



Semester 5
17 june 2024

ADVICE REPORT

**Product Suggestion Tool
for Low Pressure Studio B.V.**

PRESENTED BY:

Adam Wiszniowski - Świder

1. Abstract

This report evaluates a product recommendation tool designed for Bataleon, a snowboard company. The tool aims to suggest suitable snowboards based on user input to increase customer satisfaction. While the core functionalities for User Interface (UI) and code are established, critical issues were identified. The lack of a mobile version results in inaccessibility of the product suggestion tool to 70% of users. The current results section placeholder needs a major redesign to align with Bataleon's style guide and meet user expectations. Most importantly, the code responsible for matching user input with snowboards appears to be malfunctioning, frequently resorting to the "closest match" feature even when exact matches do exist. This might be due to data format mismatch (flexibility, terrain) or missing values (boot/weight range) in the Bataleon database. Recommendations include prioritizing mobile development, refining the results section UI, and investigating code problems within the findProduct() function to ensure accurate matching. By addressing these issues, Bataleon can significantly improve the product recommendation tool, leading to a more positive user experience and ultimately increased customer satisfaction.

TABLE OF CONTENTS

1. Abstract	2
2. Introduction	2
3. Analysis of current situation.....	3
4. Recommendation	6
5. Conclusion	7
6. Appendixes	8

2. Introduction

E-commerce product suggestion/recommendation tool is a pretty common way for the companies that are operating an online store to improve their customers user experience, as well as satisfaction with a product. This solution allows users to provide input to the system, thanks to which they'll be provided with products that will suit their needs best. Bataleon is a company that is making snowboards and snowboard related equipment and they are in need of such a tool. The primary goal of making it is to improve user satisfaction with a product, and decrease return rate, by providing users with products that are most suitable for their needs. This document is meant to provide my recommendations regarding further development of product suggestion tool for Bataleon, based on gathered research, details of which you can find in the research document.

3. Analysis of current situation

User interface design

Currently the whole solution consists of several parts. The first part is the UI, which you can find in its entirety at appendix A. In its current state, the UI consists of several elements. At the top of the product suggestion tool, there is a header. It consists of 2 practical elements, navigational buttons, and progress bar. Progress bars' task is to inform users on how far into the questionnaire they are, and indicate how soon they'll see the results. Navigational buttons task is to allow users to seamlessly change from one question to another, should they want to make change to previously answered questions. Every part of the header from border-radius value and font to background colors and gradient color of progress bar, was styled in accordance with Bataleon style guidance, in order to make sure that product suggestion tool fits bataleon website as well as possible. Unfortunately I'm not allowed to share the style guide itself due to the company policy, but the Bataleon website is a good reference on its own. A link to it you can find [here](#), or at the **Appendix B1**.

Main element of the finder that is responsible for gathering users' answers are card buttons. They consist of the name of the answer at the top left, image relevant to the answer and a description in the middle which is describing what said answer is representing. Additionally at the bottom of each card button, you can see an additional green button. Despite a whole card being a button on its own, I've added the additional green "pill" button in order to highlight that it's clickable, as well as to better fit into the Bataleon website. I wanted to make the buttons and layout to feel more similar to other elements on the website, you can see the Bataleon website available at **Appendix B1** for reference. You can also visit **the prototype report** to see how the prototype was iterated.

User interface Issues

Currently there are some minor problems regarding the UI of the product suggestion tool. In the recordings of the testing available in the Usability test report , and also available at **Appendix C** you can see that when users are scrolling down to the results, the red window that is displaying contains the results. It was meant as a temporary placeholder used to verify the functionality. The prototype for the results section is available at **Appendix D**. Its design however was not approved by the company stakeholders, and is only my proposal for the design. Unfortunately due to time constraints I wasn't able to implement it in the coded version.

Another issue regarding the UI is the mobile version. Similarly to the results section I do have the prototype for it, which you can find at Appendix E, however due to time constraints I wasn't able to implement it in the coded version. Fortunately the functionality of the finder will remain the same in the mobile version, but what needs improvement is the styling of it. Its design was approved by the stakeholders, however it was based on version 4 of the prototype of the desktop version, and might benefit from an update.

Code design

Codewise, this solution consists of several elements in a few different coding languages, which sum up to over 1000 lines of code. The languages used are JavaScript, which is used for the overall functionality of a finder, such as switching between questions, displaying appropriate navigational buttons, and providing users with a snowboard sufficient for their needs, HTML and CSS which are used for making the website itself and styling it, and .liquid which is a coding language used by Shopify which is used to call snowboards from the website. Firstly, the visual site is operated on HTML and CSS. Going from the top, navigational buttons are assigned JavaScript functions that are responsible for displaying questions that each of the buttons is representing.

