ELIXIR TECHNICAL DOCUMENTATION

horizontal line

***CHANGES MADE BY ARPIT***

**MODULE: ADMIN**

Database changes-> models.py

* Edit the classes and objects to match the new database.
* db.sqllite change the old data and map to new database.

Form changes -> forms.py

* mostly model based forms
* changed the form classes according to new database.
* some fields that were directly a part of the model are now properties hence not excepted as direct fields in the model form.
* They were manually added. The choices were defined with querysets. The validation for these fields was also manual.

Functionality changes -> views.py

* The fields which were manually dealt with were saved with custom code as well.
* Similarly the errors that were thrown by these fields on validation had to taken through custom code and returned as json. Most other functionality changes were also dealt with here.

javascript > static/js

* some of the forms were not delivered in one go. The first part was delivered directly and the other ones were delivered on ajax calls. These were written in js files.

templates ->

* the view/edit/add question templates had to be made separately for each question type. these were made in template files.

***CHANGES MADE BY DHIVYA & DHANYA***

**MODULE: ADMIN**

**FUNCTIONALITY: QUESTIONS**

1. **Crossword Questions (SPOKEN ENGLISH)**

1. Created 6 HTML crossword pages with a common javascript and CSS file.

HTML Pages:

1. crossword25.html
2. crossword26.html
3. crossword27.html
4. crossword28.html
5. crossword29.html
6. crossword30.html

JS file:

1. commonjs.js

CSS file:

1. styles.css

**Logic**

* Hardcoded the question in the js file.
* The user can check whether the entered values are correct using the check button .
* There is a clue button which can be used to find the correct value for a particular cell/box.
* “Clear all”, will clear all the entered values.

**Functions**

* initializeScreen function is called on body load which initializes the puzzle.
* The puzzleArrayData gets data from the items in the preparePuzzleArray function .
* addHint function contains the hints for the puzzle.
* clearAllClicked function will clear the entered answers and makes the puzzle empty .
* checkClicked will check the answers entered by the user with the already hardcoded answers and will display red in the cell when the answer is wrong and green when it is right.
* clueClicked function will display the correct answer for the selected cell .

1. **Word Search Questions (SPOKEN ENGLISH)**

1. Created 6 HTML word search pages with a common javascript and CSS file.

HTML Pages:

1. wordfind25.html
2. wordfind26.html
3. wordfind27.html
4. wordfind28.html
5. wordfind29.html
6. wordfind30.html

JS file:

1. wordfind.js

CSS file:

1. style.css

**Logic**

1. In word search, the letters are displayed randomly and there are 6 levels of question.
2. The user should connect the letters by dragging the mouse pointer.
3. Solve button can be used to find the answer.
4. **Match the following Questions (SPOKEN ENGLISH)**

* ElixirWebApp > user\_student > templates > match\_the\_following >

1. Created 6 separate html pages for match the following and one JavaScript file.

HTML Page:

1. match\_level1
2. match\_level2
3. match\_level3
4. match\_level4
5. match\_level5
6. match\_level6

JS file:

1. logic.js

**Logic**

* Check if all the boxes are filled, if not filled display alert message and highlight the empty boxes with red border.
* Check with the answer (passed as an argument from the html pages) and display Correct Answer! (in green color) if all the answers are correct. And display Wrong Answer! (in red color), if it is wrong.

**MODULE: STUDENT**

**FUNCTIONALITY: UI (SPOKEN ENGLISH)**

1. **Help pages**

**ElixirWebApp > user\_student > templates > help > spoken\_english > words > 6 html pages**

* Created 6 separate html pages for Lesson (Words module).

HTML Page:

* lesson1
* lesson2
* lesson3
* lesson4
* lesson5
* lesson6

1. **UI CHANGE IN DISPLAYING THE PROGRAMS.**

* Added images and changed the font of all the “Programs” in **s\_home.html** file.

1. **UI CHANGE IN DISPLAY OF THE PROGRAM->MODULES->LEVELS (SPOKEN ENGLISH)**

* Displayed all the modules and levels of the Spoken English program in a single page (edited in **spoken\_english.html**)
* Make changes as per the matrix of questions
* Linked all the type of questions to Modules and Levels

1. **OTHER UI CHANGES**

* Add breadcrumbs

**MODULE: STUDENT**

**FUNCTIONALITY: ASSESSMENT(SPOKEN ENGLISH)**

1. **GENERAL ASSESSMENT.**

* In the SPOKEN ENGLISH program for a few modules added question for general assessment in the database and displayed them based on the module and level .

**LOGIC:**

* When general assessment is clicked it will go to **standard\_test** in views.py file .
* There the questions for the particular level is taken from the database and it then renders **standard\_test.html.**
* From standard\_test.html it then goes to **ajax\_standard\_test** in views.py .
* In ajax\_standard\_test it will check the question type and will display the particular html page based on question\_type.

1. **AUDIO/VIDEO/TEXT (SPOKEN ENGLISH)**

* In the SPOKEN ENGLISH program for a few modules added questions for audio, video, text in the database and displayed them based on the module and level .

**LOGIC:**

* Same logic like general assessment, the questions will be taken based on the level and the question\_type\_id.
* Here **av\_test** and **ajax\_av\_test** methods are used in views.py file

**MODULE: ADMIN**

**FUNCTIONALITY: QUESTIONS**

**PROGRAM - EDUCTION TO EMPLOYABILITY (E2E)**

1. Resume builder
2. Logical Reasoning
3. Aptitude test

Created e2e.html page

**Changes in views.py**

**In spoken\_english function:**

if program id(pk2) = 3

return e2e.html

else

return spoken\_english.html

**Resume builder**

* ElixirWebApp > user\_student > templates > resume\_builder > index.html
* **Levels -** resume builder
* index.html: Added breadcrumbs and question mark icon with a popover

**Logical Reasoning:**

**Levels:**

* level1
* level2
* level3
* level4
* level5
* level6

**Aptitude test:**

**Levels -**

* logical reasoning
* numeric reasoning
* verbal reasoning
* vocabulary test

**Question type considered for Logical reasoning and Aptitude test**

* Fill in the blanks
* Multiple Choice
* Single Image Based question