INSTITUTO SUPERIOR TÉCNICO

Departamento de Engenharia Informática

Project Assignment for Ubiquitous and Mobile Computing

MEIC/METI 2017-2018 – 2nd Semester

HopOnCMU

1. Introduction

The goal of this project is to develop a distributed mobile application named HopOnCMU, which aims to provide tourism quizzes on hop-on-hop-off buses. HopOnCMU will allow users to answer quizzes about monuments that they visit on hop-on-hop-off tours and to compete with other users for prizes given out at the end of the touring day to those with the most correct answers.

Consider the following example of a HopOnCMU usage scenario. A group of users takes a hop-on-hop-off bus in Lisbon and drives to a monument (say the Belem tower). There, users download from the monument's WiFi access point the questions for HopOnCMU related to the monument. As the tourists answer the questions, they can share partial results (namely how far all of them have come along and the ranking of the passengers with more correct answers). They continue to travel by bus to several more monuments where the process is repeated at each one. At the end of the touring day, the tourist with the most correct answers gets a prize from the tour operator. The following section explains the system functionality in more detail.

2. Specification

2.1 Baseline Functionality

HopOnCMU is a mobile application that allows users to obtain and answer quizzes. Users can: i) receive quizzes from a centralized server (see also project advanced features); ii) share results with fellow users (see also project advanced features); iii) periodically submit their answers to the centralized server, and iv) receive prizes at the end of the day.

2.1.1 Architecture and Functions

The basic architecture of HopOnCMU relies on a central server and a client mobile application. The central server is accessible through the Internet and is responsible for managing the state of the system. The client is a mobile Android application that users install and run on their devices and allow users to perform the following functions:

F1. Sign up
F2. Log in / out
F3. List tour locations
F4. Download quiz questions (at the corresponding monument)
F5. Post quiz answers for one monument
F6. Read quiz results (number of correct answers and ranking)

The sign up operation allows users to create a new user account in the system; the user must enter a username and a code obtained from her bus ticket. The client then contacts the HopOnCMU server, which must ensure that the new username is unique and that the code has not been used to sign up before (the code should be used as the password every time the user logs in). If the operation is successful, the user can then log in and start a new session on the client.

To perform useful functions on HopOnCMU, the user must log into the system with his account credentials (username and ticket code), which must be validated by the server. If they are valid, the server must generate a new session id, keep an internal record associating that session id to the username and return the new session id to the client. However, if the credentials are invalid, the server must return an error. Only users with valid session ids will be allowed to perform additional functions (F2-F6). The log out function ends the current session.

Once a session is active, users are able to manage quizzes. A quiz is a set of multiple choice questions related to a monument. Quizzes can only be obtained while connected to the WiFi network at the monument the quiz relates to.

2.1.3 Quizzes

Quizzes are sequences of multiple choice questions. They must be answered between two stops on the tour: the stop where the quiz was obtained and to which the questions are related, and the following stop. At all tour stops users may submit the results of the last quizzes they answered and may also indicate that they are finishing the day's tour at that stop. Prizes are assigned for the users who answer more correct questions in a particular day (which may for some users involve multiple trips on the buses). The questions/answers are received/sent from/to the server at the bus stops using the monuments' WiFi networks.

2.1.4 Quizz Progress Sharing

Between bus tour stops, users can share the progress of their quiz answers with other users, so that everyone participating in the quiz is as much as possible aware of how many answers each user has answered (and gotten right). Students should design a protocol resorting to WiFi Direct to share the users' progress among client devices.

2.2 Advanced Features

In addition to the baseline features, students must also implement two advanced features (each worth circa 3 points) in order to get the full 20 points as described below:

Timed quizzes. In this more advanced feature, quiz results are going to be compared not only by the number of correct answers but also by how fast the quiz was finished. This feature aims to speed up the delivery of quizzes and quiz results between bus stops for users who do not want to pay for using the mobile network by relying on other tourist on the bus who have access to the mobile network (think, for example, of a bus with tourists from the EU with no roaming charges helping tourists from other continents). Students should design solutions that ensure the reliability of the timestamp that marks the moment when the user finished the quiz. Solutions that reduce the number of messages exchanged and/or the power consumption will be favored.

Security. In the baseline setup, security is absent. Students are invited to enhance the security of the system in terms of attaining the following security goals: (1) ensure that the communication between clients and server is secure against an adversary with the ability to eavesdrop the network, modify the packets exchanged or introduce new packets, and (2) ensure that a message receiver can authenticate the message as published by a specific user and that it has not been tampered with during the transmission.

