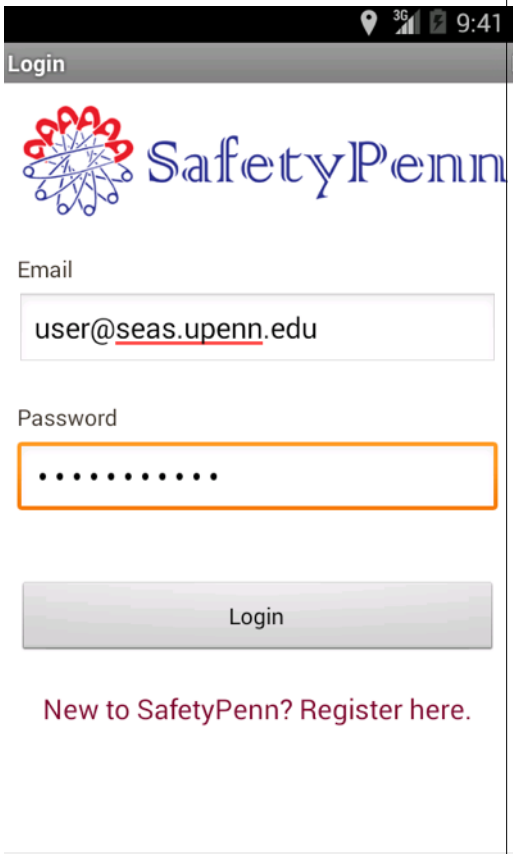
SafetyPenn

User Manual

Ali Altaf
Elliot Boschwitz
Johanna Martens
Brian McNeely

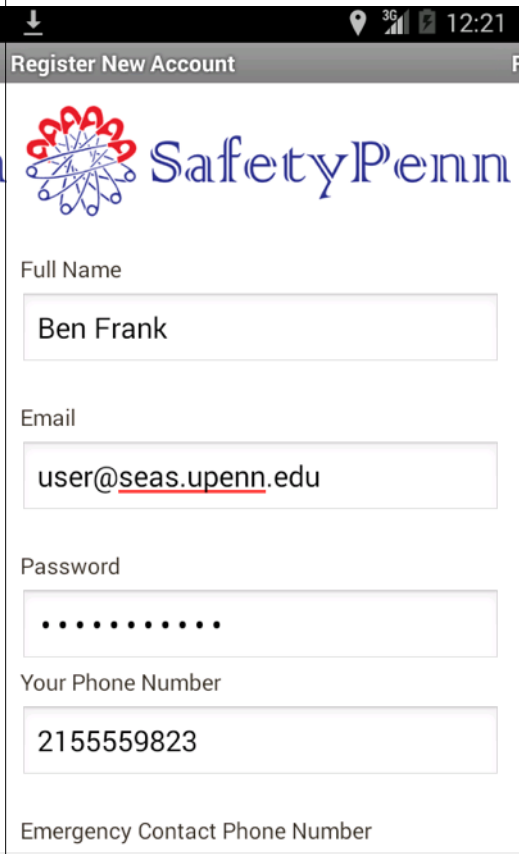
Login



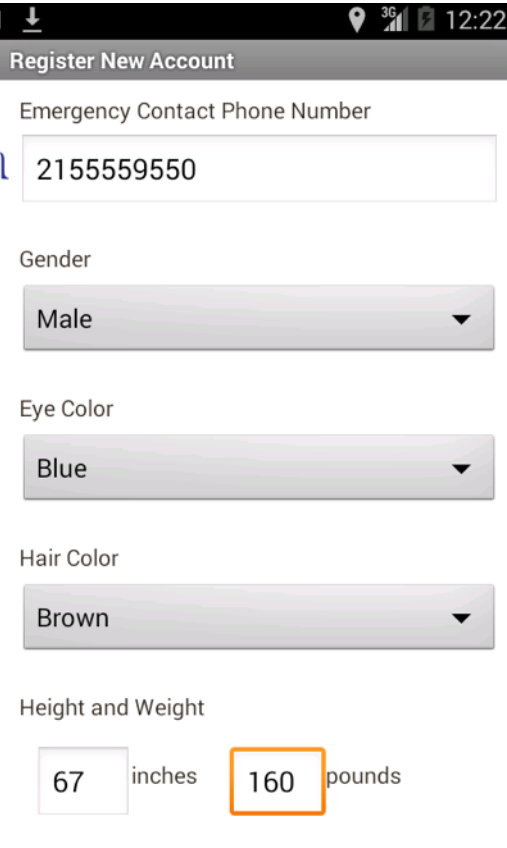
If the user is not logged in or has not registered, the log-in screen is the first to appear.

