Computer Science IA – Ugo Janssen
Candidate Code – kqb412

Criterion A

Scenario - Rationale - Success Criteria

Scenario

My client is my friend’s little brother who has just started to take part in the IB Diploma Programme. He has just started the IB deciding to take on computer science, physics, and biology at higher level. After a few months in the IB, he realised that he isn’t struggling with the problem-solving concepts, however, consistently keeps missing marks on terminology or small mark questions about the tiny things that the teacher’s don’t go over.

My client also mentioned his concerns about his current studying methods: Feynman technique, blurting method, spaced repetition, and mind mapping were not working. But enjoyed active recalling. The information he would get online were either outdated or irrelevant to the IB. This allowed me to think of a good IT solution for my internal assessment.

In order to further analyse my client’s problem, numerous discussions with my client were needed. He described what he wanted: a bank where my client could access all the information needed, a flashcard system with the option to select specifically what topics and subtopics to revise as he liked an active recalling method, and finally a friendly interface and that all the data comes from the IB Oxford Books.

Rationale

A website serves as the most optimal solution as it offers a space where my client can bookmark and return to effortlessly. Also designed to be user-friendly, and not just a source of information but a practical tool for active revision. The software would have two sections, a bank of information and a flashcard system, both linked to together. The following three languages will be needed: HTML, CSS, and JavaScript for structuring, designing and interactive study tools. I decided to use these three:

<u>HTML</u>
Standard markup language for creating web pages.

CSS
Used to style website ensuring interface is user-friendly improving accessibility.
Can make the study experience more enjoyable and less monotonous.

JavaScript
Scripting language for creating interactive elements.
Flashcard system that allows my client to choose specific topics and subtopics with a checkbox.
Enables my client to reveal answers and advance cards which is an effective tool for reinforcing knowledge through active recall.

Success Criteria (SC)

- The program should present a comprehensive list of topics for each subject my client studies.
- The program should enable the client to select preferred topics and delve into specific subtopics through checkboxes.
- The program should display a list of information for the chosen selected topics/subtopics.
- The program has another section for a flashcard system for each subject.
- The program will have a flashcard with a ‘show answer’ feature to reveal information instantly.
- Navigation through flashcards with a ‘next’ control to move other definitions and clear previous answers.
- The client should have the ability to easily navigate through different sections of the program.

My interview conversation with my client will be stated in Appendix A

Word Count – 451

Criterion B

Record of Tasks – Test Plan – Design

Record of Tasks

Task Number	Planned Action	Planned Outcome	Time	Completion Date	Criterion
1	First meeting with client	Getting information about my client's requirements and what she wants	30m	14/02/2024	A
2	Outlining scenario	Write down about my client's issues which guides me to my project	30m	15/02/2024	A
3	Producing rationale	Explain why the function of my study program and the coding language I chose	45m	15/02/2024	A
4	Second meeting with client	Briefly explain my idea for the study program and get feedback	30m	18/02/2024	A
5	Listing success criteria and finishing criterion A	List the main points that my program must complete by the end of the project	1h	18/02/2024	A
6	Prototype sketching	Making a prototype of what the program should look like	45m	20/02/2024	B
7	Draw flowchart diagram to	Created a flowchart on Miro	1h	23/02/2024	B

	understand program working process				
8	Think about test plan	Finished detailed test	45m	23/02/2024	B
9	Start writing the code for the information (subtopics + topics)	Implement HTML structure on the information pages Writing JavaScript functions for dynamic content CSS to style the pages in order to match prototype design	4h	25/02/2024	C
10	Use test plan for the information page	Ensuring accuracy of the content for each subtopic using Testplan	20m	25/02/2024	B
11	Start developing flashcard code and test	Develop flashcard functionality in JS, implemented show answer feature	4h	28/02/2024	C
12	Test Plan the whole app	Comprehensive testing of entire application using detailed test plan	25m	28/02/2024	B
13	Finish Criterion C	Document the code and provide comments for clarity	2h15	01/03/2024	C
14	Finished third meeting	Discuss on how my client enjoyed the app plus feedback	30m	02/03/2024	
15	Ideas how to improve the product further	Gather feedback and identify areas for improvement	45m	02/03/2024	E

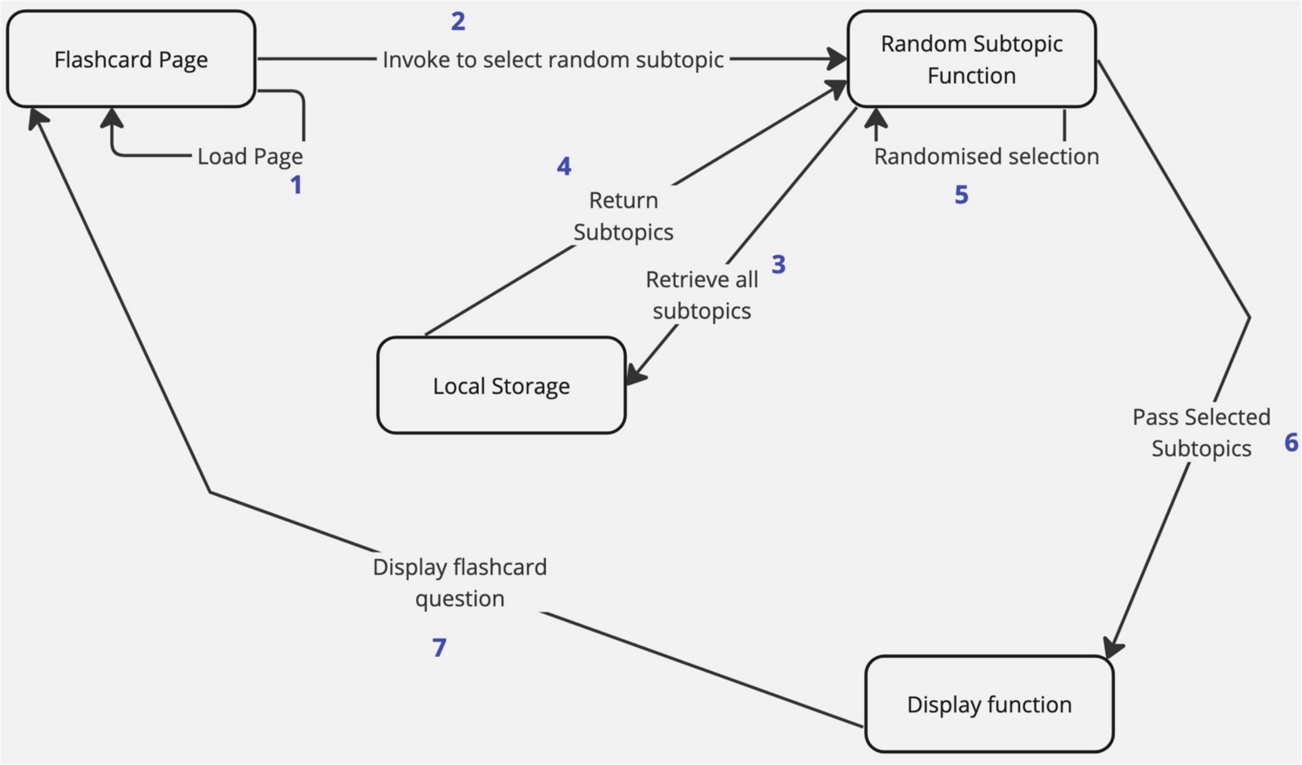
16	Record video to show programs functionality	Record the walkthrough of the program’s features	15m	03/03/2024	D

