Wildlife Center of VA Volunteer System Setup and Tips

We recommend running the PHP version, which uses PHP and a little JavaScript, along with a MySQL database. The development and testing environment was completed on Amazon Web Services(AWS). It is recommended to stand up the system on an AWS instance managed by the Wildlife Center of VA's tech team.

**Some helpful tips:**

* "Code" (aka password) for current volunteers to create a profile is "wcvcreate".
* All php email communications currently use temporary email accounts. They will need to be changed to the Wildlife Center's desired email account(s).
* Team Lead's information has been prepopulated with SQL insert statements within the Create Statements file.

**Steps for Setup:**

1. Setup an AWS EC2 instance running Microsoft Server 2012 and install PHP v7.1 and MySQL Server v5.5.
2. Connect to your AWS server through Remote Desktop Connection to view the server's "desktop".
3. It is recommended to install Notepad++ for code viewing and editing.
4. Run SQL create statements, prepopulated insert statements, triggers, stored procedures, and events.
5. Set a Windows Task Scheduler to run a couple times a day that runs emailAlertHidden.php. This script emails volunteers that forgot to clock out. An event in the MySQL database runs every 15 minutes to determine if someone forgot to clock out and will set the clock out time to the current time after 24 hours of being clocked in. The emailAlertHidden.php simply emails volunteers that forgot to clock out with link to update the clock out time to the correct time.
6. Setup database and password (two options)
   1. Create database called "wildlife" with password "Twspike1994?" in MySQL. (Recommended for simplicity)

OR

* 1. Create database and password with desired values. Then go through every php file and change current database name and password to chosen values. (Recommended for security)

1. Create a Google (Gmail) account following Setup wizard.
   1. This account will be shared among Team Leads or refer to **10d**.
2. Select "Options" from navigation bar, select "Calendar".
3. On side navigation locate "My calendars" and select the drop down.
4. Select "Create new calendar".
   1. Fill out all required fields (Calendar Name and select appropriate Calendar Time Zone)
   2. **Enable** "Make This calendar public" by selecting the checkbox listed beside this label.
   3. Ensure "Share only my free/busy information (Hide details)" is **not checked.**
   4. **OPTIONAL**:
      1. If this account's credentials are private:
         1. Add each team lead's email address
         2. Under "Permission Settings" Select "Make changes", then click "Add Person". Repeat as needed.
   5. Select "Back to calendar".
5. On side navigation locate the calendar you just created and select the drop down menu beside the name.
   1. Select "Calendar Details"
   2. Locate "Calendar Address:" and make note of the "Calendar ID:"
6. Setup an Google API account through Google's developers console
7. Create a new project, name it accordingly.
8. Select "Library" on API Manager navigation bar.
   1. Select "Calendar API" under "Google Apps APIs"
   2. Select "Enable"
9. Select "Credentials" on API Manager navigation bar.
   1. Select "Create credentials" drop down.
   2. Select "API key" option.
   3. Copy and store this key for future implementation.
10. Located in this folder, open "calendar.php"
    1. On line #283 change this to: googleCalendarApiKey: 'your key here'. (From 15c)
    2. On line #325 change this to: 'your calendar id' (From 11c)
11. Located in this folder, open "calendar2.php"
    1. On line #266 change this to: googleCalendarApiKey: 'your key here'. (From 15c)
    2. On line #296 change this to: 'your calendar id' (From 11c)

**System Maintenance:**

* System maintenance should be minimal
* The occasional upgrade or bug fix may be necessary