To register, press the “Register here.” line.

New User Registration

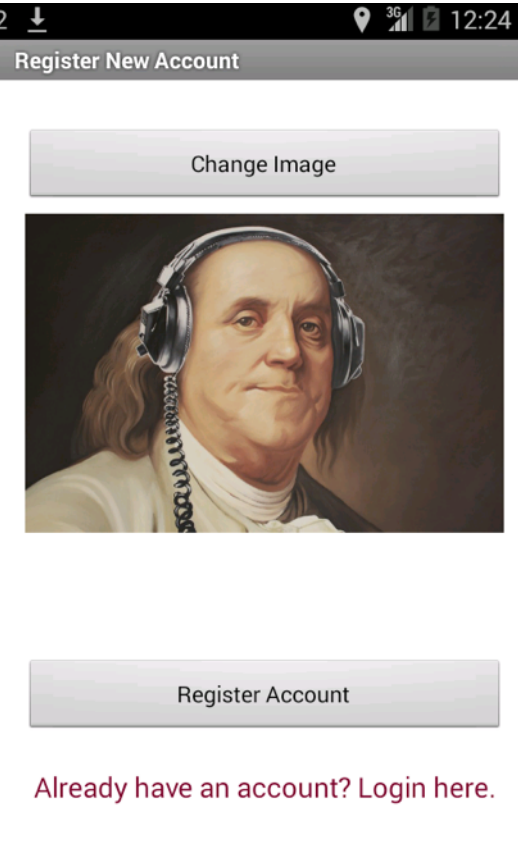


To register a new account, you must first provide your name, email, account password, and cell phone number.



Please enter your emergency contact phone number, which will be contacted in the event of danger.

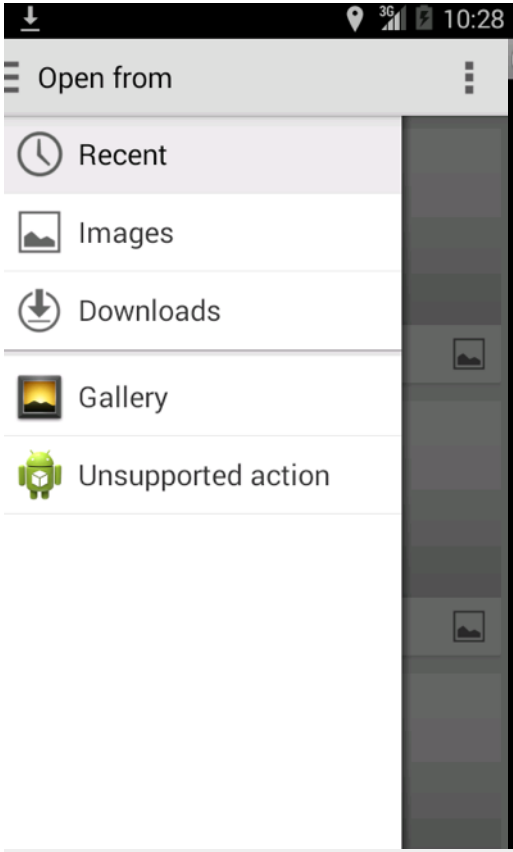
Information about your physical description must also be provided. This information will only be seen by Penn Security.



Please chose an image for your profile. Note that an image thumbnail, as seen above, will only present itself after an image is picked (see the following page for how to chose an image).