Test plan

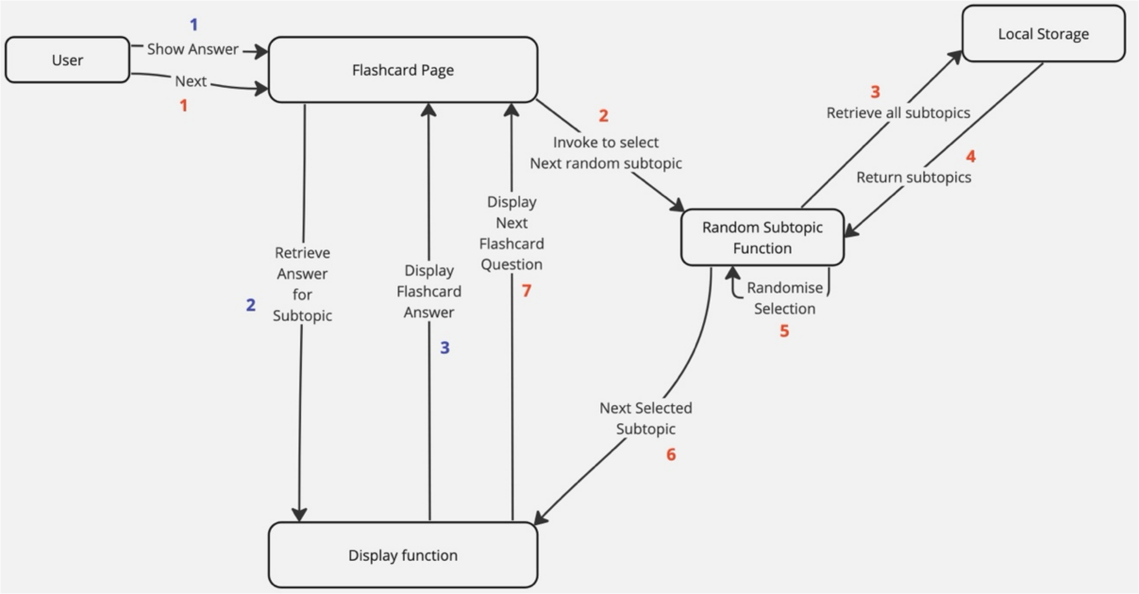
Success Criteria Number	Action for Testing	Method of Testing
1	Visually check the home page, topics page, subtopics page and flashcards page display correctly	Load the home page to ensure all the contents are present and properly formatted. Navigate to each page: topics, subtopics, and flashcards pages checking each for correct display
2	Check that topic selection links to the correct subtopics	Select a topic, observe if the corresponding subtopics are automatically appeared in the next page
3	Validate the subtopics functionality	Choose and submit the subtopics for a given subject and verify that the submission accurately reflects the selected subtopics
4	Assess the visibility of the flashcard section from the homepage	Click on the flashcard button on the home page and ensure the section appears
5	Testing the ‘show answer’ button/feature on flashcards	Use the show answer button on a flashcard to reveal the answer that should be displayed immediately
6	Evaluate the functionality of the ‘next’ button in the flashcard page	Press ‘next’ on a flashcard and confirm that the previous answer clears and a new definition display
7	Inspect all navigation buttons for proper operation	Use the ‘submit’, ‘go back’ and ‘next’ buttons to navigate, each should perform as expected

