CS 251: Lab 10 and 11: Project Feeder

- Due: There are effectively 3 deadlines for this Nov 1 (worth 4%), Nov 2 (worth 4%), and Nov 5 (final deadline), all at 8pm. See inline for the part deadlines. You cannot take any lateday for the last final deadline. There is no scope for late hours penalty either. Any late submission for the final project will get a summary zero.
- Please write (only if true) the honor code. If you used any source (person or thing) explicitly state it. You can find the honor code on Piazza.

Overview

There are students who start off on the project as soon as it is announced, and then there are some who tend to slack off and even forget about it. There are people who wish they had been reminded of an assignment, or even an exam that they have missed. Now imagine that we had "someone" who can organize all the academic schedules, be it an upcoming exam or a project deadline, and notify you as per your need. We want to be that someone through our "Feed'er" app. As soon as you register on ASC, and download the app, and contact a web server, you will have your customized time table. As the course moves along, it syncs every event related to the course to offer timely notifications. This is one half of the student story.

Then there is another half – the quality of the course. Every semester online feedback is collected for all theory as well as lab courses. But most of the times it's just crying over spilt milk. Won't it be much nicer if the feedback could be incorporated in the same offering? Also, some feedback is temporal which the instructor should be made aware of then and there. Considering all these factors and the advantage of feedback in general, we wish to add a real-time feedback feature to our application.

That's what Feed'er is all about!

Objective

One objective of this lab is to create a robust application. That is, we would like you to focus on polishing working features rather than making more half-working features. Accordingly, minimum working features should be complete to earn marks for extra credit.

Also you should be using `git` throughout your project. That way we will know how often you have been working and what your progress was. This will also enable us to give you marks in case you have worked a lot but could reach only some goals. Also divide your project into chunks so that group members can individually contribute. So the submissions come from different users[1].

There are two majors stake holders in this application - the teaching team (we use the word "instructor" interchangeably below), and the students. Instructors access a web interface to add courses, solicit feedback, place deadlines, and view student responses. Students access a mobile app to register for courses, provide feedback, get notified about deadlines, and so on. The webapp is Django based, and will have a sqlite3 database (running on localhost). All the data requests by the mobile app (Android based) will be made to the Django server which will fetch data from its database and fulfill requests by the app. All team members must work collaboratively on a (very important) private repo such as bitbucket.

A. Django

Current day web application are based on a framework. You can think of a framework as an abstraction which provides several generic functions common to applications to make development easier and

[1] Agreed this can be gamed.

faster. There are several web frameworks available, each with a specific purpose. So while there is no clear winner at the moment, we will be using Django, a python-based web framework, to build a web interface for the backend of our application. Django allows one to develop applications in a modular fashion, and is easier to scale as compared to many other frameworks.

Task 0 - Getting Ready

1. Go through the official django tutorial. Before going for the big beast, first try to create a simple "Hello World" django app. Get yourself acquainted with the basic structure of a Django app.
2. Stick to Django version 1.10 for consistency purposes
3. Browse through all the major python files created and understand the functionality of each of them, especially - manage.py, settings.py, views.py, urls.py, models.py.

Lab 10 - Project Specifications

1. Redo **Task 0**.
2. Setup a **sqlite3** database for your project. You can decide the so called database "schema" on your own, however, you have to satisfy the basic requirements and constraints mentioned in the problem statement.
3. Add instructor login feature for the interface. Instructors should have the option to sign up using their email id or Google/Facebook account (add support for at least one social login), and there should be at least one instructor login with a default password to be specified. No verification is needed for instructor sign ups. (You may optionally add additional instructor logins.)
4. Apart from instructor, there should be a special admin login. She alone should be able to add courses, and enroll students (to be selected from a list of registered students) for courses. Each newly added course should have, by default, two feedback forms: midsem and endsem course feedback, and two deadlines: midsem and endsem exam dates. You can decide the format of these feedback forms (see below).

   Important: **This part – the admin interface – is due on Nov 1.** Keep in mind that we will grade the admin interface only on the basis of the submission made on Nov 1, so make sure you think things through.
5. On login, the instructor should have four basic features on the welcome screen viz.
   - Add Feedback: The mandatory fields should be course code, feedback name, deadline and questions. For questions, the instructor should be given option to questions asking for ratings on a scale of 1 to 5 (5 is best). Any instructor can add feedback for any running course, no check needed on the instructor. We assume that the instructors are trustworthy.
   - Add Assignment deadlines: The mandatory fields should be course code, assignment name and deadline. Note that assignment deadlines are just to serve as a reminder. No submissions are expected on the app.
   - View Feedback: Instructors should be able to select the feedback and view (anonymous) results in two formats: individual responses of all students in a table and aggregate results for objective questions. Do not mention student details, only the responses. Also the table containing the feedback should not contain any student information.
   - View Deadlines: Show running and finished deadlines separately.
6. It's important that you handle all relevant invalid cases like illegal user access, and database errors. For example, it is illegal for the web server to accept requests from non-authorized devices (such as the internet). (You should of course feel free to keep it on during the debug phases.)

Extra Credit

1. You may add support for other question formats too. We expect that :) Overall we expect the feedback to be well thought of and your creativity is useful here.
2. You can use graphs or charts to show the aggregate results in a meaningful manner.