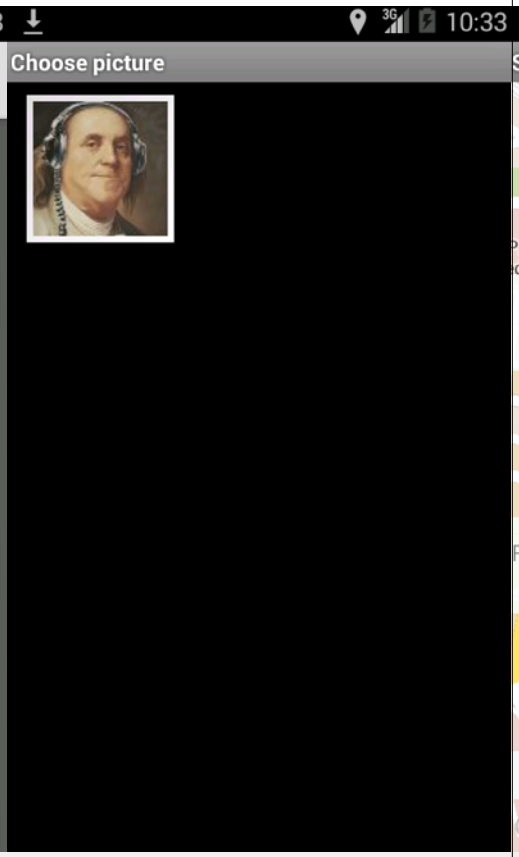
Once the registration form is complete, press “Register Account” to go to the map screen.

Choosing a Profile Picture



In the registration screen, pressing the “Chose Profile Photo” button will bring you to a photo picker screen.

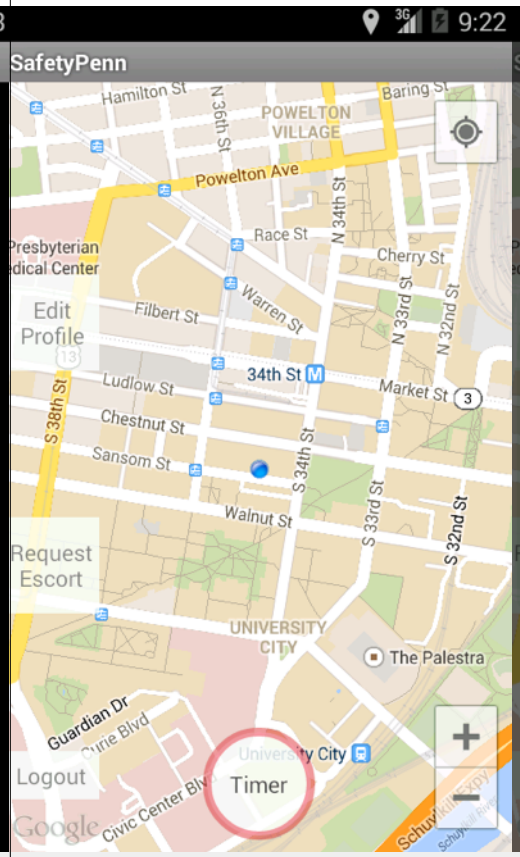
Please make sure to select an image from “Gallery”.



In the Gallery, selecting a photo will return you to the registration page.

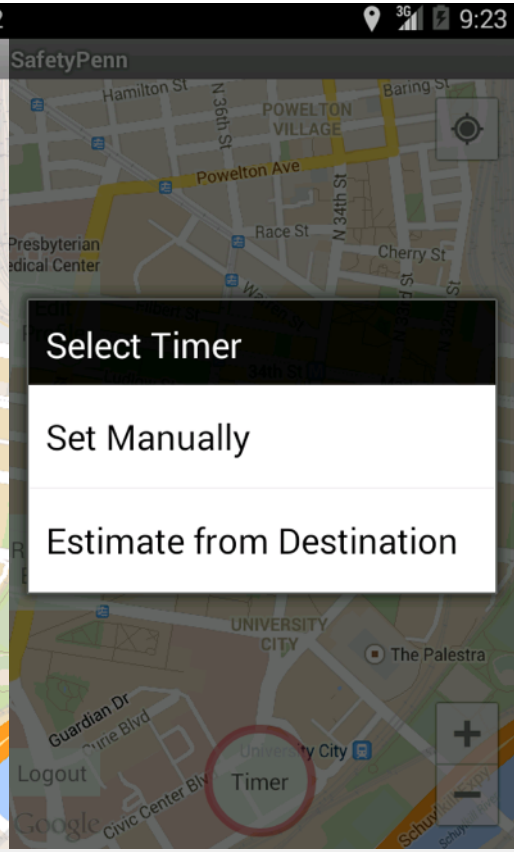
If all other registration fields are completed, you can register your account.

Home Screen



The Home Screen will appear after successful log in or registration.

From here, you may edit your profile, request a Penn Escort, logout from the session, or start a new timer.