Design

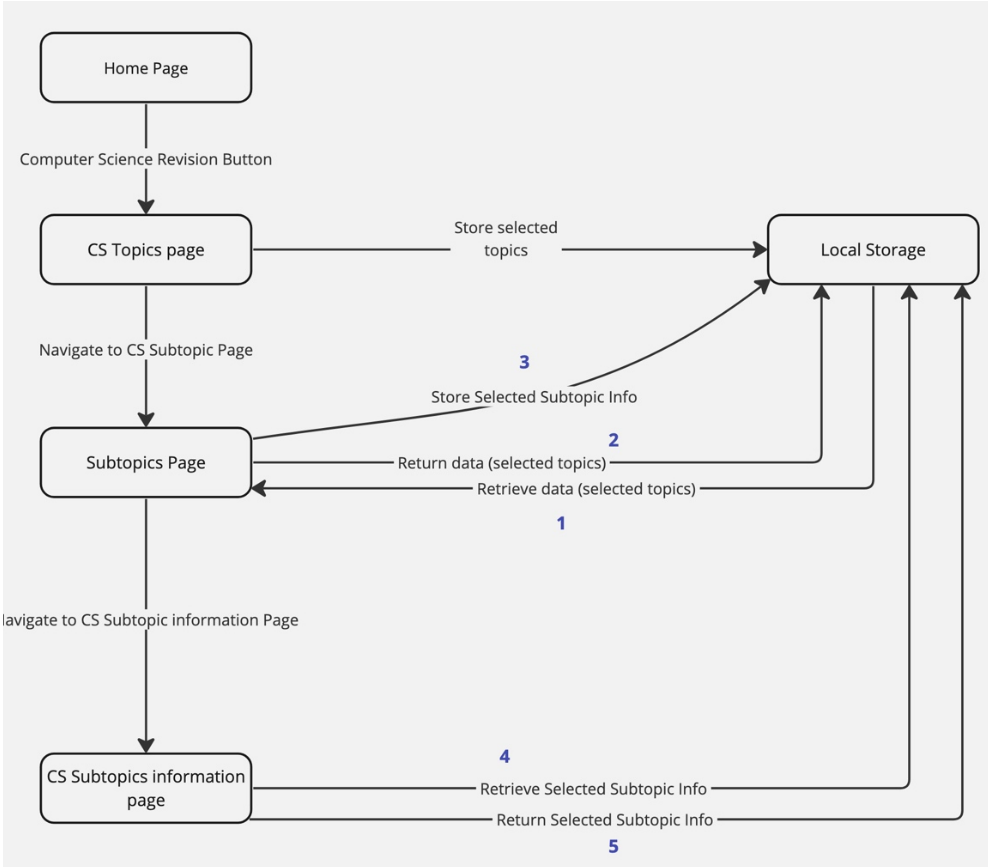
Page Load Flowchart



User Interaction Flowchart



Information pop-up flowchart


Home Page
Computer Science Revision Button
CS Topics page
Store selected topics
Local Storage
Navigate to CS Subtopic Page
3
Store Selected Subtopic Info
Subtopics Page
2
Return data (selected topics)
Retrieve data (selected topics)
1
lavigate to CS Subtopic information Page
4
CS Subtopics information page
Retrieve Selected Subtopic Info
Return Selected Subtopic Info
5

Note that for the figure above, it is the same process for all the 3 other subjects. For the sake of repetition, I decided to do it for computer science only.

Graphical user interfaces (GUI)

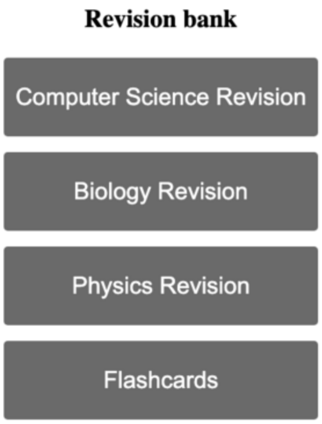
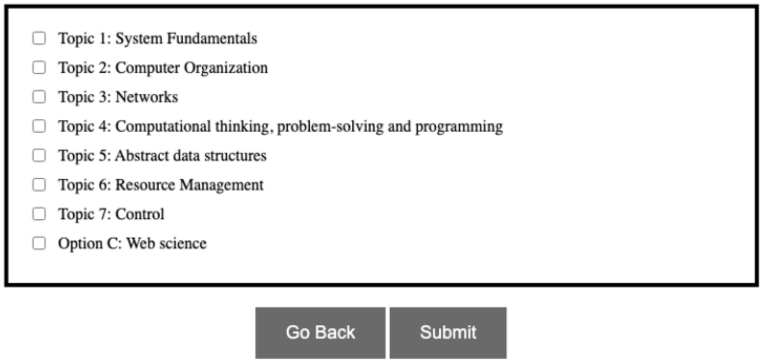
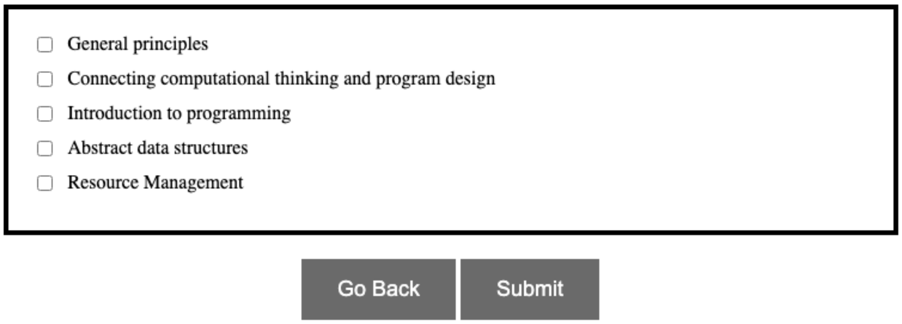


Figure 1: Home page



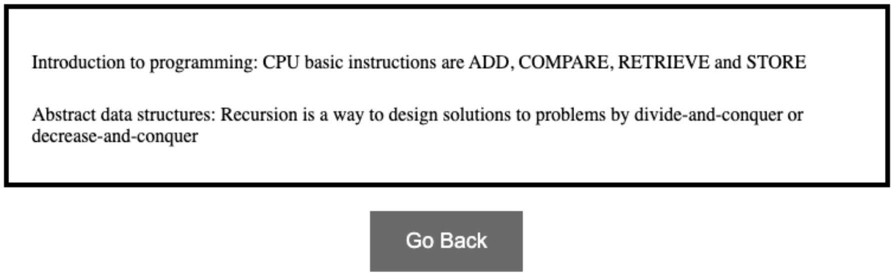
- □ Topic 1: System Fundamentals
- □ Topic 2: Computer Organization
- □ Topic 3: Networks
- □ Topic 4: Computational thinking, problem-solving and programming
- □ Topic 5: Abstract data structures
- □ Topic 6: Resource Management
- □ Topic 7: Control
- □ Option C: Web science

Figure 2: Syllabus outline



- □ General principles
- □ Connecting computational thinking and program design
- □ Introduction to programming
- □ Abstract data structures
- □ Resource Management

