3

HOMESTAY MANAGER

Contents

Introduction

You are tasked with developing a mobile application in Android intended to match visitors/home-seekers with families/home-havers. (details below). Each team will have (and is encouraged to exercise) a large amount of creative liberty with this project, however, certain requirements are mandatory. We are more than willing to award extra credit for creativity. Grading is based on:

1. Your *Requirements Document,* for which you should write English text and UML diagrams including:
    1. Use Cases
    2. Class
    3. Sequence
    4. Collaboration
    5. State Chart

    The Requirements document is due Friday, April 18; details about submission will be posted.

2. A *specification document* in Z. Details will be posted.
3. *Demonstration* of a working system. The demonstrations will take place the last week of class, with details to be posted. For the demonstration, grading will be based on:
    a. Functionality: (Does it achieve our minimal expectations? Does it do more than what is minimally expected?)
    b. Aesthetics (Is the interface interesting?)
    c. Usability (Are there any significant hurdles to overcome when using it?)
    d. Implementation (Are there any significant bugs or design flaws? When you demonstrate it to us, we will try to break your system.)

After completion, students should expect to walk away with the following:

1. Experience with real-world and practical software design/development
2. Knowledge of Android programming
3. Use of UML and Z in the development process
4. Well-earned and long-lasting sense of achievement hopefully not forgotten the first week in summer.

5. Potential cash-in-hand from local company looking for this kind of software; more about this later.

The HOMESTAY MANAGER

Homestays are a type of rental agreement wherein visitors rent rooms from local families. Unlike traditional renting/subletting, visitors (in our case students) are not only expected to reside with their host families, but also to interact with them on a daily basis, in essence becoming "part of the family." This arrangement is mutually beneficial, awarding host families extra income while providing students with a place to stay where they are immersed in the local culture.

Arranging a homestay can be logistically challenging and as such some are set-up via third parties, such as schools or dedicated companies. Your task for this project is to design a mobile application (Android) that can be used by students, home families and a third party broker to arrange homestays. The most successful project may have the option of being purchased or worked on further by/with a local company.

Expected details and functionality are below.

Creating user accounts

The application should provide minimal functionality for three kinds of users:

- Visitors/students, who can search for available rooms
- Host families, who can advertise their rooms
- Administrators, who can monitor and modify submissions by any user, as well as issue approvals or rejections of any proposed matching

Profile Creation

Details about host families and students are collected and stored in the form of profiles. Both students and host families have the option to designate certain fields as "private," which will impact their ability to matched later on (private information will not be publicly revealed, but will be used internally by the application).

The typical profile for a host family should have the following details:

1. Name
2. Photo
3. Availability to host students; dates for which their home is available.
4. Mailing address

5. Email address
6. Phone number
7. Smoking preference
8. Pets (number of dogs, cats, etc.)
9. Distance from UCDavis (can be calculated from the address to a reference address)
10. Gender preference (some families prefer only one gender)
11. [Optional] Short blurb

Student profiles might contain the following:

1. Name
2. Photo
3. Arrival and departure dates; the period(s) for which they need housing.
4. Mailing address
5. Email address
6. Phone number
7. Allergies (food, environmental, etc.)
8. Smoking preference
9. Pet preference (at least two ewoks, no grues, etc.)
10. Family structure preference [number of children, ages, genders]
11. [Optional] Short blurb.

Operation

When a student wants to find a host family to live with, the student creates an account in the homestay manager and fills in his or her profile details. At this point, two options are available for matching.

1. Manual selection
2. Placement wizard

MANUAL SELECTION

In manual selection, the student gets to pick the host family based on public profile details about the host family (due to privacy issues, details like name, photo, address, etc., need not be exposed to the public). For this feature, the application needs to have a dash board with multiple filters (e.g., no smoking, two children, etc.). A good reference/potential jumping off point would be https://www.airbnb.com/, where filters for search are displayed prominently and a map-like interface shows potential matches. The student has the option to select up to three host families in order of preference from the list which are sent to the administrator for approval.

One note about manual selection is that the filters should only iterate through public information, as using a filter on private information would reveal it.

Keep in mind that a student might have to move among host families during the period of rental. And, of course, the student is allowed to select homes that meet the availability – and other – constraints of the owner.

PLACEMENT WIZARD

The application will also provide a "placement wizard" that ranks host families based on the student's preferences, displaying the public information of the host families along with their rank information. Students then will have the option of selecting up to three host families to be sent to the administrator for approval.

As an example, as soon as the placement wizard option is chosen, a list should be available to prioritize preferences, e.g., some students are allergic to dogs, so in their preferences they have to mention they can not have dogs. They might also prefer to live with a family closer to the school (say within ~1 mile). The priority of the distance preference is clearly less compared to the preference of no dog. A different user may not mind dogs (but prefers no dogs if such an option available), but wants to live very close to campus. This preference ordering may vary for different users and thus the placement process of assigning a host family to a student will be much more effective if the preferences are prioritized. The priority can be 0 (absolutely necessary; if there is no family meeting the preference with priority 0, return a "no family available" response), 1, 2, 3, etc. It should be noted that more than one preference can have the same priority.

The matching algorithm and exact implementation details of the placement wizard is open-ended -- anything from a greedy algorithm to a complex optimized placement algorithm using machine learning can be used. The way preferences are modeled and used is also open-ended: ECS160 students are encouraged to come up with their own models and theories for "best matches" – see below.

Approval

After the administrator receives the potential matches from a student, he or she forwards the majority of that student's information to all hosts that that student selected (sans self-identified private information). If the hosts are interested in that student, the administrator arranges for a group chat between him/herself, the student and the host family. Once the group chat is complete, the student and host family are given the option to accept or reject the matching. If both parties accept, all designated "private" information is shared; if either reject, the process starts over.

Optimizations and challenges

The algorithm of the placement wizard is open ended. A few optimizations are discussed in the following sections. These are just a few suggestions, more creative and efficient optimizations, as well as solutions to as-of-yet unforeseen challenges are encouraged.

HOST FAMILY SCHEDULE FRAGMENTATION

The placement wizard should always try not to create small fragments, i.e., it should avoid minimizing the amount of time the host family is being unused or the student is forced to move – or worse, sleep in the library.

Note: We will show you how Z can easily express these kind of optimizations – much easier than running code.

MULTIPLE HOST FAMILIES

In case there is no single host family schedule fitting the required duration for the student, then an attempt can be made to place the student with multiple host families. Again, please come up with optimizations. Here, the student could have his or her stay split between two families to avoid the lack of a single family for the entire duration.