Overview of what's been done:

- Backend prototyping is done [that is, *feasibility* prototyping]. Chosen server language is Java, through GAE.
- Frontend prototyping is done. Client's first screen is done.
- Existing CalFit codebase successfully imported and converted into an Eclipse Android project (from .sh project).
- REST schema is decided to be an identical copy of what workouts CalFit currently "saves." That is to say, we're**not** using passing up a constant stream of data a la WaveTI.

Overall, we're on schedule to finish our first iteration within the next 2 weeks. The protocol freeze we talked about will still happen at the end of iteration 1 (due June 22nd).

Design decisions to be made:

- The app's method of talking to a backend is still fuzzy/undecided. Some possible options:
  - **Keep saving as is, but silently save the data online.** When the user "saves a workout" in CalFit, the app will also silently upload his data to GAE. The data would also be silently retrieved from GAE at various points in the code, if there is a checksum mismatch between the internal SQLite data and GAE's datastore. To save the data online for each *user*, we can avoid the concept of forcing the user to enter a unique "username" by using the phone's unique device token.
  - **Use option 1, but make it seem to the user like the data is being streamed to the server.** This way, we'd continue to use GAE, but would hide the "save" functionality from the user. Instead, the user would simply work out, and the app would automatically save the data to the phone internally and also to GAE. We'd need to decide at what points the app would automatically save (eg: whenever the rate of kCal consumption dramatically drops, calculated using an easy derivative).
  - **Use option 2 temporarily, but later switch over to a revised, mobile-friendly WaveTI for true streaming data.** This one's self-explanatory, but also incredibly hard to implement (to my knowledge) using SQL or GAE's datastore.
- The accelerometer->kCal algorithm is broken, assuming the user has no idea that the calculations work IFF the phone is strapped to his body (as opposed to just holding it in his hand). How do we deal with this?
  - **Simply tell the user that the algorithm only works when it's strapped to him.** If we do this though, we can't ensure that there's no cheating (recall that we've incentivized this experiment as part of the actual research).
  - **Fix the algorithm to work regardless of how the phone is placed on the body.** But that will be hard to do in 3 weeks, and certainly impossible for the interns to work out in that time -- we have no knowledge of the algorithm or its research. Also, we'd be recreating existing business logic at that point -- business logic that probably exists in several workout apps on Google Play.