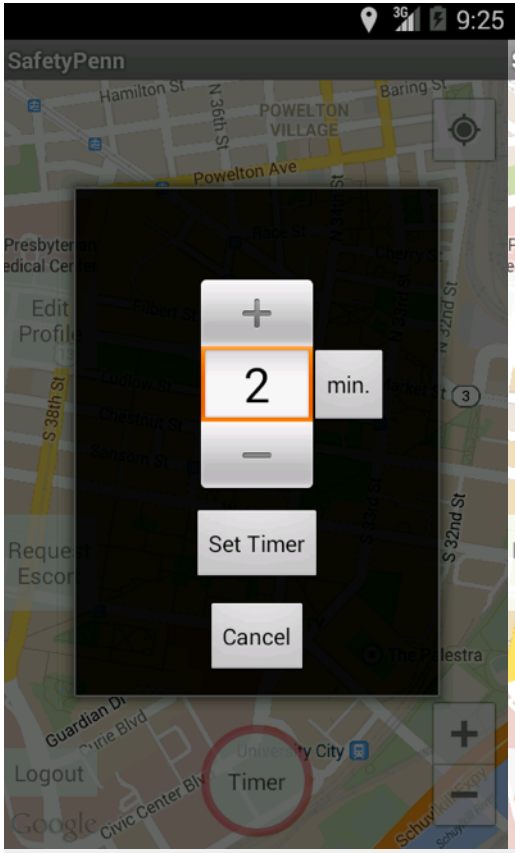


Upon pressing “Timer”, you may select from one of two timers.

“Set Manually” will allow you to set your own custom timer, if you already have a set time in mind.

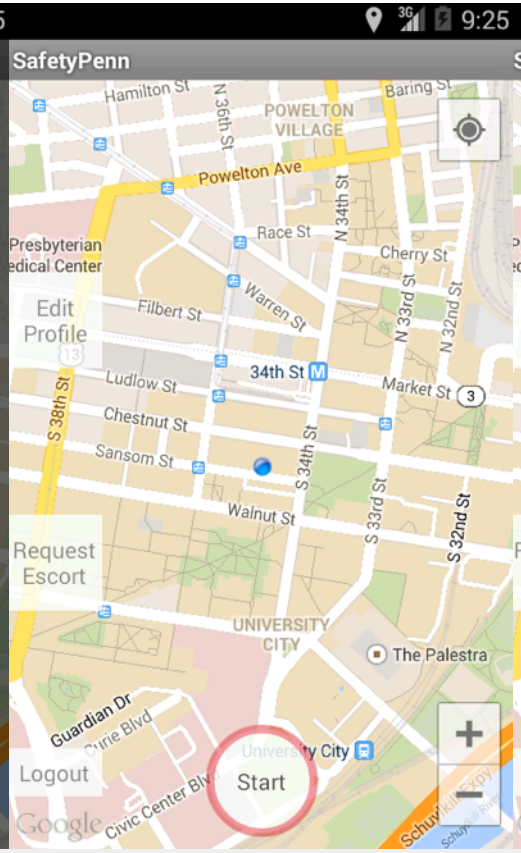
“Estimate from Destination” will calculate the walking distance between two locations, and set the timer accordingly.

Setting Timer Manually

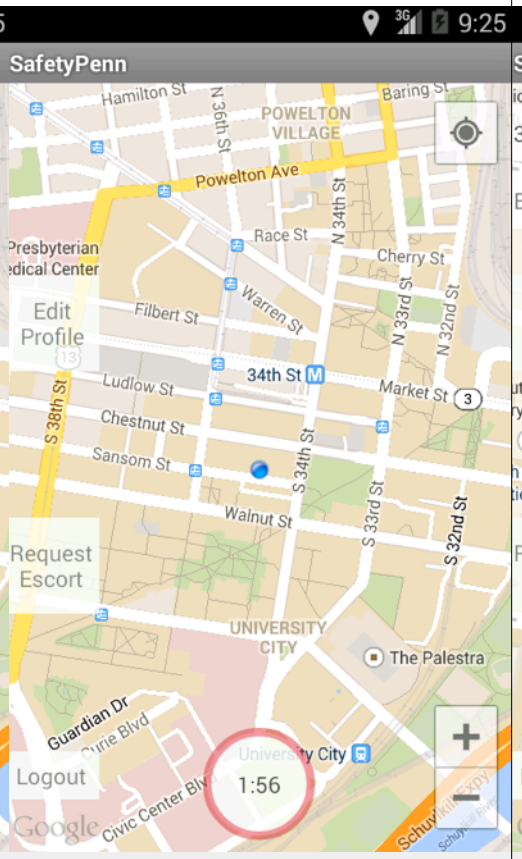


To set a timer manually, select the “+” and “-“ buttons to increment and decrement the timer length.

