**Examlet**

**Complete site-map**

**Version 2**

As a site map this should look something like this:

Home:

1. Info/about
2. Login / Registration / (Auto login) (For steps to sign-up, please hold Ctrl while clicking here on the number: **1** )
3. Following page:
   1. Answer
      1. Test-setup
         1. Which exam?
            1. (by default, only offer the ones the user scheduled for themselves during sign-up)
            2. NB: A pre-made exam will already be designated as MCQ, Calculation-based, short answer, or essay based.
         2. Question selection options
            1. Number of questions

Let users pick exactly the number of questions they wish to comprise their exam (please have a classy way of entering this information, not just a crude text box.

* + - * 1. Total length of the exam

Let users enter the total number of minutes their exam is to comprise (please have a classy way of entering this information, not just a crude text box.

* + - * 1. Would you the user like to receive answers after each question, or at the end?

If the user wishes to receive the answer after each question, this will be called “Revision mode”.

If the user wishes to receive all the answers at the end of the exam, this will be called “Exam simulation mode”

This also has implications for the ‘feedback’ section.

* + - * 1. Price

Free only

Mix of free and premium

Premium only

* + - * 1. Question selection algorithm

Specific questions from that exam (no ‘Artificial Intelligence’)

Intelligent selection (AI)

Where the user has attempted questions previously:

Tally and identify the keywords where the user’s performance is below the overall mean for questions with those keywords

Tally and identify the keywords where the user’s performance is below the overall mean of *their cohort*’s mean score on those same keywords

Increase the likelihood of high-ranking questions with these applicable keywords populating the user’s test-set. 1 of every 3 questions should involve these keywords, until the user’s average has risen to at least the mean, on both situations listed above.

If the user asks to have the answers delivered at the end, the full number of questions within the test bank will be selected before the exam starts. At the end of the exam, the answers to the questions will be displayed, one at a time, followed by a feedback survey, then the answer to the next question, and so on.

If the user asks to have the answers delivered after each question, a feedback survey will be displayed after each question, before asking the next question. Each question will be selected individually (dynamically), and selection of the ‘next’ question will consider all feedback responses to date – including the most recent question.

* + - * 1. There should be a check box “Remember these settings” which causes these settings to be selected by default, the next time the user answers questions.
        2. At the bottom, a button to proceed to testing should be available. When clicked:

Question selection algorithm begins:

If the user has elected to conduct a non AI-exam (e.i)

All the questions used for the test should be strictly drawn from the questions available for the specific subject the user is enrolled in (within the specific university – **don’t forget this important point: (3.c**)

If the user has selected an AI exam

65% of the questions used for the test should be strictly drawn from the questions available for the specific subject the user is enrolled in (as above)

30% of the questions should be populated with questions which are **low** in **competent keywords (vi.4.d)** and **high** in **incompetent keywords** **(vi.4.e)**

**There needs to be a solid/reliable search algorithm performing this function**

This may assist in reducing the incidence of duplicate questions, from the respondent’s perspective

This will target the learning of respondents, so that their deficiencies are addressed adequately, by allowing high-quality questions from around the world which are rich in ‘incompetent keywords’ to populate the question set for the examination.

A search should be performed looking for questions to which addresses the ‘incompetent keywords.

In order to be considered for the search, the question quality must be at least 60%. The upper limit to the question selection will depend on whether the user has requested free-questions only (minimum question quality = 60%, maximum question quality = whatever the ‘premium quality % cut-off’ is set to), mixed (minimum question quality = 60%, maximum question quality = 100%), or premium-questions only (minimum question quality = whatever the ‘premium quality % cut-off’ is set to, maximum question quality = 100%)

In order to be considered for the search, the question must contain two previously assessed keywords: one keyword marked as ‘incompetent’, and another keyword which may be marked as either competent or incompetent.

First, a question addressing two of the most incompetent keywords’ with the most severe deficiencies the question bank for specific subject the user is enrolled in (within their university) should be searched.

If a suitable question cannot be found from the above location, then the search should be broadened to include all subjects within the respondent’s degree, but within that same university

If a suitable question cannot be found from the above location, then the search should be broadened to include all subjects within the same degree, in other universities (in the same country as the respondent)

If a suitable question cannot be found from the above location, then the search should be broadened to include all subjects within the same degree, in other universities (anywhere around the world)

If, after all that, a question cannot be found, there would be one of two causes: 1) The site has only been recently launched, thus the question bank hasn’t been populated yet, or 2) There is a spelling/grammar error in keywords somewhere.

