case Diagram

2. Routes:
   - loading a ready-made route (track) – prepared in advance;
   - creating a track along the route traveled;
   - creating a track on a map;
3. Analysis and tracking:
   - auxiliary information (calculation) for route length;
   - time at set average speed and;
   - route deviation monitoring;

4. Multimedia and location:
   - taking photos along the route, creating a marker on the map for each point where there is a photo.

The relevance of the topic is determined by the increased interest in independent tourism, ecotourism and mountain trekking. More and more tourists are looking not only for natural experiences, but also for opportunities to document, share and analyze their routes. Providing a digital tool to help them in this endeavor is a logical and timely step in the development of tourism services.

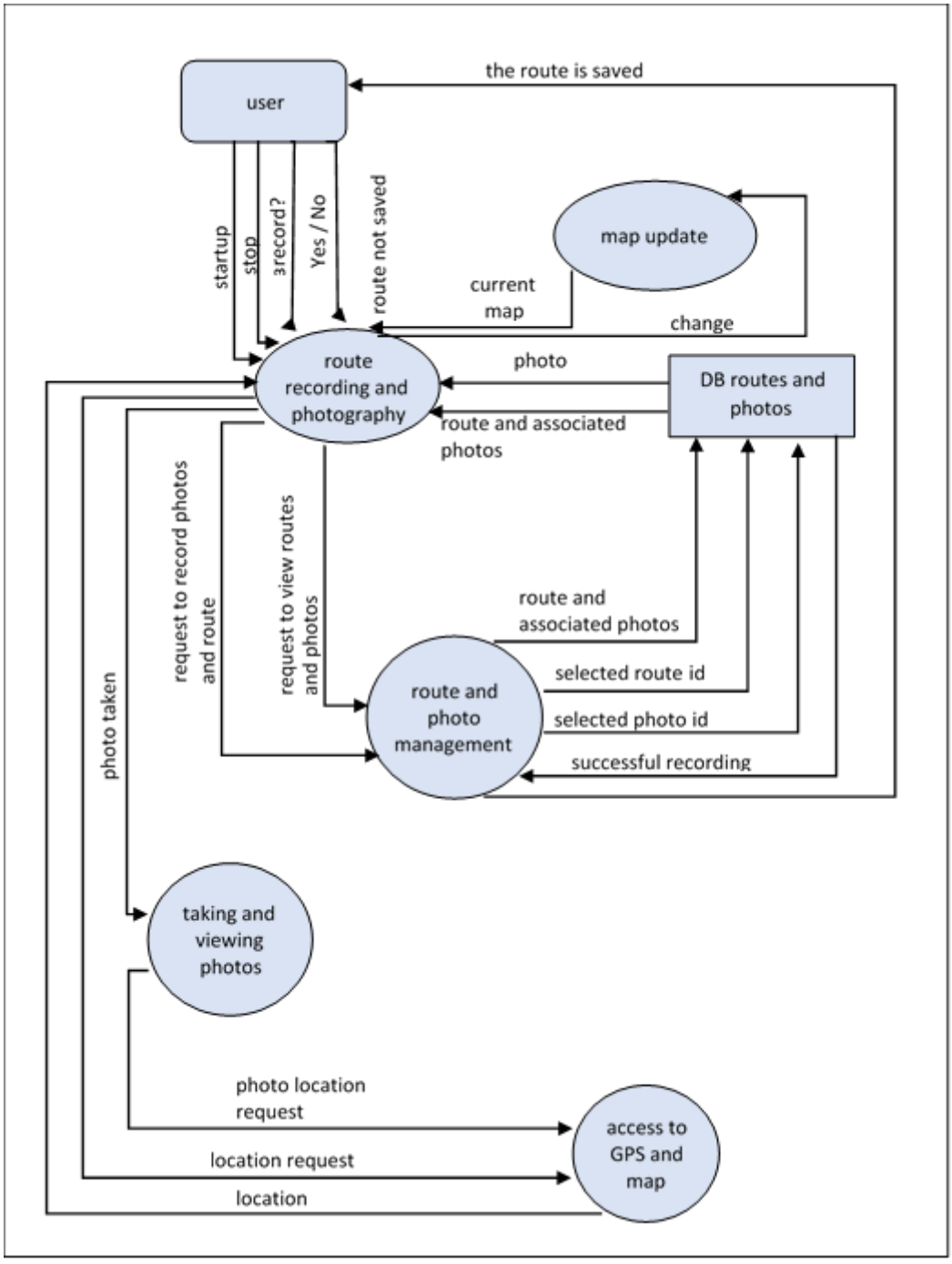

Fig. 2. – Data Flow Diagram

The development of such an application not only increases the quality of service in the tourism sector, but also provides added value for customers - through convenience, security and a personalized experience. It also opens up new opportunities for marketing, analysis of customer preferences and optimization of the routes and services offered.

In the study of existing solutions, four of the more popular navigation and hiking applications were examined: AllTrails, Mountain Maps, Gaia GPS, PeakVisor. After analysis, the following generalizations can be made:

General advantages:

- **Rich route database:** All apps offer access to thousands of predefined trails with information on length, difficulty and elevation gain.
- **GPS navigation and recording:** Allows real-time movement tracking and route recording.
- **Offline support:** Ability to download maps and use without the internet (mostly in paid versions).
- **Community and ratings**: Includes user reviews, photos, and sharing experiences.