Figure 3: Subtopic List

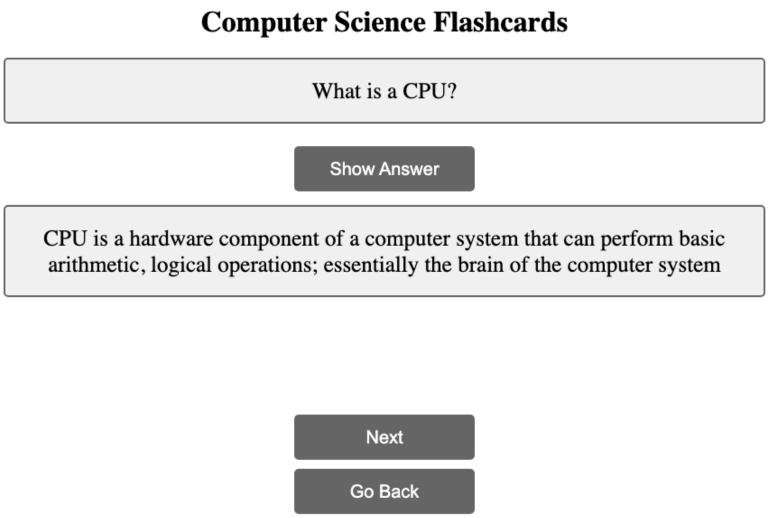


Introduction to programming: CPU basic instructions are ADD, COMPARE, RETRIEVE and STORE

Abstract data structures: Recursion is a way to design solutions to problems by divide-and-conquer or decrease-and-conquer

Figure 4: Information on selected subtopics

The structure of Figures 2,3, and 4 are all user-friendly due to maintaining a consistent and clean design minimizing distractions focusing on the user’s attention. The checkboxes allow for straightforward multi-selection and "Go Back" and "Submit" buttons provide a clear means to navigate on the website.



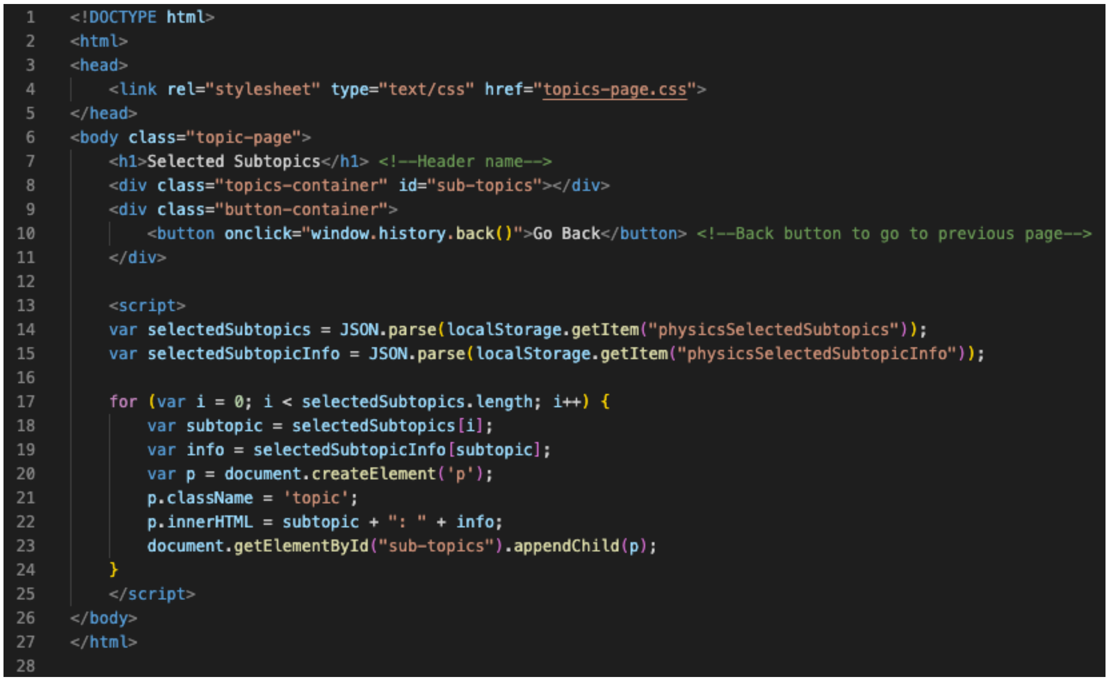
Computer Science Flashcards

What is a CPU?

CPU is a hardware component of a computer system that can perform basic arithmetic, logical operations; essentially the brain of the computer system

Figure 5: Flashcard system

Criterion C



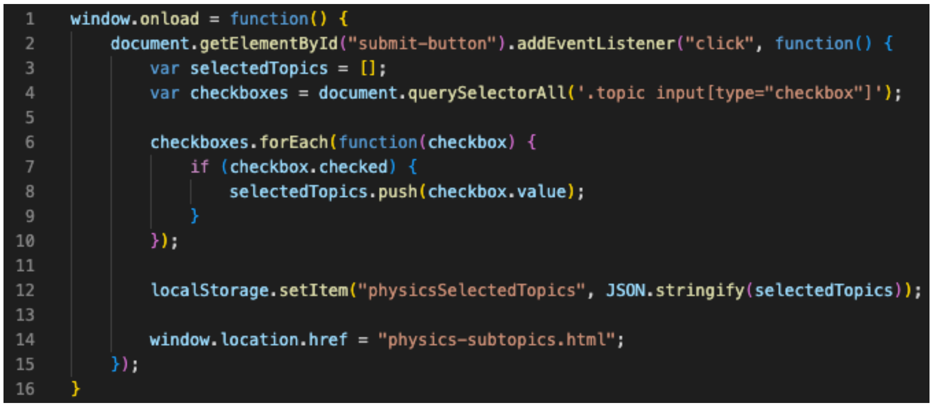
```html
<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" type="text/css" href="topics-page.css">
</head>
<body class="topic-page">
    <h1>Selected Subtopics</h1> <!--Header name-->
    <div class="topics-container" id="sub-topics"></div>
    <div class="button-container">
        <button onclick="window.history.back()">Go Back</button> <!--Back button to go to previous page-->
    </div>

    <script>
    var selectedSubtopics = JSON.parse(localStorage.getItem("physicsSelectedSubtopics"));
    var selectedSubtopicInfo = JSON.parse(localStorage.getItem("physicsSelectedSubtopicInfo"));

    for (var i = 0; i < selectedSubtopics.length; i++) {
        var subtopic = selectedSubtopics[i];
        var info = selectedSubtopicInfo[subtopic];
        var p = document.createElement('p');
        p.className = 'topic';
        p.innerHTML = subtopic + ": " + info;
        document.getElementById("sub-topics").appendChild(p);
    }
    </script>
</body>
</html>

