Vikram Krishna

What was implemented: Everything but the last 10 points(last bullet point

Database Implementation:

I used 2 different tables within a database to normalize the database, one table stored 3 columns; primary key id, geocode, address, name while another table stored primary key id, geocode id, latitude, longtitude, and time. The mapping of the 2 tables was from geocode column to geocode column, where geocode was a unique integer(came from primary key id) which could be used to have a 1 to many relationship. This allowed me not to store redundant information, and allow for multiple check ins to be associated with 1 name, and postal address.

Code Implementation:

My code is broken up into 3 separate activities, the first activity uses the google api client and associated functions to create,and write to a database, update the current location textview, and current address textview. My 2nd activity deals with the google maps, and uses the standard UI controls of the google map, along with another activity to stimulate the popupwindow. This activity also uses the google api client on the backend in order to determine when to use the popupwindow

To solve the race conditions, I simply used a synchronized block around the location object that is returned by onLocationChanged, I used a global variable that stores the returned location object, hence I just use a synchronized block to set the global variable = returned location, to ensure that the synchronized block is a short segment of code that runs quickly.