Pressing “Set Timer” will set up the timer to your set time.



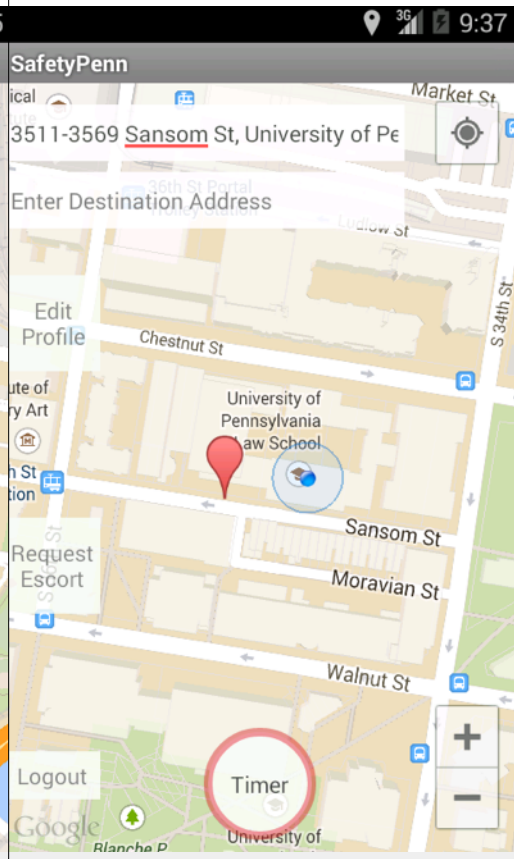
The timer is ready to start when the bottom button labels “Start” instead of “Timer”. Press the button to start your timer.



After pressing “Start”, the timer will display its status on the bottom button. To cancel the timer, simply press on the bottom button to return to the default Home Screen.

If the timer expires before user cancellation, Penn Police will be notified immediately.

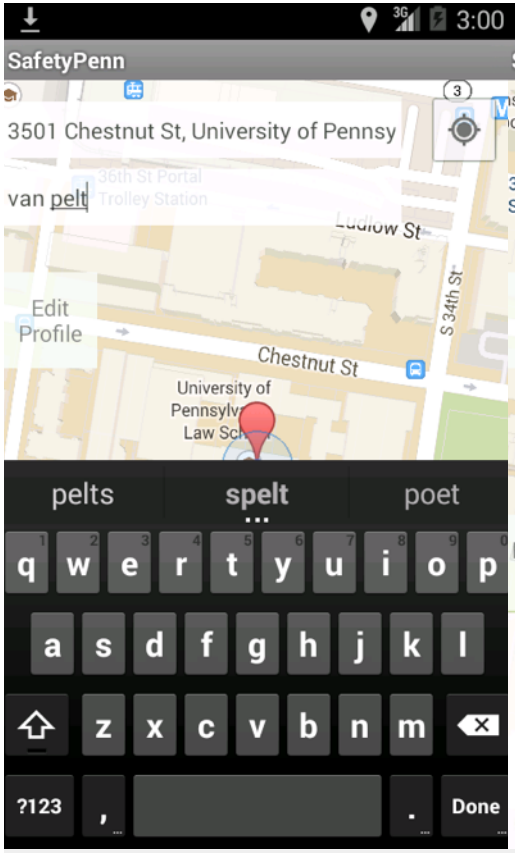
Set Timer By Destination



The other timer option is to set the timer with the calculated walking distance between two locations.

Two text fields will appear on the top of the screen, representing the start and destination locations. The start address is set to your current location by default. You may update the start location by dragging the dropped pin.