```

Figure 6: physics-subtopics-existing-page.html

Name of file	Techniques used
physics-subtopics-existing-page.html	For Loop DOM manipulation Array Indexing

Figure 6 shows the lines of code being displayed in HTML and JavaScript. It starts by retrieving items (stored under **physicsSelectedSubtopics** and **physicsSelectedSubtopicInfo** in JSON format) from the local storage using the **localStorage.getItem() method**. A for loop is then used so that it can iterate through the **selectedSubtopics** array containing the names of the selected subtopics. While the for loop is iterating through the subtopic array (variable) it is assigning its corresponding subtopic information through its array index. The info variable is assigned the value of the subtopic key in the **selectedSubtopicInfo** object. After all this a new paragraph (**<p>**) element is created using the **document.createElement() method**. The property **(p.innerHTML)** of the paragraph element is set to a string that combines the subtopic and info values so that it can be displayed as the content of the paragraph element. This string displays the content of the paragraph element. The element is appended as a child to an element with an id (sub-topics) using the **appendChild() method**. Adding another paragraph element to the website to make it visible.

In summary, retrieves data from local storage. Iterates over the selected subtopics, creates a paragraph for each subtopic so that it can append the element to the website. Allowing my client to see the information on the page based on subtopics chosen.



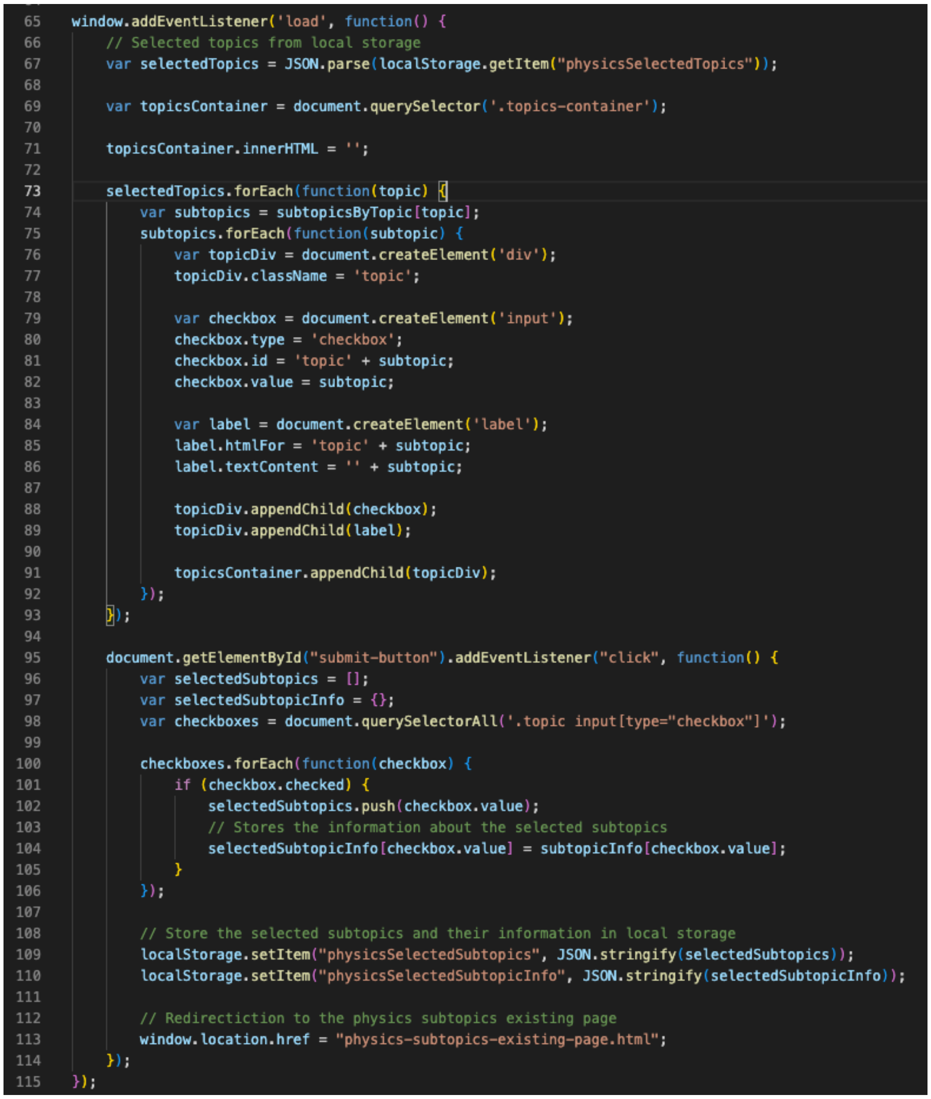
```javascript
window.onload = function() {
    document.getElementById("submit-button").addEventListener("click", function() {
        var selectedTopics = [];
        var checkboxes = document.querySelectorAll('.topic input[type="checkbox"]');

        checkboxes.forEach(function(checkbox) {
            if (checkbox.checked) {
                selectedTopics.push(checkbox.value);
            }
        });

        localStorage.setItem("physicsSelectedTopics", JSON.stringify(selectedTopics));

        window.location.href = "physics-subtopics.html";
    });
}
```

Figure 7: physics-topics.js

Name of file	Techniques used
physics-topics.js	Event handling DOM manipulation Array manipulation Control Flow Technique (Conditional Testing) - If statements

Figure 7 shows lines of code written in JavaScript. The first line indicates that the function will be executed when the window loads (**window.onload**). The button (**id: submit-button**) has an event listener so that it performs the function inside the event listener when the button is clicked. An empty array (**selectedTopics**) is initialised to store the selected topics. The **querySelectorAll method** selects all the checkboxes with the "**topic**" class. Later on, stores them in the **checkboxes** variable. It is then later on iterated over each checkbox using the **forEach** method. For each checkbox, it will check if it is selected (i.e box will become blue) using the **checked** property and a **Control Flow Technique** (Conditional Testing). If the box is blue, then it will retrieve the value using the **value** property and append it to the **selectedTopics** array using the **push method**. After iterating all checkboxes, it stores the **selectedTopics** array in the browser's local storage. Then the array is converted into a string using the **JSON.stringify function**. And the **window.location.href** function directs my client to the outline of the subtopics page.