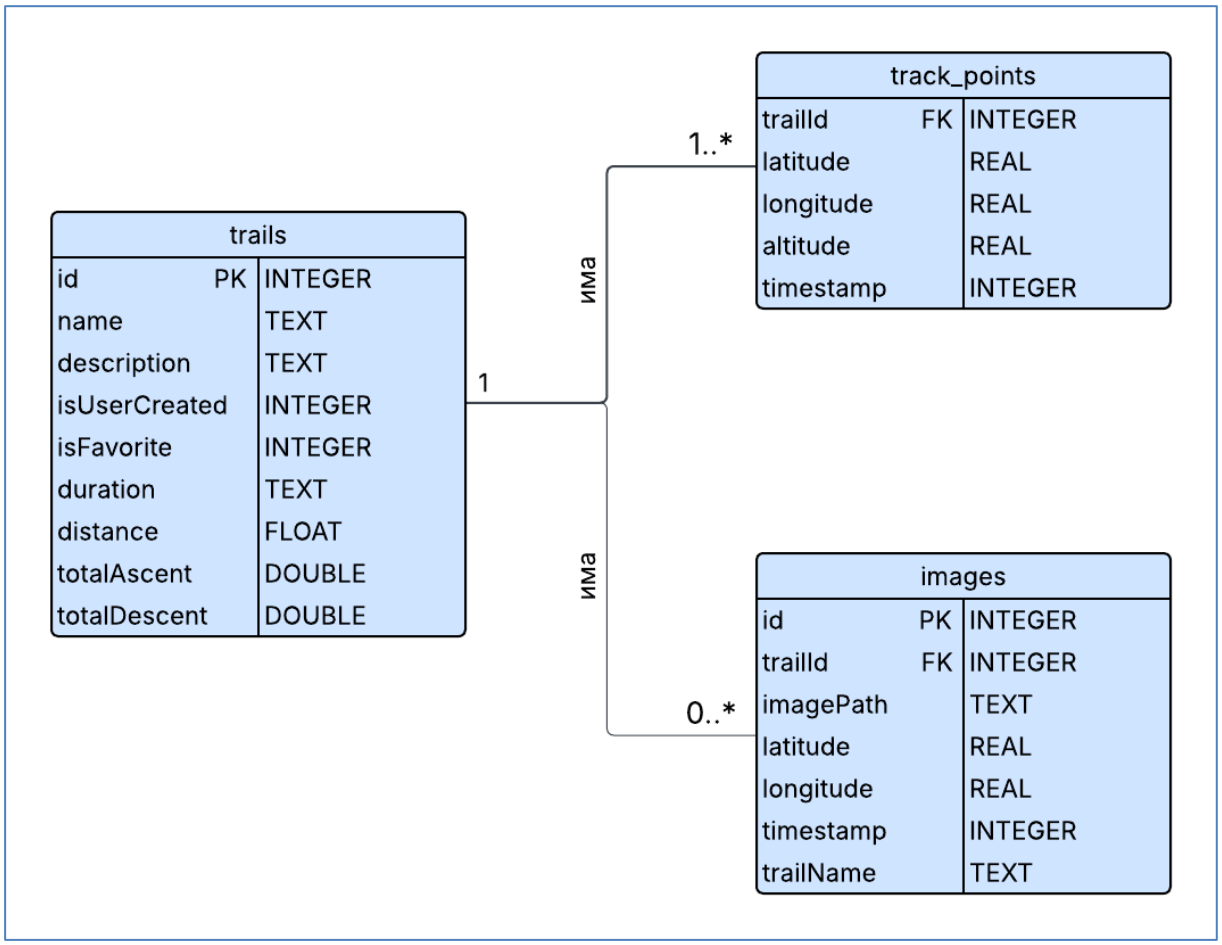

Fig. 3. – Conceptual model of a database

Disadvantages:

- **Limitations in the free version:** Important features such as offline maps, deviation warning, 3D views or specialized layers are paid.
- **Complexity or overloaded interface:** Some apps (e.g. Gaia GPS) require time to learn and set up.
- **Internet or subscription dependency:** In the absence of offline access or an active subscription, functionality may be limited.

This provides a basis for formulating a goal and objectives for the implementation of the project:

Goal: Development of a lightweight, offline-working Android application for tourist routes, using the Google Maps API for visualization, providing the ability to:

- creating a new route/track in real time via GPS tracking, saving (by assigning a name, description and image) and reusing routes (tracks). Dividing routes into own, favourite and predefined.
- visualization and navigation with real-time GPS tracking
- monitoring for deviation from the loaded route, detection and signaling of deviation

- add and view photos at waypoints. Automatically link the image to the current GPS coordinates and time of creation
- offline access to selected routes and images (pre-stored on the device).

Tasks to achieve the goal:

- Building a local database (SQLite) to store routes, images, and user activity information.
- Integrate Google Maps SDK for visualization of routes and user position.
- Real-time route recording and playback, including notifications when you deviate from the path.
- Adding the ability to attach photos to waypoints with coordinate and time information.
- Creating a route selection interface (second Activity) with images and descriptions.
- Providing basic offline functionality, including visualization of previously downloaded routes and images.
- Multilingual support for wider accessibility.