At the beginning, however, only the first button that is responsible for “Advancement level” is displayed, since it's the first question that is being asked. This practice is in place to display only the navigational buttons that are meant for a particular target group, because navigational buttons represent questions that are being asked, and displaying questions that are not asked to the user would be unnecessarily confusing. Additionally, the button that is showing the question that is currently displayed is always marked by having a much darker shade of gray added as a background color in order to indicate which question is currently being asked.

For answering questions, the user needs to click on card buttons which are responsible for storing answers and changing which question is displayed. The values that correspond to each of the answers are stored in data-value. The first question, however, has an additional function to it. Because we need to incorporate different questions for different users, an additional value is needed. After selecting the advancement level, one of 3 functions is activated. Those functions have a “chosen_path” value in them, and depending on the given answer, this value will be either 0 (beginner), 1 (intermediate), or 2 (advanced). Based on this value, other functions will change which navigational buttons are displayed. The question about shape is only asked to advanced users, therefore it doesn't need any if statements. The snippet of code you can find at **Appendix F**.

In comparison, the last 2 questions about terrain and about sizing are displayed to all users, but beginner or intermediate users shouldn't see the options for the shape of the board, the snippet with the code you can find at **Appendix G**. It includes 2 “if” statements, which are there to only display what needs to be displayed. All of the questions are using the same template, just with different values for the questions, the template you can find at **Appendix H**. This particular one is from a question about terrain.

How it works

After gathering values from html, the Javascript part of the code does a few things (besides what I have already mentioned). Firstly it calls snowboards from shopify database using liquid language, and makes an array with all the necessary values. Thanks to this solution, product suggestion tool will remain relevant even when the snowboards in stock change. This part you can find at Appendix I. It assigns values using `{{product.metafield.global.X}}` to corresponding values of the snowboards array. Later function `findProduct()` is executed, which is the most important function in the program, which you can find at Appendix J.

Firstly it collects data-values from every picked button, and stores them in constant values, which is a great way of collecting user input. Later it compares those values with the values of each of the boards, and limits the number of matches to best 3. Limiting the number of answers is further described in Competitive Analysis and Best good and bad practices report. In a nutshell this process is there to ensure the user receives meaningful help, that will allow them to pick a product that best suits their needs. If there were no direct matches it will analyze which values were fulfilled by the available boards, and will give an increased score value for each of the parameters that board is fulfilling. Lastly it will take the boards that were best fitting and create a div for them at the bottom of the page.

Code issues

Currently there are several issues regarding the functionality of the finder. During the testing, for which the whole documentation is available in Usability testing report document, it was observed that every time out of the combine 8 the system didn't provide the user with the direct match. Instead it provided them with the closest match option. While the purpose of the "closest match" feature is to provide users with match even when no direct match is found, it is very concerning that the system is unable to find any direct match despite providing many different inputs. Furthermore it was observed that despite multiple attempts the only closest matches that were assigned to the user were one of 3 boards, namely Party Wave - 164, Bataleon 20Y Board - 156 and board finder product 2-156. After closer inspection only 2 similarities were found between those 3 boards. First being that the level of advancement is being marked as "Intermediate" and "Expert" on all of them. Second is that on all of them there seems to either be an issue with calling numbers for boot size and weight or that those parameters are not available in the Bataleon database that is operating in shopify. You can find the exact specifications for those boards that are being called by the system in the **Appendix K**, the last of the arrays located in the appendix is one of the other boards that is not being provided as a closest match for comparison.

4. Recommendation

User interface Recommendations

In my opinion the most important feature of the UI that the product suggestion tool is currently lacking is the mobile version. In today's world the majority of the internet traffic is performed by mobile devices, 60.67 % to be exact, according to WhatsTheBigData Website[1](GilPress, 2024). Furthermore, data from Shopify regarding the EU Bataleon website traffic, available in **Appendix L**, shows that 70% of all traffic on the EU Bataleon website comes from mobile users. This means that without a mobile version, around 70% of users won't have access to the product suggestion tool, greatly decreasing its usefulness. For this reason, I recommend rapid implementation of the mobile version of the product suggestion tool to ensure all users have access to it.

