I’ve eliminated most of the unnecessary stuff so that there are only 3 mySQL tables left. I’m going to outline the tables and how each of the functions work.

I’m generally not too worried about security; we’re not dealing with super secure information. If they’re little things, like hashing and salting passwords, then do it, but there isn’t much need for 2048 bit encryption. The biggest threat to this system is fools who screw around and accidentally break something, not dedicated hackers.

People Table					
Email	Password	Name	Phone Number	Organization	Captain for events
Email address for the volunteer. Is not null	Password set by volunteer. Is not null	Volunteer’s name	Volunteer’s phone number	Organization that the volunteer is associated with. Can be ‘none’	If null, this volunteer is not a captain. Otherwise, this cell contains an array of the event IDs that this volunteer is captain for.

	Events Table							
Event ID	Event Name	Location	Sponsor	Start Time	End Time	Description	Max Number	What to bring
int ID	Name of the event	GPS location or address. Used to direct volunteers to location	Sponsor for event.	Time PST when the event will start.	Time PST when the event should end.	Description of event	Maximum number of people at the event	Array of stuff to bring (doesn’t have to be an array, just might make it easier for formatting). (eg, shovel, gloves, sunscreen, water)

Association Table(Which volunteers are at which events)			
Email	Event Id	Start Time	Total Time
Email address of the volunteer.	ID of the event that this volunteer is going to	What unix time this volunteer checked into this event. Default to -1 meaning that the volunteer has never been checked in. Equals 0 when the volunteer has been checked out.	When the volunteer is checked out, current unix time-start time = total number of seconds that the volunteer contributed.

Front End Stuff

Signing up a new volunteer

1. Check that email address is not the same as another person.
    a. If email address is same, redirect to login
2. Enter name, phone number, email straight into the table. If name or email is not filled in, ask user to provide email and name.
3. Hash and salt password and enter that into the table. If password not filled in, ask user to set password.
4. Organization is a drop down menu.
    a. Find all unique organizations already in the people table, if the volunteer chooses one of those, then enter that organization into the cell.
    b. If the volunteer chooses ‘none’, cell= null.
    c. If the volunteer chooses ‘other’ then have a text box pop up and enter a new organization in. This organization will then be available for other people to choose.
5. Create a session for web browsers and put the email address as a session variable. For phones, store the email address as a static variable. This isn’t exactly secure, maybe you can find a better way.

Signing in a volunteer.

1. Check email exists.
    a. If not, ask “did you sign up for impact?” If no, then redirect to sign up. Otherwise, try again.
2. Check password against hashed/salted one that is in the database. If wrong, then try again. If wrong 2 times, then ask did you forget the password? If the user forgot the password, ask for their name, phone number, email, and organization, then allow them to set a new password.

Signing up a volunteer for an event.

1. Send the email and event ID to the server.
2. Check to make sure that this volunteer is not captain for this event.
3. Add a new row with provided email and event ID. Start time = -1, Total time = 0.

What if a volunteer wants to back out of the event?

1. Volunteer clicks "cancel RSVP button"
2. Send the email and event ID to a "remove RSVP script"
3. Delete the row that matches the email and event ID if start time = -1

Captain's app differences:

1. If this volunteer is captain, display a "View Volunteers" button.
2. When clicked, go to a page with all the volunteer's names. Display them in red if start time = -1, display in grey if start time = 0, display in green if anything else. (red means never checked in, green means currently checked in, grey means checked out.)
3. Swipe or something to view details, call the person, or send them an emeow.
4. Do the same thing on the website, except also include a CSV list of every volunteer's emeows, which should make it easier for captains to email all their volunteers.

Checking a volunteer in:

1. When captain checks in, find that person's email this event's ID.
2. Send that to the server "checking in" script and set the start time to now.
3. Refresh the page.

Checking a volunteer out:

1. When the captain hits check out, find that person's email and this event's ID.
2. Send that to the server's "checking out" script.
3. Subtract start time from current unix time to get the number of seconds of volunteering. Set total time to this value. Then set start time to zero.
4. Refresh the page.

Displaying an Event

Be nice and display things like directions to the event, have a button to call the captain, or a button to email the captain. Show the description and what to bring. Stuff like that, I think you guys know what to do here.

Back end stuff

Total number of hours volunteered.

1. Sum the total time.
2. Convert to hours.

Check to make sure captains are checking people out.

1. Use CRON to run a check 15 minutes after the end of the event.
2. SELECT * FROM assocationTable WHERE eventID = (the event we’re checking) AND startTime =/= 0 or -1 [yea, the code’s all wrong, that’s your job]
3. Send that table in an email to Chris Apple and tell him to scream at the captain for not checking people out. Use a whip only if necessary.

Reminding people about the event.

1. Use CRON to run this script 15 minutes before the start of the event.
2. SELECT * FROM assocationTable WHERE eventID = (this event)
3. Find all the people’s email addresses and either send them an android push notification or something similar to remind them.

How many hours has this person contributed (sorta front end, you can show them this info)

1. SELECT * FROM assocationTable WHERE email = (this email)
2. Sum totalTime. Display in hours.

How many hours has this organization contributed.

1. Find all unique organizations.
2. SELECT * FROM peopleTable WHERE organization = (this organization)
3. This will return a list of people. Run the previous script to find the number of hours they contributed.