LAM Project 2025

Federico Montori, Lorenzo Gigli

April 2025

Rules

In this document we describe two possible projects for the exam of "Laboratorio di applicazioni mobili" course. The project can be implemented by **groups of maximum 2 people** (individual projects are obviously allowed as well), provided they implement the project integration. Each student/group can choose to develop one of the projects proposed here (valid until February 2026) or suggest something else based on his/her personal interests. In the latter case, project proposals **must be individual only** and should be submitted via e-mail to Dr. Lorenzo Gigli `lorenzo.gigli@unibo.it`, with a brief description of the application goals, contents, and a list of **all** the features that you propose to implement. In extremely exceptional cases we can allow group projects outside the one proposed here, but they will need to be discussed with us in advance. The use of Git (or any other versioning system) is strongly encouraged. The following project description contains a minimum set of requirements for the application. Students are strongly encouraged to expand the track, by adding new features to the application, and/or further customizing the contents. Any questions regarding the projects are to be asked to Dr. Lorenzo Gigli via e-mail (`lorenzo.gigli@unibo.it`). Regardless of the choice, every student/group is required to produce:

1. *One or more mobile applications*, there is no constraint on the language and the framework used: it can be native or hybrid, but it cannot be a Web Application (*i.e.* it must run natively on the device, and there must be a reason to do so).

2. *A project report*, which is a document that describes the application produced, focusing on the workflow and the design choices. In particular, the Report should be named `SURNAME1_SURNAME2.pdf` and contain:

    - The name, surname, email, and matriculation number of each component of the group.
    - Overview of the application with screenshots.
    - Implementation details on how you chose to implement the functionalities.

A good report is probably between 10 and 15 pages. Less than 5 pages are probably too few, and more than 15 are probably too many. The quality of the report WILL be part of the evaluation. **A good report also contains about 70% of implementation details and choices and only the remaining 30% of screenshots, overview...**

3. *A presentation*, which consists of a set of slides that will help you in the project discussion. They should contain a brief recap of the report alongside screenshots of the application. We suggest producing around 10 - 15 slides since the presentation time is approximately 10 minutes (a group discussion may last longer). Furthermore, **each** component of the group must know the details of the entire implementation and will be possibly asked about it during the project discussion.

The **CODE** of the mobile application and the **REPORT** have to be uploaded exclusively on the Virtuale platform in the dedicated section[1] (there are 6 deadlines throughout the year). They must be enclosed in a single .zip file named `SURNAME1_SURNAME2.zip`. In the case of a group, a single component is in charge of completing the upload. If the archive is too big for Virtuale, you can remove the directory "build" from your code (Android only) or any auto-generated set of files. If the archive is still too big, then the student(s) must share the zipped code via Drive/OneDrive, etc. In this case, the students must still upload on Virtuale a zip file containing the Report, the link to the shared code, and its **hash**, so that we can prove that it was delivered on time. Projects delivered via e-mail will **NOT** be taken into account. The SLIDES, instead, must be brought along on the day of the oral examination.

Do not forget that the oral examination consists also in a theoretical part, which is **individual**. Therefore, knowledge of the topics discussed in class (mostly Android, but also iOS) is required. The exam consists of the project discussion (which is per group) and the oral examination (which is per student) and **must** be booked via Almaesami.

Alternatively, the student may choose another option: The Seminar! Details about it are at the bottom of this file. If the student chooses this path, which must be first acknowledged by us, then it is not necessary to deliver a project at all.

The Project: "Travel Companion"

Overview

In this project, the student is required to develop an interactive mobile application that assists users in planning, tracking, and documenting their travel experiences. The app must allow users to create trip plans, log their journeys using GPS, capture moments with photos and notes, and visualize their travel history

[1] `https://virtuale.unibo.it/course/view.php?id=28374`

through maps and statistics. Trip-related data must be stored in a local database, and the app must provide an interface for users to start and stop journey logging. Additionally, the application must support background functionalities, such as sending notifications about nearby points of interest.