In this case, skip this combination of keywords, and try to find a new combination of keywords, starting again from here (vii)

This completes the search for 1 question using the AI question selection algorithm.

The questions comprising the final 5% should be conducted exactly the same as (b), but the difference is that instead of the respondent’s incompetent keywords forming the basis for the search, the incompetent keywords from the respondent’s entire cohort should be used for the question search. Please see here (vi.4.e) for more information.

See this point for whether the user wishes the exam to occur in revision or exam simulation mode (c)

Revision mode:

Each subsequent question is selected into the question set for the exam only after the previous question has been attempted and feedback given. In particular, the overall tallies for competent and incompetent keywords should be reassessed after each question.

Exam simulation mode:

In this case, all the questions for the question set should be gathered at the start of the exam. This is necessary as the user won’t be giving feedback after each question – their question will be marked, and feedback will be given, all at once – at the end of the exam.

Screen goes black except for a white column occupying the middle 50% of the page. The lateral edges of the white column should fade to the black background over a moderate distance. It is intended that this appear as a white sheet of paper which fades into the black background, with nothing else visible on the screen except the contents of the question – as it would appear on an exam paper.

Options like pause, exit, etc are only accessible by moving the mouse to the top of the screen, where these basic menu items appear

Text in the middle of the screen, saying “Ready to start the exam?”

A button in the middle of the screen saying “Start”

Start leads to a countdown from 10, 9, 8 etc

Test beings:

The question is displayed with the Question vignette positioned against the top left margin (the margin should be about 2-3cms from the edges of the white column

If the question style is Multiple Choice, then the 4 (or 5) options available as responses should be shown. If the question style is other than Multiple choice (essay, calculation, short answer, instructional), then no avenue for a user response should be provided

FOR CLARIFICATION OF QUESTION TYPES, SEE “EXAMLET – QUESTION STYLES”

If the exam is being taken in “Revision mode” **(Ctrl+Click this: d)**, then there should be a button which causes the correct answer/explanation to be displayed after each question is attempted “Show Answer”. (If in Exam Simulation mode this information should be provided at the end of the exam)

For Multiple Choice questions, a green tick should be displayed next to the correct answer. If an explanation for why each of the individual options is correct/incorrect is provided, then these should be displayed next to the option. The overall explanation should be provided underneath all the options. Viewing the answers like this should not cause any shifting of the other text on the screen.

For other question types: The overall explanation/example response should be provided underneath all the options. Viewing the answers like this should not cause any shifting of the other text on the screen. In the case of calculation – based exams, the example response will require the display of a scanned/photographed image of calculations worked out manually on paper.

There should be a button which, when clicked, expands to reveal the full reference citation provided by the author. If the reference is a textbook, then the Amazon affiliate advertisement should be displayed (as an image showing the cover of the reference text). If the reference is a doi or other journal article, then this box should expand to reveal hypertext which says “Open the journal reference in a new window”.

When clicked, a page should be opened whereby the automatically enters the string of the DOI listed by the author, into this web-page: <http://dx.doi.org/>

For example, the DOI : **10.1093/bioinformatics/btg1041** should lead to the following window being opened, when the hyperlink referred to earlier (Ctrl+Click this: c) is clicked <http://bioinformatics.oxfordjournals.org/content/19/suppl_1/i302>

If the student got the question right, then the keywords associated with the question should be entered under a column called ‘**Competent Keywords’** under the current user’s database entry. Similarly, the keyword should also be tallied under a column for the subject as a whole, marked ‘Competent keywords’

If the student got the question wrong, then the keywords associated with the question should be entered under a column called ‘**Incompetent Keywords**’ under the current user’s database entry. Similarly, the keyword should also be tallied under a column for the subject as a whole, marked ‘Incompetent keywords’

Even if the student elects to do a Non-AI exam (e.i), the keywords should still be recorded against the user’s database table – as these statistics will be used for reporting which keywords certain universities etc are competent or deficient in, overall. This, in turn, will be used when students conduct an AI exam (e.ii), whereby the overall keyword deficiencies of that university are considered when choosing which questions are to form the examination.