This approach allows for the creation of a customized offline application that fills the gaps of existing platforms and meets the needs of a specific user group.

System design, description of the used technologies

1. **Use-Case Diagram – fig. 1:** presents a logical model of the interaction between the user and the main functionalities implemented in the system. The main actor is the User, who interacts with the mobile application through a graphical user interface.

The diagram includes **basic**, **extended** (<<extend>>) and **included** (<<include>>) functionalities that reflect the logic and conditions for activating various actions.:

- *Loading Google Map*. Requires <<include>> connection to the Allow Location Access feature.
- *Visualize location on a map.* Includes <<include>> connection with GPS access permission.
- *Start Hike*
- *Stop Hike*
- *Route/track selection*
- *Route/track search.* Extended via <<extend>> links to*:*
  - Search(looking) for a route by name
  - Add/remove from favorites
  - View/delete photo
- *Recording a current route.* The feature includes Recording images and their locations when photos are taken
- *Take a photo and add a tag*
- *Tracking a selected route and for deviations*

Some non-functional requirements must also be reflected in the system design:

- **Platform:** The application must run on Android devices (version 11 (API 30) and above).
- **productivity:** The application must use the device's resources economically and not unnecessarily strain the battery during GPS tracking; Working with the map and images must be smooth, without delays or crashes.
- **Security and access rights:** Request permission to access location, camera, and storage as required by Android; The app should not send personal data outside the device.
- **Sustainability and extensibility:** The code should be structured modularly so that new features can be added in the future (e.g.: sharing routes online, cloud synchronization, etc.).