Another problem with the UI is the lack of appropriate UI for results. Current solution was only meant as a placeholder, and requires a significant update. Technically it does provide information about the product, however it's insufficient in fulfilling bataleon style guides and is not providing the users with experience that is expected of an online store in the 2020s. The prototype that is available at **Appendix D** is my proposal of how the results section could look like. Its inspiration was the product page on Bataleon website, an example for which you can find at **Appendix B1**. This prototype however wasn't approved by the stakeholders, and Ronald, who is my company mentor and a stakeholder, expressed that it is possible that the design for the results section is going to be designed by one of the designers from Bataleon. Therefore my recommendation would be to either implement a coded version based on my prototype for the results section, or to design and implement a whole new solution.

Code Recommendations

Currently there are some problems regarding the code part of the product suggestion tool. The biggest one being that the system is exclusively providing users with "closest match". While it is a good thing that the "closest match" feature is working, the fact that it's the only way in which the user is receiving a match is concerning.. Most likely the piece that is creating problems is located somewhere in the findProduct() function. Within that function there is const filteredSnowboards which is looking for best matches, and if none are found, the closestMatch variable is created. The issue seems to be located somewhere within those elements, most likely in filteredSnowboards, because if there are no exact matches detected by this part, the closest match feature is activated. The problem might come from the fact that the values that are provided by the user to the system are not exactly the same as the ones that are being called from the Bataleon database. This problem applies to flexibility, which is an int in the code but a float in the database (which, according to my company mentor Cleay, who is a programmer, shouldn't be an issue); to all of the terrain options, because the database provides ratings for each terrain instead of directly declaring which terrain the board is made for; and to boot sizing range and weight range, which seem to not be called properly, instance of which you can see at "Whatever-138" board in Appendix K. Those were the possible sources of the problem that I could find, but it might be located somewhere else.

Possible solution would be to reevaluate the way in which the system is comparing the boards to user input, and adding score to boards. Also the system seems to need a code snippet that would handle the issue regarding terrain. Unfortunately I wasn't able to fix those issues myself due to time constraints. My advice for the company would be to analyze mentioned elements of the code, and test described solutions in order to fix them

5. Conclusion

In conclusion, this report documented the development of a product recommendation tool for Bataleon, a snowboard company. The goal is to enhance user satisfaction by suggesting suitable snowboards based on their input. The report covered the current functionalities of the tool, encompassing both the user interface and the underlying code structure. It also identified critical issues in these areas.

Firstly, the lack of a mobile version. This problem is significant considering that 70% of users (see **Appendix L**) visiting the website would not have access to the product suggestion tool. This issue can be solved by either implementing the proposed prototype (see **Appendix E**) or collaborating with Bataleon designers for a new solution and implementing it.

Second problem with the UI is the unstyled results section. The prototype for its design does exist (see Appendix D), but it wasn't approved by the stakeholders, who suggested that Bataleon designers should create it. My recommendation is to either implement the proposed prototype or ask designers to create a new prototype and implement a version that they create.

Last problem regarding the solution lies in the functionality of the code. Currently it only provides users with a non perfect match, and when it does so it was observed to only provide one of 3 options. The issue seems to be located somewhere within the `findProduct()` function, however I was unable to solve it within the project timeframe. Possible solution is to reevaluate the scoring system, and to ensure that the data that is called from the database matches the user input, or is translated in a way that would allow the system to display a correct match.

6. Appendices

Appendix A

Board Finder V9.7

Advancement

Beginner



"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua."

Beginner

Intermediate



"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua."

Intermediate

Advanced



"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua."

Advanced

Board Finder V9.7

Advancement

Shape

Twin



"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua."

Twin

Directional Twin



"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua."

Directional Twin

Directional



"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua."


Directional

Advancement

Shape

Camber


Low Camber



Chill rides, big fun! These boards have mellow arches for playful turns, butters, and catching pow without a hitch.

Low Camber


Medium Camber



Happy medium all around! These boards offer a balance of pop, stability, and ease of turning, making them great for all-mountain adventures.

Medium Camber

High Camber



Grip and rip it all season! These boards have strong arches for powerful carves, precise control at high speeds, and locked-in landings on jumps.

High Camber

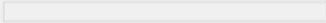
Advancement

Shape

Camber

Flexibility

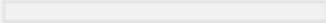
Soft



Effortless control! These boards bend easily, making them perfect for beginners, jibbing, and powder riding.

Soft

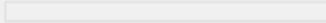
Medium



All-mountain magic! These boards offer a balance of forgiveness and stability for all riding styles and terrains.

Medium

High



Power and precision! These boards hold their edge at high speeds and excel in carving and aggressive riding.

High

Advancement


Shape

Camber

Flexibility

Fav terrain


Park



Built for tricks! These boards are playful and flexible, perfect for jumps, rails, and mastering new maneuvers.