There should also be a small button which, when clicked, exposes the overall tallies for each response. The overall number of respondents should also be indicated

There should be a tree , (minimised initially), which when maximised, provides branches each of the following groups, where there is at least 1 other respondent:

By Institution type (School/University) (each trunk item itself should be selectable; alternatively users may select one of the following branches or sub-branches) 🡪University/school name 🡪 By Degree name (only applies to Universities) 🡪By anticipated final year of study (can apply to both school and universities) (FOR CLARIFICATION, SEE EXAMLET – EDUCATIONAL REGISTRATION REQUIREMENTS)

By Sex

By Age

By Country 🡪 by State

There needs to be a mandatory feedback box, where users evaluate the question and it’s explanation (If in “Revision” mode, this survey should be displayed simultaneously with the answers, after each question. If in “Exam Simulation” mode, this survey should be displayed with each corresponding answer, when they all are displayed at the end of the exam).

Feedback polls (out of a total of 4, where 4 is high/excellent)

Quality of the question (**alpha**)

Quality of the explanation (**beta**)

Amount learned from this question (**gamma**)

In all cases, whether Revision or Exam Simulation mode, the user MUST provide this feedback in order to view the next question.

These statistics should be recorded against the question in the database, and the sum of these tallies for the question should used to establish if the question should remain a free question, if it should be demote to become removed from the question bank, or if it should be promoted to become a premium question.

(The final feedback item – in retrospect, the similarity of given question to one that appeared on an exam, will be submitted either via mobile devices or via a separate section of this website. It will be called **delta**)

There should be a comments box, which is initially minimised, but expands to reveal other user’s comments and allows the user to input their thoughts. Even when minimised it should display “Comments (x)”, where x are the number of comments recorded against the question.

There should be a timer, viewable from the hidden menu bar (**Ctrl+click this: ii)**. According to user preference, the timer should either display the time remaining for the entire exam, the time remaining for the current question, or “Average time taken per question vs. Average time permitted per question (total exam time divided by total questions)”.

If the user is in Revision mode, the timer should pause once they’ve hit the “Show Answer” button. The timer should stop completely once the final question is completed in “Exam Simulation” mode.

There should be a button to progress to the next question, which clears the screen and displays one single question on the screen, as per **(Ctrl+click this: 2)**

If the user elected to undertake an “Exam Simulation”, then all the answers to all the questions, and the correct responses, references, statistics (via the tree), feedback, and comments etc, should be displayed individually, but in sequence. The reason for displaying the answers individually here, rather than as a list, is that I need to force the users to provide feedback on the questions. This feedback is absolutely core to the site’s functionality. Denying them access to the next answer in their exam is the best way I can think of to encourage them to fill the feedback section in.

* + 1. Account management
       1. Transfer money in/out
          1. Deposit in (default)
          2. Withdraw
       2. Purchase a membership
    2. Statistics
       1. Default to user’s personal statistics
       2. Allow simple options allowing the ability to view all the different types of stats we discussed
  1. Write
     1. Question regurgitation
     2. Pick-up a question from the holding bay
     3. View question performance and usage
  2. Discuss
     1. Chat forum
     2. Structured feed of comments on questions the user has either written or commented on
  3. Other
     1. User options
     2. Account management
     3. Schedule a new exam
     4. Statistics