3. Implementation

The target platform for HopOnCMU is Android version >= 4.0. Communication between tour passengers should be based on WiFi Direct. The official course language for the project is Java but other language may be allowed (check with the course staff). Students may use Android APIs freely. However, third-party libraries are not allowed unless explicitly approved by the course staff.

Geographical coordinates can be expressed as a list of WiFi IDs, meaning that if a device is able to sense a WiFi signal from any of the WiFi IDs in the list, HopOnCMU will acknowledge that device as being positioned on that location. This ID can be the SSID of a real WiFi router or (set of WiFi routers), for example "museu-oriente".

Unfortunately, it is not possible to provide real Android devices and BLE beacon stickers to test the project. Therefore, the project must be tested on a software-based emulation testbed, which comprises the native Android emulator and the Termite WiFi Direct emulator. This testbed will allow you to emulate: 1) users devices executing the mobile application, 2) communication between devices, 3) beacon signaling, and 5) communication with HopOnCMU server. The testbed can be set up on a single computer. This computer can also be used to host the HopOnCMU server.

To implement the HopOnCMU scenario, a single WiFi network should be used for all communication. The fact that a client is near a monument, is determined by the fact that the client device detects a WiFi beacon with a SSID named "M**n**" where **n** ist the number of the monument.

Students should divide the emulator running the HopOnCMU app in two groups. Let's call them native tourists and foreign tourists. These should correspond to odd numbered emulators and even numbered emulators respectively. All tourists may download quizzes and upload results when at a monument (near a "Mn" WiFi beacon). Foreign tourists cannot communicate with the WiFi network when they are not at monument and must therefore ask native tourist for help via WiFi Direct if they want a native tourist to upload a quiz result to the server (via WiFi). Native tourist always have access to the WiFi network but cannot download quizzes when they are not near a monument.

The Android emulator comes natively with Android SDK and provides support for GPS and Internet connectivity. However, it does not emulate WiFi Direct and Bluetooth APIs.

Termite is a WiFi Direct network emulator. Termite can emulate a wireless network of virtual nodes that move over time and are able to interact opportunistically whenever they are located within close range. Virtual nodes consist of Android emulator instances running the test application. The developer is responsible for specifying the topology and dynamics of the virtual network.

In the context of HopOnCMU, Termite must be used to emulate WiFi Direct communication between user devices. Termite will be made available on the course website.
The interface to be provided by the HopOnCMU app should be simple and, at the same time, complete, i.e. show in a clear way all the supported features.

4. Development Stages

We recommend that you develop the project in five stages:
1. **GUI design:** study the requirements of the project and design the graphical user interface of your application. Create an activity wireframe of the application.
2. **GUI implementation:** implement all graphical components of your application including the navigation between screens. At this point, do not worry about networking. Use hard-coded data to simulate interaction with the server or beacons. Make sure to design your application in a modular fashion.
3. **Communication with server:** implement the central server and extend the mobile application in order to communicate with the server.
4. **WiFi Direct communication:** complete the baseline functionality of the project by implementing WiFi Direct communication. You only need to use Termite at this point.
5. **Advanced features:** implement advanced features regarding security and robustness.

5. Grading Process and Milestones

The projects will be evaluated based on several dimensions. The most important points are: implemented baseline and advanced features, modularity of the implementation, technical quality of algorithms and protocols, and resource efficiency decisions. We will also assess the responsiveness and intuitiveness of the application interface. However, the aesthetics of the GUI will **not** be graded. Therefore, as said above, students should keep the GUI as simple as possible while providing the required information.
There are two important project milestones:
1. April 13: Project Checkpoint – Students should submit their current prototype of HopOnCMU so that they can receive feedback on the architecture and status of the prototype.
2. May 18: Project Submission – a fully functional prototype of HopOnCMU and a final report. The prototype sources and the report must be submitted through the course website. A template of the report will be published on the website. The report must describe what was and was not implemented, and it is limited to 5 pages (excluding cover). The cover must indicate the group number, and name and number of each of the group's elements.

6. Collaboration, Fraud and Group Grading

Student groups are allowed and encouraged to discuss their project's technical solutions without showing, sharing or copying code with other groups or any other person from within or without the course. Fraud detection tools will be used to detect code similarities. All instances of fraud will disqualify all groups involved and will be reported to the M.Sc. coordination and to IST authorities. Furthermore, within each group all students are expected to have a working knowledge of the entire project's code.

GoodLuck!