In summary, the code listens for a button click event to retrieve selected topics, stores them in the local storage and redirects my client to the outline of the subtopics page.



```javascript
window.addEventListener('load', function() {
    // Selected topics from local storage
    var selectedTopics = JSON.parse(localStorage.getItem("physicsSelectedTopics"));

    var topicsContainer = document.querySelector('.topics-container');

    topicsContainer.innerHTML = '';

    selectedTopics.forEach(function(topic) {
        var subtopics = subtopicsByTopic[topic];
        subtopics.forEach(function(subtopic) {
            var topicDiv = document.createElement('div');
            topicDiv.className = 'topic';

            var checkbox = document.createElement('input');
            checkbox.type = 'checkbox';
            checkbox.id = 'topic' + subtopic;
            checkbox.value = subtopic;

            var label = document.createElement('label');
            label.htmlFor = 'topic' + subtopic;
            label.textContent = '' + subtopic;

            topicDiv.appendChild(checkbox);
            topicDiv.appendChild(label);

            topicsContainer.appendChild(topicDiv);
        });
    });

    document.getElementById("submit-button").addEventListener("click", function() {
        var selectedSubtopics = [];
        var selectedSubtopicInfo = {};
        var checkboxes = document.querySelectorAll('.topic input[type="checkbox"]');

        checkboxes.forEach(function(checkbox) {
            if (checkbox.checked) {
                selectedSubtopics.push(checkbox.value);
                // Stores the information about the selected subtopics
                selectedSubtopicInfo[checkbox.value] = subtopicInfo[checkbox.value];
            }
        });

        // Store the selected subtopics and their information in local storage
        localStorage.setItem("physicsSelectedSubtopics", JSON.stringify(selectedSubtopics));
        localStorage.setItem("physicsSelectedSubtopicInfo", JSON.stringify(selectedSubtopicInfo));

        // Redirectiction to the physics subtopics existing page
        window.location.href = "physics-subtopics-existing-page.html";
    });
});
```

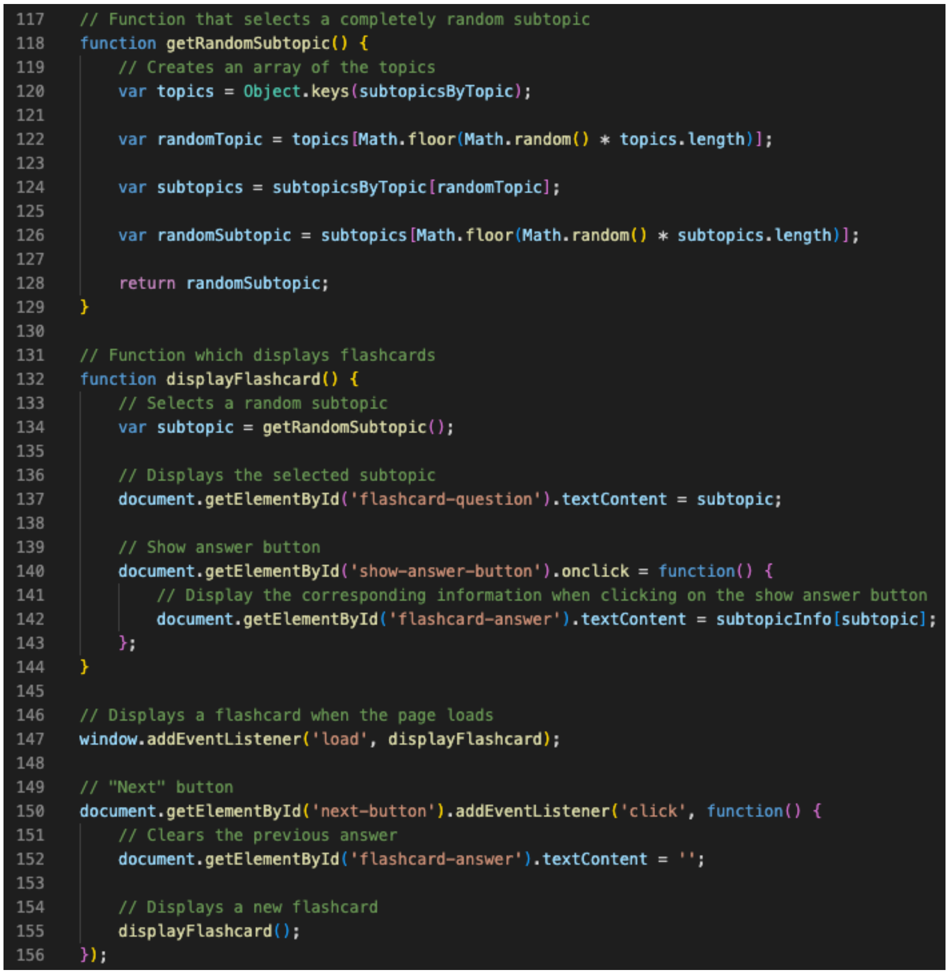
Figure 8: physics-subtopics.js

Name of file	**Techniques used**
physics-subtopics.js	Object-oriented programming Event handling Randomization Iteration (array) Nested loops Conditional statements JSON parsing and stringification

Inside the load event of the window object, an event listener is added meaning that the code will be executed once the page has been loaded. The **localStorage.getItem()** method gets the selected topics from the local storage which are converted as a JSON string. Therefore, it is converted back to an array of topics using the **JSON.parse() method**. The name '**topics-container**' is the class name which is assigned to the variable **topicsContainer**. After, the **innerHTML** of the variable is cleared so that it removes any content inside the var. The **forEach()** method then iterates over each of the topics selected (blue box). And for each one, it gives back the corresponding subtopics through the **subtopicsByTopic** object. The **forEach()** has a nested loop that creates a new div (topic) which has a checkbox and a label for each subtopic. After which **createElement()** method is created (for the checkbox) and it is assigned: type, id, and value. A **htmlFor** is associated with it for the corresponding checkbox. In the **topicDiv** element, the checkbox and label elements are as child nodes. This element is then added as a childnode to the **topicsContainer** element.