Examlet,

Steps in the educational component of user registration

Version 2

Steps to sign-up

1. Users search for their university (in most cases, they should be able to find it in our list)
   1. In a drop down list under the search form the user types in, the top 4 universities which contain the text the user has entered displayed in real-time, even before the ‘enter’ key is pressed. The portion of the text identified in this list should be ‘bolded’. The user should be able to select the university from this list, but also hit the ‘enter’ key and be brought to a search page which displays all the matching universities.
   2. If, after a search has been completed, the university cannot be found: (note, users must search first)
      1. Prompt the user to submit the details of the university
      2. The site should then send the administrator(s) an email and provide an inbuilt notification that a ‘new university’ request has been submitted
      3. The administrators should be able to review the details of the request and either:
         1. decline the request
         2. accept it,
         3. make modifications to the information the user submitted, before accepting it
      4. If accepted, the university should be added into the database, and the next user should be able to find it by searching
   3. Users should be able to flag the university name as inappropriate (offensive, incorrect, etc), and the administrators should be notified of this when it happens.
   4. Administrators should be able to delete, modify or add university details at any time.
      1. If deleting a university, they should be asked to confirm they wish to perform that action.
   5. Each university should primarily stored within the database as a number, so that modifying the university name does not affect the ability to index questions to the university
2. They search for their degree name
   1. All the points in (1) above apply, except:
      1. They should be able to search by degree *name*, eg Bachelor of Science (Biol), or by the degree *code* the university uses for that degree (eg BS12)
      2. This search shouldn’t search for ALL degree names in the database – it should only search for degrees which have been added under the heading of the university the user selected in (1)
      3. The addition of a degree name to a university should not require any intervention from the administrator – it should just happen automatically.
      4. Once per day the administrator should receive 1 email which allows the admin to see at a glance all the degree names which have been added to the site by users.
   2. When a user adds a degree name, they should be prompted to add the degree code if they haven’t entered one, but they should also be able to select an option like ‘There isn’t a degree code’ if that is the case.
   3. Users should also select their year that they anticipate will be their final year of the degree/program (not the year of graduation – these can be different)
3. They search for the names of the units/subjects they are currently enrolled in (eg, statistics 101, biology 103)
   1. All the points in (2) also apply for subject names.
   2. Users may be enrolled in more than one subject at any one time
   3. This search shouldn’t search for ALL subject names in the database – it should only search for subjects which have been added under the heading of the university the user selected in (1) (note – a humanities student may do a science subject as an elective subject. The search for subject names should draw from all subjects available at the university (1), not all subjects available within a given degree(2)
   4. When a user adds a subject name, they should be prompted to add the professor or chair of the subject if they haven’t entered one, but they should also be able to select an option like ‘I don’t know / There isn’t one’ if that is the case.
4. Based upon the selections made in (3), the site should proactively suggest exams which have been created for the subjects the student has chosen.
   1. This is an opportunity for the user to select which sets of past questions they will get exposed to.
   2. Users should be able to select check-boxes for the ones they are interested in.
   3. As with (1), (2) and (3), they have the option to create a new exam.
      1. They should select whether the exam is multiple choice, short answer, mathematical/calculation based, or essay based, or instruction-based
      2. Note: I also want the site to allow students to pass on what was in practical/instructional exams, for which there is no correct answer, instead students are asked to practically demonstrate something while being observed by an examiner. These items will simply serve as a tip-off about what to expect in the practical exercise.
   4. They should also enter the date and time of the exam
      1. This particular information will vary from year to year, and won’t be relevant for successive cohorts. Thus, this data should not be associated with the exam name, but rather it should be associated with the user’s profile and used to ‘time’ the automated emails the user receives before and after their exam, etc
5. In addition in the new version of this document, and not following from the previous information in sequence
   1. Facebook Connect:

Where possible, Facebook Connect should be utilised to mine the user’s birthdate – particularly their birth year. I do not feel that we can trust users to enter in their correct birthdate when asked routinely.

Examlet

Explanation of the different ‘question’ styles

Version 1

* Multiple choice questions
  + These will be the only type of question where the user actually interacts with the site/database.
  + Users enter in their answer, and feedback/explanations are given
* Essay style questions
  + Users won’t interact with this type of question.
  + The essay instructions, as occurred in the exam, will be presented to the student.
  + If the student wants to, they can write their answer on a piece of paper.
    - There won’t be an option for the student to submit their response to the site, or anything like that.
  + There should be an button that the student can click, when they are ready to, which displays the answer that the user that picked the question up from the ‘holding bay’ wrote
* Short answer
  + The same situation as for essay questions applies here.
  + If there are a number of short answers questions which all relate to each other, these questions should be all be asked at once, stacked on top of each other on the same page. When ‘show answer’ is clicked, all the answers to all the questions (submitted by the student who picked the question up from the holding bay) should all be shown at the same time on the one page.
* Calculations
  + The same situation as for essay questions applies here.
  + The difference is that the response from the student who picked the question up from the holding bay will be a calculation.
    - When ‘show answer’ is clicked, the scanned image of the ‘model answer’ should be displayed
  + If there are multiple associated questions asked, the questions should all be asked at once, and once the ‘show answer’ button is clicked, all the answers should all be shown at once (as with the short answer questions)
* Instructional/practical
  + These should be treated the same way as short answer questions. The answer the holding-bay retriever gives may be the marking criteria, if they have access to it, or any other suggestions about how to approach the task, etc. I’ll leave it up to the holding-bay retriever to type whatever information they feel is appropriate/beneficial as the answer.

Examlet,

Business-model site requirements

Version 2

The site ‘currency’ will need to be either: Dollars, Credits or Karma.