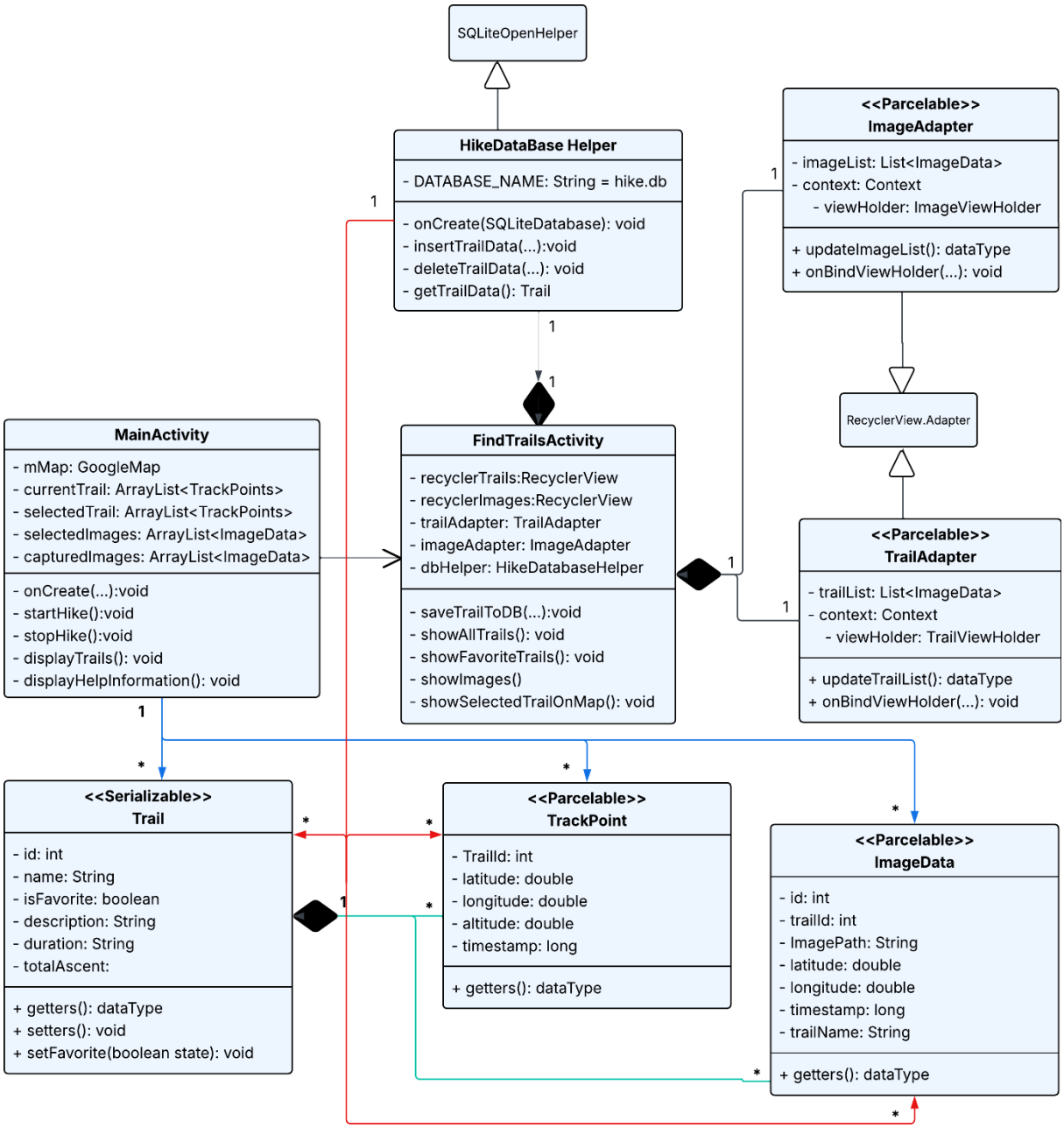

Fig. 4. – Class diagrams

2. DFD/Data Flow Diagram - fig. 2 - describes the main processes, the flow of information and the connection to the relevant repositories**.**

Basic processes:

- *Capture and view photos and tags*
  - Allows taking photos, placing tags, and choosing to view or delete.
  - GPS connection for location request when taking a photo in the process of recording a route and taking a photo
- *Route and photo management*
  - Responsible for creating, storing, and retrieving routes and associated images.
  - Receives requests for recording or viewing from the Route Recording and Shooting process
  - Saves data in the Routes and Photos database
  - Provides data to the user when requesting to view a route or photo