B. Android

The world is contracting with the growth of mobile phone technology. One of the most widely used mobile OS these days is Android. Android is a software stack, comprising not only the operating system but also middleware and key applications. You will build an Android app that serves the purpose of giving feedback and other book keeping tasks.

Task 0 - Getting Ready

1. Install Android Studio from here
2. Get started with "Building Your First App"
3. Understand the concepts of Activity and its life cycle, App Resources and Layouts
4. Stick to Android Ice Cream Sandwich as minimum SDK for consistency purposes

Lab 11 - Project Specifications

Name your app and the Android Studio project as `FeederXY` where `XY` is your group number (e.g, group 07 will name their app as `Feeder07`). For connecting to the server, assume the django server will be running on `localhost:80XY` where `XY` is your group number. We will be testing your app on an emulator for purposes of standardization and grading but your code should be working on an actual phone. (We can ask you to show it during the final viva.)

Minimum features

1. Add student **login** feature for the interface (for the purpose of this lab assume the credentials are given to the student in the form of the admin process, ideally we expect this to be the ldap password). This authentication should be through the django server. The student should not be prompted for logging in until he logs out.

   Important: **This part is due on Nov 2.** We will not evaluate this part again.
2. Also include a feature to **log out** of the app.
3. After logging into the app, the student will be shown a **Calendar View** which should at least support the following features. We recommend you to use external libraries (e.g., Caldroid).
   (a) The dates having any event (feedback or assignment deadlines) for the registered courses of the student should be highlighted. You should use different colors to highlight the events for different courses (you also need to handle cases where there are multiple events on the same date).
   (b) Tapping the date should reveal (calendar should still be visible) the list of events for the date. Event in the list should be clickable as necessary.
   (c) On clicking the feedback deadlines, the student should get the form to fill that feedback only if the student hasn't already filled it. Otherwise the option is disabled. The feedback form should not be a simple Web View but should have UI elements provided by Android.
   (d) On clicking the assignment/exam deadlines, the student should get details related to the deadline.

Extra credit features

1. There should be an option to filter the events shown on the calendar based on criteria such as course code, feedback/assignment deadline, etc.
2. During the interim when the connection fails (e.g. lack of wifi), there should a local cache of the data about the registered courses, deadlines etc. and this cache should be updated accordingly when the device is able to connect to the server. It should still be possible to give feedback, for example, in an offline manner.
3. Add an interesting logo to your app

Submission Guidelines

1. The reflection essay is super important for these two labs and should include all aspects of this lab and also relate to any previous lab that made this lab useful. Include both an executive presentation (upto 5 slides), and a LaTeX report containing pictures.
2. When you submit, please document individual percentages such as Student 1: 80%, Student 2:100%, Student 3:10%. In this example, the second student will get full marks (10/10) and the first student will receive 8/10.
3. Do include a readme.txt (telling me whatever you want to tell me). Do include group members (name, roll number), group number, honour code, citations etc.
4. Include your project's bitbucket link in `readme.txt`. Include only source files for Lab 10 and Lab 11.
5. The repo should contain three sub folders `lab10` and `lab11` and `cs251project`.
6. The sub folder `lab10` should contain everything related to Lab 10 and only to Lab 10. Submit on moodle as well as keep it on bitbucket.
7. The sub folder `lab11` should contain everything related to Lab 11 and only to Lab 11. Submit on Moodle as well as keep it on bitbucket.
8. The sub folder `cs251project` should contain everything related to the entire project but divided into two parts `django` and your entire Android Studio project folder \verbFeederXY+.
9. The folder and its compressed version should both be named `project_groupXY` for example folder should be named `project_group07` and the related `tar.gz` should be named `project_group07.tar.gz` while submitting on Moodle.

How We will Grade You

Due November 1 and Nov 2. Marks aside, weightage equal to two labs.

1. Admin interface with appropriate login [7 Marks] (Due November 1).
2. Functionality to add courses (with default feedback forms and deadlines) and enrolling students for admin [14 Marks] (Due November 1).
3. Android Login [15 Marks] Due November 2.

Due Nov 5. Extra credit points are available to you only if you get the basics right.

1. 15 marks are reserved for the reflection essay and the presentation. Don't take this lightly because many of the subjective elements below will be biased by this document. (Hint: Do NOT start working on this last.)

2. Django [60 + 25 Marks]

   - Instructor interface with sign up and sign in features [8 Marks]
   - Functionality to add feedback and deadlines for instructor [20 Marks]. Robustness of software (e.g. to check illegal accesses) will be part of this and the next part.
   - Viewing results of feedback and schedule of deadlines [20 Marks]
   - Overall aesthetics, UX of the webapp [12 Marks]
   - **EXTRA CREDIT:** [25 Marks]
     - Support for additional question types [upto 15 Marks]
     - Aggregate results as graph/chart [10 Marks]

3. Android [50 + 25 Marks]

   - Log out feature [2 Marks]
   - Calendar with events highlighted appropriately [10 Marks]
   - Displaying a list of events on clicking a date [6 Marks]
   - Clickable properties of the events [6 Marks]
   - Working Feedback form [12 Marks]
   - Displaying Assignment/exam deadline details [7 Marks]
   - Design, robustness, and aesthetics [7 Marks]
   - **EXTRA CREDIT:** [25 Marks]
     - Filters for the calendar [8 Marks]
     - Local cache of data [15 Marks]
     - Logo [2 Marks]