Sam Borowsky

Blog 4

Since my last blog post, our group has exclusively worked on incorporating internal and external storage into our app. By internal storage, I mean information that is saved within a .plist Xcode file, and by external storage, I mean the use of an SQL database. The internal storage stuff is relatively simple, as the .plist file is essentially a dictionary that contains key:value pairs. For example, the .plist dictionary corresponding to a registered user has the form, firstName: Sam, lastName: Borowsky, email:borowskys16@mail.wlu.edu, cellNumber: (910) 655-6516. A new user is required to supply this information that is written to the .plist dictionary upon registration because it is retrieved and used in various aspects of the app. The external storage side of things, and particularly the PHP development, was initially completely foreign and confusing to me. I have no previous experience with the PHP programming language and only minimal exposure to SQL database creation and interaction. Luckily, Professor Fox is very knowledgeable about these subjects and has helped us greatly by creating the external storage system for us. He also significantly modified our existing app so that it now interacts with (sends information to and draws information from) the external database. The SQL database is basically a massive table of information corresponding to all the users of the app. Each user is assigned a unique phone code during registration that serves to identify his or her device. Upon successful registration, the same information that is stored in the .plist dictionary is sent to the SQL database. Additionally, the SQL database tracks other necessary information, including the user’s driver rating, his or her current location, and his or her “currently\_driving” status (which is indicated by a “0” or “1” value). The app interacts with the database in a couple key manners. For one, when a user navigates to the “Scoop Up” tab and clicks the “Start/Stop Scooping” button, the appropriate information is directed to the database. If the user’s “currently\_driving” status equaled “off” before the button click, it switches to “on” in the database after the click. Similarly, if the user’s “currently\_driving” status equaled “on” prior to the click, it is switched to “off” in the database after the click. This “currently\_driving” value determines which users are displayed in the “Available Drivers” table within the “Scoop Me” tab; more specifically, upon visiting the “Scoop Me” tab, the app populates the “Available Drivers” table only with users whose “currently\_driving” value equals “on.” Furthermore, the app depends on the SQL database to calculate the proximity of each available driver to the user who is seeking a ride, in the sense that it accesses the SQL database for the available drivers’ current latitude and longitude values. Although this external storage implementation is great progress for our group, there remain flaws that must be fixed. Most importantly, it seems the app is unresponsive to live users clicking the “Start Scooping” button to indicate that they are ready to drive. When any live user does this, his information does not appear in the “Available Drivers” table. At this point, the “Available Drivers” table only displays fake users, who I manually input into the database, with “currently\_driving” status equal to “on.” We anticipate that we will be able to release our MVP once we fix this bug so that actual users appear in the “Available Drivers” table. One final update is that Professor Fox helped revamp the sign in/register components of our app. Specifically, Professor Fox recognized that it was clunky and inefficient to force the user to sign in with his or her username and password each time he or she loads the app. Instead, he set the app up such that registration is a one-time phenomenon that happens the first time a new user loads the app. Once a user has registered and confirmed his or her valid W&L email account, each subsequent time he loads the app he is immediately directed to the “Scoop News Feed” tab. This is possible because behind the scenes, when a returning user loads the app, the app checks the .plist dictionary to verify that the device has its own online code (this means it is registered), and then segues to the “Scoop News Feed.” We are very excited with these improvements we have made over these last few weeks and are eager to roll out our initial MVP in a matter of days.