An event listener is also added to the '**submit-button**' element. Line 95 initialises an empty array **selectedSubtopics** and an empty object **selectedSubtopicInfo** in the event listener. The **querySelectorAll()** selects all the checkboxes in the topics container. The code then keeps on iterating over each of the checkboxes and if one of them is checked, the value which corresponds to a subtopic is then added to the array. The information about the subtopic is stored in the object using the value as a key. The information is stored in the local storage and converted to JSON strings. And finally redirects my client to the page.

In summary, the code generates checkboxes for each of the selected subtopics so that my client can select multiple ones. Stores the subtopics and information in a local storage and redirects my client to another page where the information is displayed.



```
// Function that selects a completely random subtopic
function getRandomSubtopic() {
    // Creates an array of the topics
    var topics = Object.keys(subtopicsByTopic);

    var randomTopic = topics[Math.floor(Math.random() * topics.length)];

    var subtopics = subtopicsByTopic[randomTopic];

    var randomSubtopic = subtopics[Math.floor(Math.random() * subtopics.length)];

    return randomSubtopic;
}

// Function which displays flashcards
function displayFlashcard() {
    // Selects a random subtopic
    var subtopic = getRandomSubtopic();

    // Displays the selected subtopic
    document.getElementById('flashcard-question').textContent = subtopic;

    // Show answer button
    document.getElementById('show-answer-button').onclick = function() {
        // Display the corresponding information when clicking on the show answer button
        document.getElementById('flashcard-answer').textContent = subtopicInfo[subtopic];
    };
}

// Displays a flashcard when the page loads
window.addEventListener('load', displayFlashcard);

// "Next" button
document.getElementById('next-button').addEventListener('click', function() {
    // Clears the previous answer
    document.getElementById('flashcard-answer').textContent = '';

    // Displays a new flashcard
    displayFlashcard();
});
```

Figure 9: physics-subtopics.js

Name of file	**Techniques used**
physics-subtopics.js	Random number generation DOM manipulation Event handling JavaScript object manipulation

The line of the code starts off with a function (**getRandomSubtopic**) which chooses a random subtopic from the set of topics/subtopics. An array (**topics**) inside this function is created using the method (**Object.keys**). This array contains all the information available in the object (**subtopicsByTopic**). A random topic from the array is selected using the functions, **Math.random()** and **Math.floor()**. It then gets back the subtopics linked with the randomTopic and stores them in the subtopics variable. And the function returns the randomly selected subtopic.

A function responsible for displaying a ‘flashcard’ with a completely random subtopic and the corresponding information is defined (**displayFlashcard**). Then the selected subtopic is shown (**id: flashcard-question**) using the **textContent** property. An HTML element is attached to an event listener (**id: show-answer-button**) meaning when the button is clicked, a function will trigger so that the corresponding information is shown for the selected subtopic. Another event listener is attached for the ‘Next’ button (**id: next-button**) so that when this is clicked, the previous answer will clear by setting the text (id: **flashcard-answer**) to an empty string. So then, the **displayFlashcard()** function can show a completely new flashcard with a different subtopic.

Overall, this code randomly selects a subtopic, displays it as a flashcard, and allows my client to reveal the corresponding answer. The 'Next' button can be clicked to clear the previous answer and display a new flashcard.

Word Count – 997

Criterion D

Click here to go see my video:
https://drive.google.com/drive/folders/1dghxtBv1RnVGss7KCdEcbiLmyv2c0Qac?usp=sharing

Criterion E

Success criteria evaluation

Success Criteria Number	**Evaluation**
SC1	The program successfully implemented a comprehensive list of topics for each subject ensuring a thorough revision resource
SC2 & SC3	The program’s interface allows the client to select and de-select preferred topics and subtopics using checkboxes.

SC4	The program includes a flashcard system for each subject facilitating client's preferred study method (active recalling)
SC5	The flashcard system has 'show answer' feature that functions correctly and pops up the informational immediately upon the client's request
SC6	Navigation within the flashcard section is easy with a next control for the client to advance through cards and clear previous answers
SC7	The client can easily navigate through all the sections of the program indicating user-friendly interface

Feedback from Client

After a few days of use, the client's feedback was overwhelmingly positive. Firstly, the user-friendly interface facilitated their navigation through the system making the learnability process smoother. Additionally, the flashcard system was the most effective in terms of remembering knowledge. The detailed and coverage of topics along with the ability to select specific subtopics was also praised.

Recommendation for Future Development

Flashcard creation and editing feature

An additional feature to allow my user to create and edit flashcards. This feature would allow my client to use the flashcard system the same way as note-taking. Whenever a teacher mentions something important that my client wants to remember, a flashcard can instantly be created. Not only aiding in retaining information but also reviewing it later.

Progress tracking with Gamification

Incorporating a tracking system to monitor my client's performance on the flashcards. This system could potentially track number of correct answers, incorrect answers, progress over time on various topics. Providing feedback at the end of each flashcard session can help my client to identify the strengths and weaknesses. Additionally, gamification elements such as earning points, badges, or even moving up levels can motivate my client to engage and study more.

Mobile App for flexible learning

Developing a mobile app would help my client to improve flexibility to study anytime and anywhere. Offering an experience to access information, flashcards, and track progress on the go. Also incorporating notifications for study reminders and daily challenges to keep my client engaged.