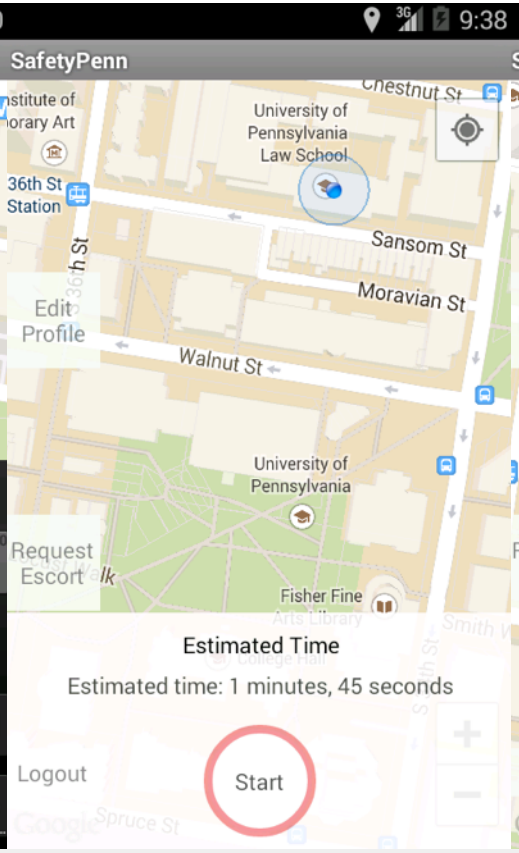
Set Timer By Destination



To set the destination location, press the second text field.

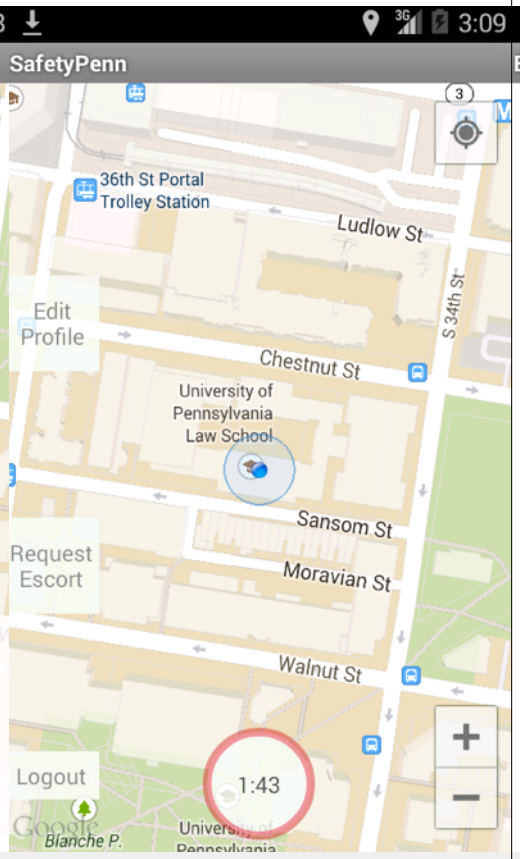
You can type anything from street addresses to hot spots on Penn, such as “Van Pelt” or “Wawa Spruce St”. Make sure these locations are within the boundaries of Penn’s campus!

Pressing the “Done” key on the keyboard will set the timer to the calculated time.



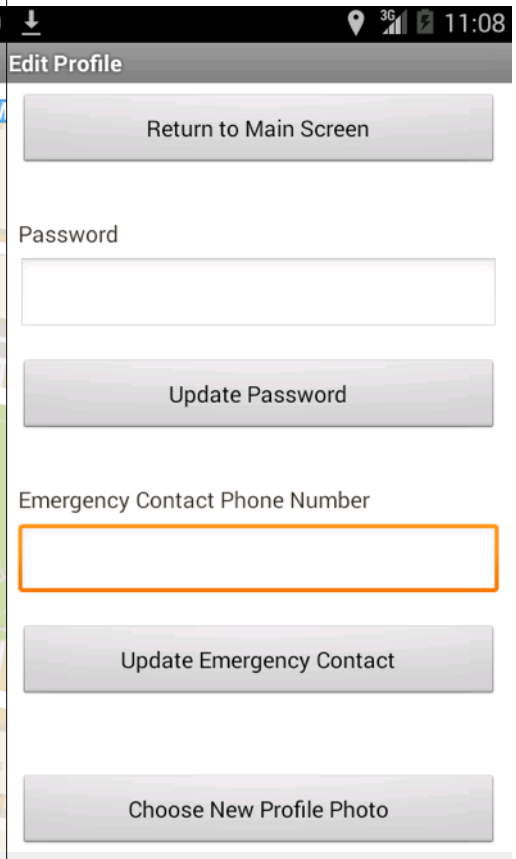
When the calculated timer is ready, the screen will display the estimated walking time from the start location to the destination.

Press Start to initiate the timer.



As before, you can cancel the timer by clicking the bottom button.