1. Need the option to switch between them
2. Site should start out without currency at first, as a pilot
   1. First type of currency will involve Karma
      1. an unmonitored system, where participation in different areas of the site yields a certain number of credits, which can be exchanged for access to premium questions and other site-features (such as in 3a.iv.1)
   2. Later progress to either:
      1. Exclusively dollars
         1. As with “freelancers.com” website
         2. With transaction fees generated
      2. Exclusively credits
         1. A monetary currency which is subject to a various conversion ratios, depending on the quantity of credits purchased in one transaction.
         2. With transaction fees generated
      3. A primarily Karma-based system, where users can ‘buy’ Karma for a price if they do not wish to earn it through interacting with the site. For instance, rating questions should generate karma.
   3. The site needs to be able to interact with:
      1. PayPal
      2. Credit cards
3. I need the option to modify:
   1. Whether to charge:
      1. Per question view
      2. For unlimited site access for a length of time (subscribers)
         1. For these users, there would not be a ‘currency’ system, as mentioned in 3b.
         2. All-subscriber-payments will be pooled
            1. 20% will be retained by the site as profit
            2. The other 80% (subscription-reimbursement) will be evenly distributed to the author of each premium question, based on the amount of days that month the question was designated ‘premium’ (premium-days)

At 23:59 GMT every day, every premium question will receive a +1 tally

At the end of each month the sum of ALL the questions tallies are added together:

The revenue generated by Question A = (subscription-reimbursement) x [(Question A’s premium-days) ÷ (ENTIRE BANK’s premium-days)]

Immediately after this payment is made, all premium-days tallies should be reset, and counting should commence for the next month.

I need the option to modify the fraction of All-subscriber-payments which will consist of subscription-reimbursement

* + 1. A combination of these two – users may decide to pay per question or for unlimited access
    2. For the freedom to selection questions for a mock-exam:
       1. Need to be able to toggle whether I force students to view questions, based on priorities I give to the question (see 3i) or give them the option to choose a subset which will form a mock examination, under certain conditions, eg:
          1. If they are interacting with premium, pay-per view questions
          2. If they pay for question-selection privileges (premium feature, user would have to pay for this feature)
       2. I need the option to toggle all of these conditions and permissions as required.
  1. The type of currency present on the site
     1. Karma
        1. As earlier discussed, high-school students will have free access to the site. They won’t have to pay for access or for question, but they also won’t be able to earn money from questions. Their currency will be Karma.
     2. Karma + (dollars/credits) [as mentioned in 2biii]
     3. Dollars
     4. Credits
  2. The conversion ratios of Credit purchases, eg:
     1. $4.95 buys 20 credits, $9.95 buys 50 credits, etc
  3. The conversion ratios of credits back-to bank account/PayPal
     1. Eg, 50 credits gets refunded back to $9.50
     2. All credits should be converted back into dollars at a flat rate.
  4. The fee generated per transaction for both types of monetised currency
  5. Whether to set a fixed price for questions, or allow a free-market approach
     1. If using the fixed price approach:
        1. I need the option to nominate how much ‘currency’ will be required to view a question
        2. I need the option to nominate how much ‘currency’ the site should be ‘*aiming’* to generate for the author
           1. I need to be able to set a time frame for when this total sum should be received by the author
  6. The criteria which must be met for a question to automatically become a premium question (i.e., 85th percentile, after 5 or more views)
  7. The criteria which must be met for a question to automatically become inaccessible to users, due to poor quality
     1. (i.e., 15th percentile, after 5 or more views)
     2. The post-exam correlation rating given to the question by exam leavers (did the question from the question bank actually turn up on the true exam?)
  8. The variables which are considered in the display prioritisation algorithm. In other words:
     + 1. The variables which are considered when deciding which order free questions should be displayed to students, eg:
          1. Percentile rank, based purely on user responses
          2. The *sum* of percentile rank multiplied by the number of ratings/views
          3. The post-exam correlation rating given to the question (did the question turn up on the student’s true exam?)

Higher weight should be given to the most recent correlation ratings. A question which occurred in an exam in 2010 should be more highly valued than a question which occurred in an exam in 2003.

* + - * 1. I need to be able to designate what weight each of these variables hold when the mock-exam question set is created
        2. The number of views a question has – helping to get at least 5 views to sort the question to the category of ‘trash’ or ‘premium’ as soon as possible
      1. The variables which are considered when deciding which order premium questions should be displayed to students, eg:
         1. Percentile rank, based purely on user responses
         2. The *sum* of percentile rank multiplied by the number of ratings/views
         3. The post-exam correlation rating given to the question
         4. The ‘value for money’ rating given to the question
         5. The number of days until the author’s payment deadline is reached
         6. I need to be able to designate what weight each of these variables hold when the mock-exam question set is created
      2. I need to be able to designate whether all users have a mix of free and premium questions in their mock exam set – regardless of whether they have credits or not
         1. If they don’t have credits, users will be prompted to deposit dollars/credits to their account each time a premium question is *due* to be displayed
         2. If the site is operating via a Karma system, the user will be prompted to *purchase* Karma for a cost.
         3. I may decide that users should have two options when selecting questions within their question set: free + premium questions, or solely premium question.
      3. For a mix of free + premium questions in a test-bank:
         1. Free and premium questions should be randomised and mixed together
         2. I need to be able to tweak what proportion should be free questions and what proportion should be premium questions