Record the Activities

The application must provide an interface for users to create trip plans and manually start/stop journey logging during travel. A trip plan includes essential details such as the destination and travel dates, while journey logging records the user's route using GPS when enabled. For each journey, the app must:

- Record the time spent traveling and GPS coordinates of the route.
- Allow users to attach photos (via the camera) and notes to specific moments or locations during the journey.
- Store all trip-related data—trip plans, routes, photos, and notes—in a local database once logging is stopped.

The application must support at least three mandatory trip types:

- **Local trip**: e.g., within the city.
- **Day trip**: e.g., a short out-of-town excursion.
- **Multi-day trip**: e.g., a vacation.

The student may introduce additional trip types. For **multi-day trips**, the application must include a feature to calculate and display the total distance traveled based on GPS data.

Users must be able to view their past trips and logged journeys in a list or on a map, with at least one filtering option (e.g., by date, destination, or trip type). Since users may not log every moment of their day, the app must handle periods without active trips appropriately — for example by marking them as "no travel" or through another approach defined by the student.

Display Charts

The application must include a section displaying at least two different visualizations of the user's travel data. Examples include:

- **Map View**: Displays recorded travel routes or generates a heat map of visited locations over a selected period (e.g., the past month).
- **Bar Chart or Timeline**: Shows the number of trips taken or the total distance traveled per month.

These visualizations must be interactive, allowing users to explore their travel history. They must be based on data stored in the local database. Students are encouraged to implement additional visualization types, such as a photo gallery organized by trip.

Perform Background Jobs

The application must support the following background operations:

1. **Periodic Notifications**: The app must send at least one type of recurring notification. Examples include:
   - Alerts about nearby points of interest (e.g., landmarks or attractions) based on the user's current GPS location.
   - Reminders to log a journey if the user has not recorded a trip recently.
2. **One Additional Background Operation**: The student must implement **one** of the following:
   - **Automatic Journey Tracking**: Using the Activity Recognition API, the app detects when the user is moving (e.g., walking, driving) and either prompts them to start logging a journey or begins recording automatically. The app may occasionally ask the user to confirm the trip type for accuracy.
   - **Geofencing**: The application allows users to define areas of interest (e.g., home, a frequent destination) and records when they enter or exit these zones using the Geofencing API. These events must be stored separately from journey logs (e.g., as timestamps or notifications), and the student must determine how they integrate into the overall functionality.

Project Integration for Two Students

For projects developed by two students, an additional feature must be implemented: the ability to track and visualize personal travel progress using a **Personalized Progress Prediction Module**. This module operates entirely on the user's device, analyzing past travel data to estimate future trends and provide insights.

Personalized Progress Prediction Module

This module enhances the application by providing:

- **Data Analysis**: The system identifies patterns in the user's past trips, such as travel frequency, distances covered, and visited locations.
- **Forecast Generation**: The app predicts key metrics (e.g., projected number of trips or estimated travel distances for the upcoming month) and displays them in an interactive format, such as charts or progress bars.
- **Personalized Goal Recommendations**: If the forecast indicates a decline in travel activity, the app may suggest new trip ideas or recommend increasing travel frequency.

- **Local Execution**: All processing occurs on the device, ensuring privacy and independence from external services.

The Project: "Trust App" (special track)

Overview

In this project, the student is required to develop a mobile application that interacts with the smart contracts of the ZONIA oracle system. The application shall enable the user to request trusted IoT data via blockchain transactions, store the retrieved data in the app's local storage, and visualize it through an intuitive user interface. In addition, the application must integrate MetaMask for wallet management and conduct all blockchain transactions on the Ethereum Sepolia network. Due to the complexity of the project, detailed information—including smart contract specifications, the relevant ABI for functions, and additional implementation details—will be provided upon request.

The Seminar

An an alternative to the project, a student may choose to propose a seminar. In this case, the student must propose a topic, obviously coherent with the course, about a technology that however is not in the contents of the course. Examples are:

- React Native
- Ionic - React
- Xamarin
- Kotlin Multiplatform

These are only examples and we welcome proposals. The seminar must be in the form of a small lecture (30-40 minutes) held by the student in front of the class, possibly during the last lectures in May. The Lecture must include slides and a little bit of live coding. If the student chooses this path, then only the oral examination will be required. Please send all your seminar proposals to me at `federico.montori2@unibo.it`.