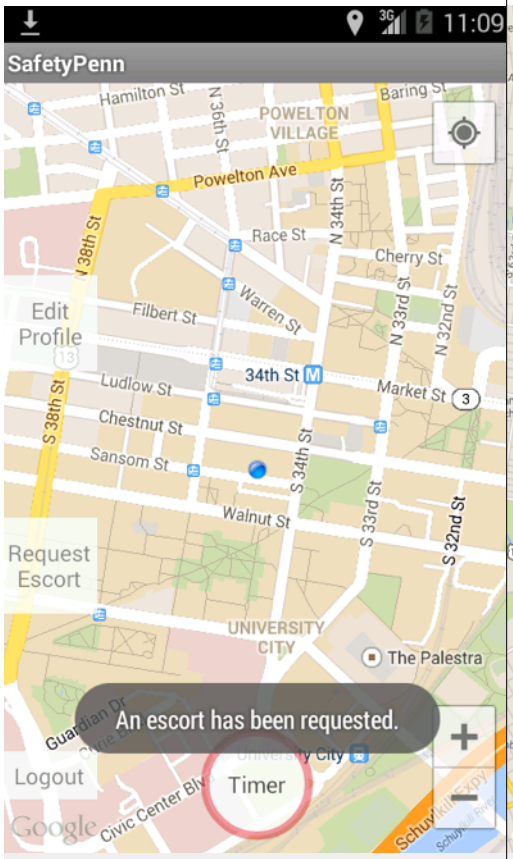
Edit Profile



Pressing the “Edit Profile” button on the home page will bring you here. You can update your account password, emergency contact number, and profile photo.

When you’re finished, press “Return to Main Screen” to return to the map.

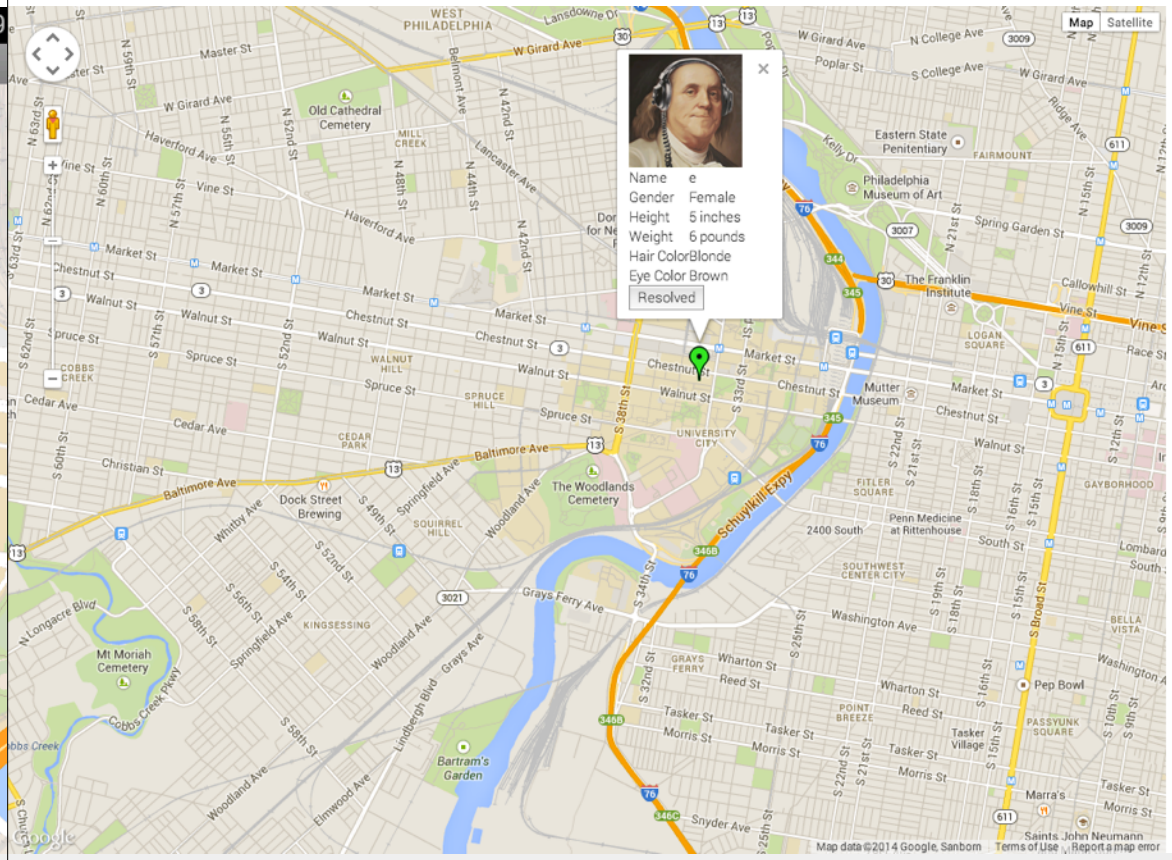
Request For Penn Escort



To request a Penn Escort, press “Request Escort” on the left-hand side of the map.