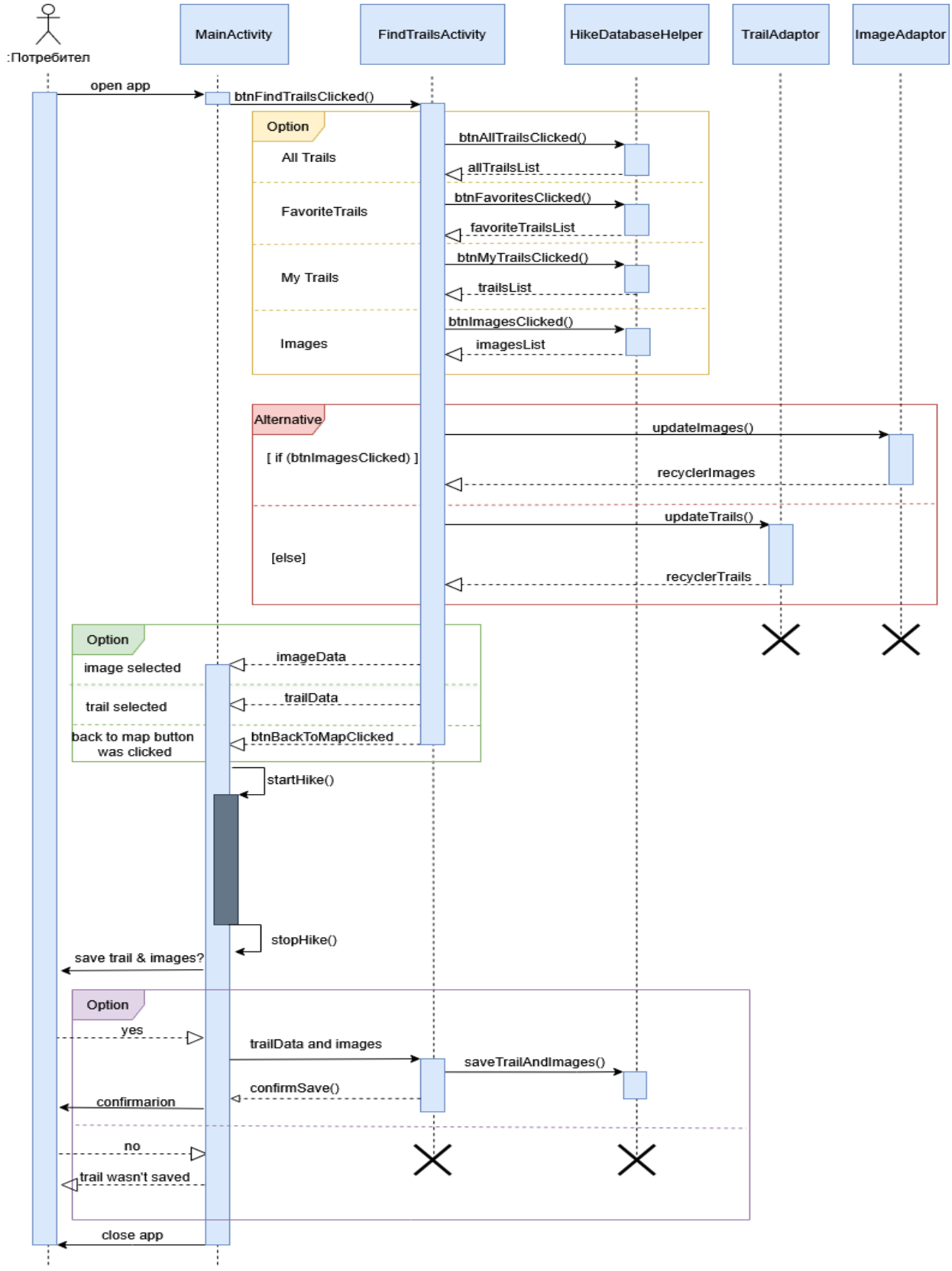

Fig. 5. – Simulation of user behavior through interaction diagram (sequence diagram)

- *GPS and map access*
  - Responsible for updating the user's current location or returning the current location when a photo is taken.
  - o Receives requests from the "Capture and review photos" and "Record route and capture" processes
  - Sends data back to update the map view

- *Route recording and photography*
  - Starts and stops a route
  - Captures images
  - Transmits data to "Routes and Photo Management" and to "Map Update"
- *Map update* - Updates the map visualization with new information – route, photos, markers
- *Data warehouse: DB Routes and Photos* - Stores all saved routes and associated images with their geolocation.

