Penn Course Review Android Application

CIS350 - Software Engineering & Design
Jinyan Cao, Connie Ho, Charles Kong, Cynthia Mai
Client: Amalia Hawkins, Kyle Hardgrave (PennApps Labs)
Project Lead/TA: Zach Zarrow
Project repository: https://github.com/jinatonic/Penn-Course-Review-Mobile-Beta

Private key: ?token=cis350a_3uZg7s5d62hHBtZGeTDl

User Stories

From app startup

- User signs in using Pennkey and password **(4 points)** (H)
    - add Pennkey authentication
    - definitely a technical challenge
    - several people working on it
    - **use webkit to use authentication and web query?**

From main search page

- User wants to see reviews for a specific course using course number: types in course number, views page of reviews for all professors of all past offerings of the course. **(4 points)** (H)
    - Create results page, dynamically populate sample entries into scrollable grid **(2 point)**
    - Search functionality **(2 points)**
        - Dynamically generate query URL based on query-type
        - JSON parsing
        - Populate local database (sqlite in Android) with results
- User wants to see reviews for a specific course using course name: types in course number, views page of reviews for all professors of all past offerings of the course. **(2 points)** (H)
    - Change search query from course number to course name
    - Can restrict to most recent X years if space is an issue
- User wants to see reviews for a specific professor: types in professor name, views page of reviews for all past courses taught by professor. **(2 points)** (H)
    - Change search query from course number to professor
    - Modify results page for professor results
    - Use search by course number as a reference/template.
- User does not know the full name of a course but wants to see reviews for it: types in course id or a portion of the course name, dropdown suggests course, user selects, views page of reviews for the course. **(2 points)** (M)
    - Auto-complete search text

        - Based on a local database table, intermittently send async query to server for a list of courses, professors, departments, etc. that are available for searching and update the table in database
        - List of autocompletes should be ranked alphabetically or by some kind of relevance system
        - PennApps does autocomplete by downloading entire JSON file of possible search terms on first site launch
        - autocomplete.json file, not part of API
- User wants to see reviews for all courses offered by a specific department: types in department name, views page of reviews for all past courses offered by the department. **(2 points)** **(M)**
    - Only possible if PCR API supports it
    - Can default to sort courses by quality

Course page

- Within a course page, user wants to sort offerings of a course by a specific field. **(3 points)** **(M)**
    - Select field by up/down arrow next to it.
    - Highlights the field that the data is being sorted by.
    - Sort within a course page by:
        - professor quality
        - course quality
        - professor difficulty
        - course difficulty
        - amount of work
        - professor name
        - semester and year offered (chronologically)
    - Dynamically repopulate results
- Within a course page, user wants to filter offerings of a course using same fields as mentioned above for sorting. **(2 points)** **(M)**
    - Support filter functionality through DB queries (simple order by parameter)
    - e.g. filter out certain professor, above a certain rating, etc.
- Within a course page, user wants to read more detailed reviews (put together by Penn's review committee) regarding a specific offering of a course from the course review page **(2 points)** **(H)**
    - Can click on a specific offering of a course which takes you to a new page that has the information

Professor page - similar in implementation

- Within a professor page, user wants to sort courses a specific field. **(2 points)** **(M)**
    - Sort within a professor page by:
        - course quality
        - course difficulty

        - amount of work
        - semester and year offered (chronologically)
- Within a professor page, user wants to read more detailed reviews (put together by Penn's review committee) regarding a specific offering of a course from the course review page. **(1 point)** (H)
- Within a professor page, user wants to filter out courses offered for the current/next semester **(2 points)** (M)
    - Need to work with Penn registrar database/API
    - Possibly incorporate "course cart" feature (like a "Favorites" list)

Department page - In the future, if/once PCR API supports it

- Within a department page, user wants to sort courses by a field. **(2 points)** (L)
    - Sort by:
        - average course quality
        - average professor quality
        - average difficulty
        - average amount of work
        - professor name
        - semester and year offered (chronologically)
        - course name/id listing (numerically)

Comparisons - In the future, pending screen size restrictions

- User wants to compare two (or more) professors for the same course. **(3 points)** (L)
- User wants to compare two (or more) courses for the same professor. **(3 points)** (L)

UI Design

Basic - functionality (H)

- Create main PCR page, include search bar, helpful hints, etc.
- Create results pages dynamically populate sample entries into scrollable grid.
    - Use paging for large result sets
- Splash screen
- App logo

Advanced - more visually appealing (M)

- Background image
- Simple main page
- Catchy logo
- Smooth metro-style grid boxes (for search results)
- Smooth transitions between pages (perhaps use loading bar while waiting for query)
- Side-to-side comparison
    - New UI page for this, not sure if we have enough room on a mobile device

- Incorporate pictures?
    - Possibly professor pictures or department logos
- Possibly have a course cart of some sort (only if we can get the next semester's offering of courses)
- Graph feature? Will be very challenging to do

Extra Notes

Possible extra features (L)

- Visualization
- Tab switching
- Use of email to get information
- Advanced search options

Technical challenges

- Get size of the database, feasible to store whole database locally?
    - or possibly store all the data that the user searches for, which are relevant to the users

1st iteration, 1st meeting (2/19):

- By next week -- basic mockup of UI, basic idea of how the API works, maybe work on basic query
- Prioritize the user stories, send it to Zach
- Should be using test driven development