Pressing this button will dispatch a Penn Escort immediately to your location. You will be alerted with a popup if the request was successful.

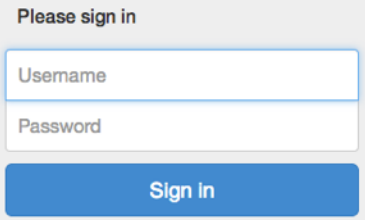
Super User Screen



If the super-user has not logged in, the login screen appears, as shown to the right of this text.

Once the super-user is logged in, he sees this interface. Red markers appear if someone's timer goes off and green ones appear if someone asks for a penn escort.

The notifications also show up in the panel on the left, and the super-user can click on 'Show Marker' to see where the user is, and his description. The information window also has a button to resolve requests.



TECHNICAL DOCUMENTATION

LoginActivity

- UserFunctions class: This class holds methods corresponding to each possible request a user can make (login, register, escort, timer, update picture, update emergency contact, update password, and logout) that create threads to handle these requests.
- LoginAsync class: This is the class that contains the specific thread code that calls jsonParser in the background to login.
- NetworkAsync class: This is the superclass to LoginAsync. It holds static fields such as the different tags associated with user requests and the server URL.
- JSONParser: This class holds the code that makes the connection with the server to send the necessary user request data.
- DatabaseOperations class-:This stores user information in the device's database which allows the device to keep track of whether a user is already logged in (if they close the application without logging out, when they open it back up they won't have to login again).

RegisterActivity

- UserFunctions class
- RegisterAsync class: This is the class that contains the specific thread code that calls jsonParser in the background to register.
- NetworkAsync class
- JSONParser class
- DatabaseOperations class

EditSettingsActivity

- UserFunctions class
- JSONParser class
- UpdateContactAsync class: This is the class that contains the specific thread code that calls jsonParser in the background to change your emergency contact.
- UpdatePasswordAsync class: This is the class that contains the specific thread code that calls jsonParser in the background to change your password.
- UpdatePicAsync class: This is the class that contains the specific thread code that calls jsonParser in the background to change your profile picture.
- NetworkAsync class

DashboardActivity

- UserFunctions class
- DatabaseOperations class
- EscortAsync class: This is the class that contains the specific thread code that calls jsonParser in the background to request an escort.
- NetworkAsync class
- JSONParser class
- Timer class: This starts the countdown timer when the timer button is pressed.
- TimerAsync class: This is the class that contains the specific thread code that calls jsonParser in the background to notify the server that timer went off.

Login Database

- Members Table: email address (primary key), name, password.
- It keeps track of users for login verification purposes.

External System/Non-Android Code

We are using an Amazon EC2 instance to host the login database as well as the PHP code that receives the user requests (through Apache http initiated on Android side) and handles them appropriately. If it is a login request, it simply checks with the local Login database. If it is another type of request, it sends the data to the superuser web server using cURL. The php code then sends a JSON object back to the android side containing success/failure information (example: {"tag":"login","success":1,"error":0,"error_msg":""}).

Web App

For the web app side we used Python along with Flask which is a microframework for Python allowing for rapid development of apps.

Routes.py

This file had all of the major logic dealing with routing urls to functions as well as interactions with the database. It has functions to deal with logging in, editing user information on the databased, adding new notifications to the database, removing notfiications from the databse and adding new users to the database. It interacts with an Amazon SimpleDB databse that has two tables:

1. Members: (email, name, eye_color, hair_color, picture, notifcation_type, gender, hair_color, emergency_number, user_number, weight)
2. Notifications: (email, longitude, latitude, notifcation_type)

login.html

Contains the basic view for logging in.

home.html

Contains the html to render the main console for the admin as well as javascript that deals with setting up the markers on the map, the buttons and the information windows for the markers as well as other necessary logic needed for a usable UI.