The application architecture divides the main responsibilities between several key components (Activities, helper classes and databases), with clear communication between them.
The main components are:

MainActivity - Responsible for starting/stopping a hike, tracking the route in real time via Google Maps API, taking photos and placing markers, using GPS for location and altitude, zooming and maintaining the map (including offline), passing the route to FindTrailsActivity when stopping the hike

FindTrailsActivity - Acts as a central point for receiving data from MainActivity; creating and saving a route, waypoints and images in the local database via **HikeDatabaseHelper**; extracting saved data (routes and images) from the database; displaying the retrieved route information and images in **RecyclerView** (uses **TrailAdapter** – to visualize a list of routes, through a layout component for them and **ImageAdapter** – to view and manage photos, through a layout component for photos); provides options for selecting and loading a route with associated photos back to MainActivity, for viewing a photo and loading it on the map in MainActivity, for searching for routes by name.

HikeDatabaseHelper - Data Access Layer implemented using SQLite. *Manages three main tables:* trails – to store information about routes (name, description, type (favorite, user-created, or default) and ID); track_points – to store GPS coordinates, altitude, and timestamps associated with a given track ID; images – to store a path to an image, geolocation, and associated route ID.

3. **Data organization - conceptual database model (fig. 3). -** the software system uses a local relational database (SQLite) that stores route data, GPS points, and images taken during hikes.

4. **Class diagrams of individual classes (fig. 4) –** presents a generalized model of the individual classes that participate in the application, the architectural dependencies of the individual components and the basic functionality of each of them.

5. Simulation of user behavior through MountainTrack's Sequence diagram (fig. 5) - To illustrate the interaction between the main modules and the user, a sequence diagram is provided, which simulates a real-world scenario of using the application.

CONCLUSION

A stable mobile system for recording and viewing tourist routes has been implemented. A mechanism for adding images and their visualization on the map has been successfully implemented. The application supports both custom and predefined routes, filtering and searching. A local database has been created with full support for routes, coordinates, photos and descriptions. The interface supports Bulgarian and English.

The application meets the basic functional and non-functional requirements set at the beginning of the project. The Google Maps API, together with SQLite and Android Location Services, provide a good foundation for creating a travel application. Support for images, favorite routes and offline modes contributes significantly to the usability.

REFERENCES

1. **AllTrails**. https://www.alltrails.com/ – Platform for sharing and navigating tourist routes. Used for user interface and feature analysis, 15.05.2025.

2. **Mountain Maps.** https://www.mountainmaps.it/en/ – Example of an application with offline functionalities and maps, 15.05.2025.

3. **Gaia GPS Community.** https://help.gaiagps.com/ – A source for analysing user opinions and problems with GPS tracking, 15.05.2025.

4. **PeakVisor.** https://peakvisor.com/ – Reference for integration of elevation information and terrain visualization, 15.05.2025.

5. **SQLite Database for Android – Full Course**, YouTube. https://www.youtube.com/watch?v=312RhjfetP8 – Training material for creating and managing a local database, 15.05.2025

6. **Google Maps SDK & Location Services – Playlist**, YouTube. https://www.youtube.com/playlist?list=PLdHg5T0SNpN3GBUmpGqjiKGMcBaRT2A-m – Used to implement maps and GPS functionality, 15.05.2025.

7. **Supporting Different Languages and Cultures. Android Developers**. https://developer.android.com/training/basics/supporting-devices/languages – Guide to adding multilingual support, 15.05.2025.

8. Lucidchart YouTube Channel. https://www.youtube.com/@lucid_software/playlists – Example visualizations of UML and DFD diagrams, 15.05.2025.

9. Kolisnitchenko D.N. Practical programming for Android, Asenevci treid, Sofia, 2014 (Оригинално заглавие: Колисниченко, Д. Н. *Практическо програмиране за Андроид*. Асеневци трейд ЕООД, София, 2014).