Word Count – 338

Bibliography

Allott, A., & Mindorff, D. (2014). *Biology : Oxford IB diploma programme. Course Companion* (2014th ed.). Oxford University Press. (Original work published 2014)

Artist, C. (2022, August 3). *Flashcard App With Javascript | Step By Step Javascript Project*. Www.youtube.com. https://www.youtube.com/watch?v=mizkwuCqFsc

Dimitriou, K., & Markos Hatzitaskos. (2018). *Core computer science : for the IB diploma program (International Baccalaureate)*. Express Publishing, , Cop.

Poulson, J. (2020, December 13). *Learn HTML, CSS and JavaScript by building a Flash Card App (Beginner Tutorial)*. Www.youtube.com. https://www.youtube.com/watch?v=urf3kHQX7aI

Roby, P., & Oxford. (2015). *IB physics. Higher level*. Osc Publishing.

Appendix A

Interview #1 – Addressing struggles

Me: Good morning. Thank you very much for finding time to have this interview with me. We've spoken before about how you are doing with the IB program, and you mentioned that you are doing fine with your standard levels but struggling with your higher-level subjects. Would you care to explain a little bit more about what exactly you are struggling with?

Client: Hi. I've been doing the IB for quite a while now and in most of my classes I'm doing quite well, and I am pretty pleased with myself. However, I've seemed to lack in my higher-level subjects as for some reason, besides the textbook there aren't enough resources online to help me and most of the information I find, is either outdated or wrong.

Me: I understand, so what you need essentially is a study software which you could use to help you with your studying. What do you mainly struggle with, is it problem-solving, terminology or both?

Client: That's exactly what I need. I wouldn't really need help with problem solving, it is mostly just the terminology that I keep forgetting. I keep missing these marks on my exams that can level my grade up by one. I want a website that would allow me to have a place where I can access all my information.

Me: Understood. So how about a website that would have all your higher-level subjects and all the subtopics within those subjects? So, it would give you the option to select which ones you are struggling in and then a list of information based on the ones you select pop up. I understand that just reading information on a page doesn't help you remember them. Do you have any studying techniques that help you best remember topics?

Client: Yes, over the past few months I've tired many. The Feynman technique, blurting and spaced repetition. But flashcards were for sure the best way for me to remember them.

Me: Wonderful, so let's say I added an option so that when you select those subtopics. You could generate flashcards based on the ones that you've selected.

Client: That sounds good.

Me: Okay good. May I ask why not an app? Why a website?

Client: A website is preferable as I'm juggling responsibilities between the IB and life outside school, so it would be a lot better to have something that is only functioned properly on my laptop and nowhere else. I also can't stress enough how this program should be user-friendly.

Me: Of course, I'll make sure I put time into the design, so it is easy to use. We'll have another meeting shortly when I can propose my rationale and success criteria for you. Thank you for your time.

Interview #2 - Proposing solution

Me: Hello again, I've made some progress on the Revisionbank project, and I'd like to discuss some important aspects with you, mainly the success criteria and the programming languages I'll be using.

Client: Hi, that sounds great.

Me: First, let's talk about the success criteria. I've outlined several key points to ensure the website meets your needs. These include easy access and navigation, the ability to select specific subtopics for study, a flashcard generation feature. What are your thoughts on these criteria?

Client: That covers everything I'm looking for.

Me: Great to hear. Now, about the programming languages – I plan to use HTML, CSS, and JavaScript. HTML will be the backbone of the website, ensuring content is well-structured and easy to navigate. CSS will help make the interface visually appealing and engaging, while JavaScript will be used to add interactive features like the flashcard system. This combination should make the website both functional and user-friendly. Does that align with your expectations?

Client: Yes, that sounds appropriate. I'm not very familiar with these languages, but I trust your judgement. The focus on user-friendliness and functionality is crucial for me.

Me: Excellent. I'll make sure to keep these priorities in mind as I develop the website. Is there anything else you'd like to add or any other feature you think might be helpful?

Client: Yes, I was wondering when you are making the flashcards, it accesses the same information of the revision bank and the flashcards so that I am used to seeing the same bits of information. Thank you.

Me: You're welcome. I'll keep you updated on the progress, and we can schedule another meeting to review the initial version once it's ready.

Interview #3 - Feedback from client

Me: Good afternoon, I’ve been eager to find out what you think about the new program. How has your experience been so far?

Client: Hello! Well after using the system for a few days, I must say that I am extremely happy with the product. It is incredibly user-friendly and navigating through this system has been a breeze.

Me: That’s very nice to hear. How about the flashcard system, has it been beneficial?

Client: Absolutely the most effective tool for me so far. I can already see improvement in my knowledge retention and without much studying, I’ve been able to get those extra few marks I mentioned in one of our interviews.

Me: Very nice! And how did you find the content?

Client: I’m really impressed with the content coverage, as well as the ability to select different subtopics has been a game-change allowing me to focus on my weak points.

Me: I’m pleased to find out that the program is living to your expectations. However, for further improvements, I have been thinking about the ability to create and edit your own flashcards What would you like to see in a future development?

Client: I’ll be honest. My expectations have been exceeded. However, that does sound very useful, especially when instead of taking notes I could turn them into flashcards. But maybe some elements to make it more engaging?

Me: I’ve been thinking about some progress tracking with some gamification elements to make it more engaging. I mean who doesn’t love to earn points and badges? It would add a nice aspect to studying.

Client: Yes, I was just about to say that. That would be perfect. I also think sometimes it gets annoying knowing I can only use the application on my laptop. I think an application where I can use it on multiple devices such as my phone would be really nice. Especially if they had study reminders etc.

Me: Excellent! Thank you very much for this meeting. The feedback was very useful.

Client: No worries and thank you for all the hard work.

Appendix B

Lines of code will be represented here:

bio-flashcards.html
bio-subtopics-existing-page.html
bio-subtopics.html
bio-subtopics.js
bio-topics.html
bio-topics.js
cs-flashcards.html
cs-subtopics.existing.page.html
cs-subtopics.html
cs-subtopics.js
cs-topics.html
cs-topics.js
flashcards-style.css
home-flashcards.html
home-style.css
home.html
home.js
physics-flashcards.html
physics-subtopics-existing-page.html
physics-subtopics.html
physics-subtopics.js
physics-topics.html
physics-topics.js
topics-page.css