Examlet

Mobile app outline

Version 1

1. Required platforms:
   1. Android
   2. iPhone
2. Required functionality
   1. The ability to log in using previously the created user account details (username and password)
   2. A link to full website, where users can create a new account if necessary
   3. The ability to enter topics, or keywords, which were covered on their exam
      1. These then get submitted into the holding bay, ready for students to create questions based upon them
      2. These also independently guide the selection of question sets for mock exams
         1. If these keywords match the keywords associated with previously created questions in the bank, these questions will be displayed for users who are testing themselves in anticipation for this exam.
   4. The ability to enter similar multiple choice questions to the ones on a recent exam paper
      1. These questions will be submitted to a holding bay, where users can elect to research the correct answer, provide an explanation of the correct and incorrect responses, and nominate the keywords/topics which are associated with the question
         1. Once this is complete, the question will graduate to the question bank for that exam.
   5. The ability to review questions which have been viewed, specifically for the exam in question
      1. Users should be able to push a button to indicate that the question was very similar to that which occurred in the exam.
         1. This will need to be recorded against the question in the central database
      2. There should be an option for the administrator to toggle whether just the questions and options are displayed (globally), or whether the correct answer and the explanations should also be displayed
   6. The ability for students to test themselves on questions before their exam, from within the application.
      1. At this stage, nominating upcoming exams, recharging user’s accounts and all other advanced functionality should be conducted from the main website, not from the mobile app.
      2. They should be able to select which exam they want to test themselves on, and the questions should be automatically displayed on the device’s screen, one at a time, without giving the user the option to select the questions.
      3. If they are a subscriber:
         1. Both free and premium questions should be displayed, according to the randomised prioritisation algorithm described in the business plan (version 2 and above)
      4. If they are not a subscriber:
         1. Both free and premium questions should be displayed, according to the randomised prioritisation algorithm described in the business plan (version 2 and above)
         2. If they have a positive currency balance:
            1. Premium questions should be displayed, charging the account each time a premium question is displayed
         3. If there are insufficient funds in the account to display a premium question:
            1. A notice is displayed prompting the user to add funds to their account, or to subscribe, in order to view that particular question
            2. The user should just be able to click ‘next’ to move onto the next question
      5. The ‘testing’ interface should be very simplistic visually. It should focus only on the question, nothing else, with a little icon in the corner which allows the expansion to other options and features etc.
   7. The user should be able to enter the actual mark they achieved for a particular exam
      1. This will be gathered and associated to the username, for use at a later date if required.
3. All of the above features should be available within the full-site, as well. The mobile application should be a stripped down version of the original.

Examlet

Search Engine Optimisation features

Version 1

1. The site needs to be structured in a hierarchical fashion.
   1. First should be the site entry point (Home)
      1. Second should be the Country
         1. Third should be the state
            1. Forth should be the university name
2. I need a new page created for each separate part of the site
   1. I need the site to ability to automatically define 4 keywords for each separate page on the site
      1. These keywords impact the meta-tags and heading for each page
         1. The ‘first’ keyword could be
            1. “Submit”,
            2. “Test”, or
            3. “Stats”
         2. The ‘second’ would be dynamic
            1. This would involve the root of the url of the university they are affiliated with.
            2. Eg, if a user is from the [University of Calcutta](http://www.caluniv.ac.in/) (url: <http://www.caluniv.ac.in/>), this keyword would be “caluniv” (I have most of this stripped-back url data already)
         3. The ‘third’ could be the country code that their university resides in
            1. (I’m mining this data currently)
            2. In this case, the country is India, the country code top-level domain (ccTLD) is “in”
         4. The ‘forth’ could be the degree which is involved.
      2. The pages need to be constructed in a hierarchical manner
      3. The keywords need to be present in the url of each page:
         1. Eg: [www.in.examlet.com/caluniv/mbbs/submit.php](http://www.in.examlet.com/caluniv/mbbs/submit.php)

