**Career Find**

Our client is a local middle school. They have given us an unlimited budget to build a career counseling application.

**Prioritized User Requirements**

1. Base System.

* A primary view of 16 career clusters allowing users to drill down into additional career information.
* A content management system allowing users to create, read, update, and delete:  
  - career clusters (name and icon)  
  - keywords; keyword-career and keyword-celebrity associations  
  - career names, descriptions, and salaries  
  - career DITL  
  - celebrity names, articles, photos  
  - <chatbot scripts>
* A database capable of storing the content listed above.
* An appealing and simple interface suitable for middle school students.  
  - uses color template provided by client (navy blue and neon green)  
  - presents information in digestible chunks  
  - presents information in ways that stand out and are engaging, like support for pictures, vibrant colors, large print text, fun fonts  
  - minimalistic design so that interface is ‘quick’ and ‘easy to use’

2. Multi-role User Account/ Profile Functionality.

* A user login page that accepts and validates appropriate credentials (username/ password… e-mail/ password).
* Assignment of users to overlapping roles:  
  - student  
  - teacher  
  - administrator  
  - counselor
* Assignment of users to overlapping user groups (a class, a grade, a team, etc).
* An interface to create, update, and delete users.  
  - all roles may have limited ability to manage their own account  
  - teachers will be able to create, update, and delete student accounts within their own classes; reset the password of students within their classes as well  
  - administrators can create, update, and delete all user types; reset the passwords of all other users as well
* Counselors should have access to content management interfaces.
* Student profiles should be able to store keyword preferences and allow students to delete these stored preferences. These keyword preferences should allow students to quickly access system information via keyword associations.
* Student profiles should provide an associated queue allowing students to queue career clusters/ careers for future viewing. Students should be able to prioritize and remove enqueued items.
* Student profiles should allow students to track what information they have and have not yet read in the system.
* <Students should be able to bookmark/ favorite/ star careers>

3. Gamification/ Reward/ Positive Operant Conditioning Functionality.

* A point based system to support progress tracking and reporting for:  
  - individual students  
  - classes of students  
  - arbitrary groups of students
* An interface for teachers to manage reward systems… reset point count, set bonus multipliers, activation over certain time periods, deactivate point count, enable randomized point rewards… set point goals
* Information radiators such as progress bars so students (and teachers) can easily see whether goals are being met or not.

4. Chatbot Career ‘Counselor’.

*NOTE: A real chat bot seems impractical for a number of reasons in addition to us likely not having time to implement one. If we give the client a chat bot, who is going to write and maintain the chat bot scripts? We do not have the content to do it and I am guessing school faculty doesn’t have time to program a chat bot either. If they only have time for simple exchanges like “What’s your favorite subject?” “Math” “You might like Finance & Accounting” we could probably provide that functionality through on screen notifications or something similar. Maybe instead of saying ‘We won’t build a chat bot’ we build this instead and see if they like it? Maybe all we really need here is this simple interface. Later on an API with sophisticated text recognition capabilities might be able to match student responses with keywords in the system… that might be us that implements that or someone else.  
  
To summarize: -an initial prompt, we parse the response for recognized keywords, we return the career cluster with the most keyword hits and enqueue it. If there are no hits we log the response for review and suggest trying again.*

* Counselors should have the ability to create, read, update, and delete dialogue tree scripts for a chatbot that will interview students.
* [Counselors should be able to target scripts for individual, classes, and arbitrary groups of students]
* Chatbot scripts should be able to parse student responses to identify potential keywords of interest and associate them with the student profiles.
* Student profiles should be able to track which chatbot scripts have been completed and which have not.
* Each time a student logs into the system an incomplete chatbot script should start if one is available.
* Upon completion of a chatbot script, the student should be given the option to complete the next incomplete chatbot script or to stop engaging with the chatbot.
* Students should be able to engage the chatbot at any time and be offered a randomly selected chatbot script irrespective of whether it has previously been completed or not.
* Counselors should be able to review anonymized logs between the chatbot and students to facilitate development and maintenance of chatbot scripts and other content.