Park


Groomed



Carve like a pro! These boards offer stability and control for fast cruising and laying down epic turns.

Groomed

Powder



Float like a dream! These boards have wider shapes and softer flex to keep you riding on top of fresh snow.

Powder

Advancement

Shape

Camber

Flexibility

Fav terrain

Sizing

Weight

Measurement

lbs

Weight in pounds

Boot Size

Size System

US

Size

Select boot size

Find my perfect board

Appendix B1

<https://eu.bataleon.com>

/

Appendix B2

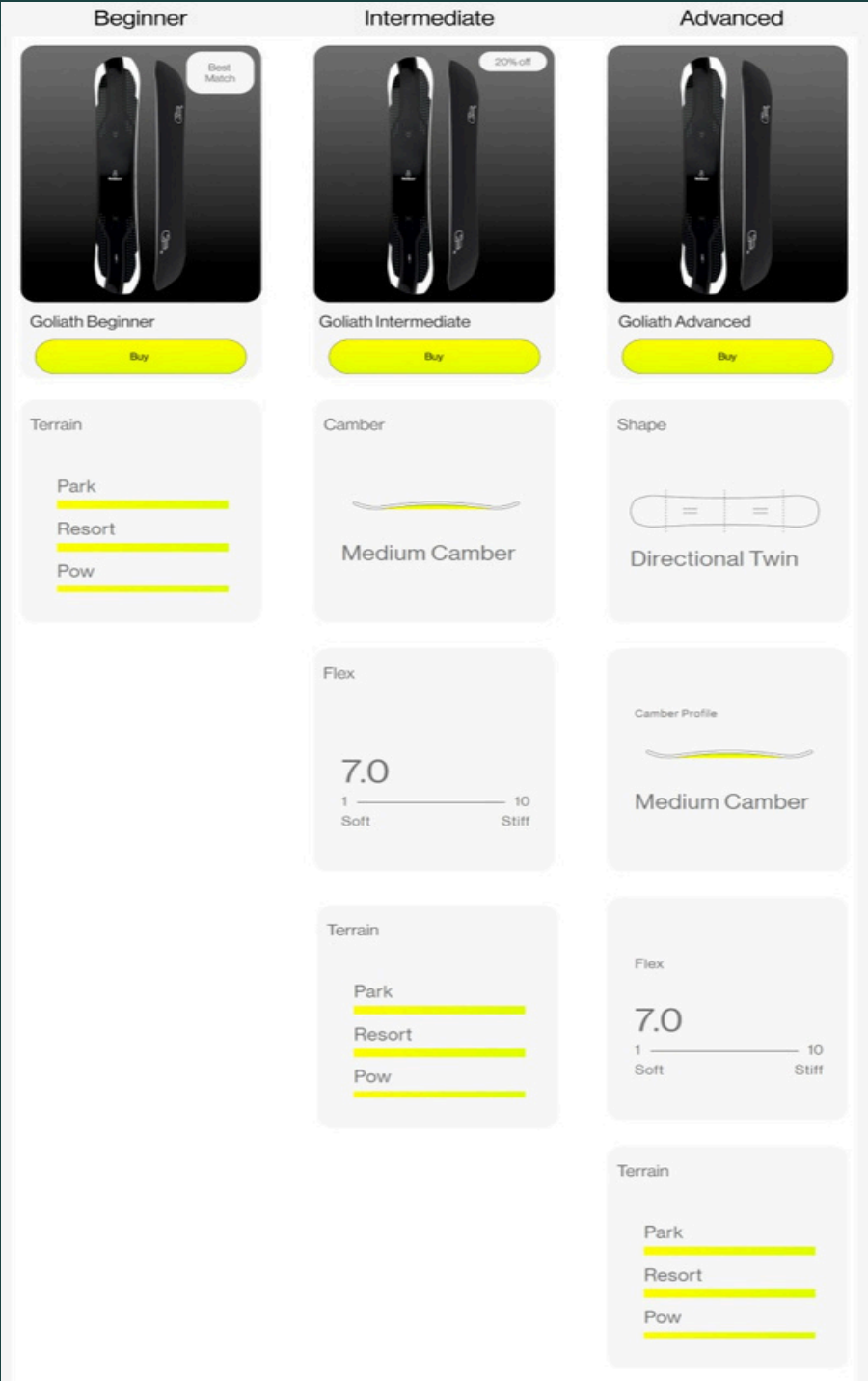
<https://eu.bataleon.com/products/bataleon-bataleon-20y-board-2024-mens-snowboards>

Appendix

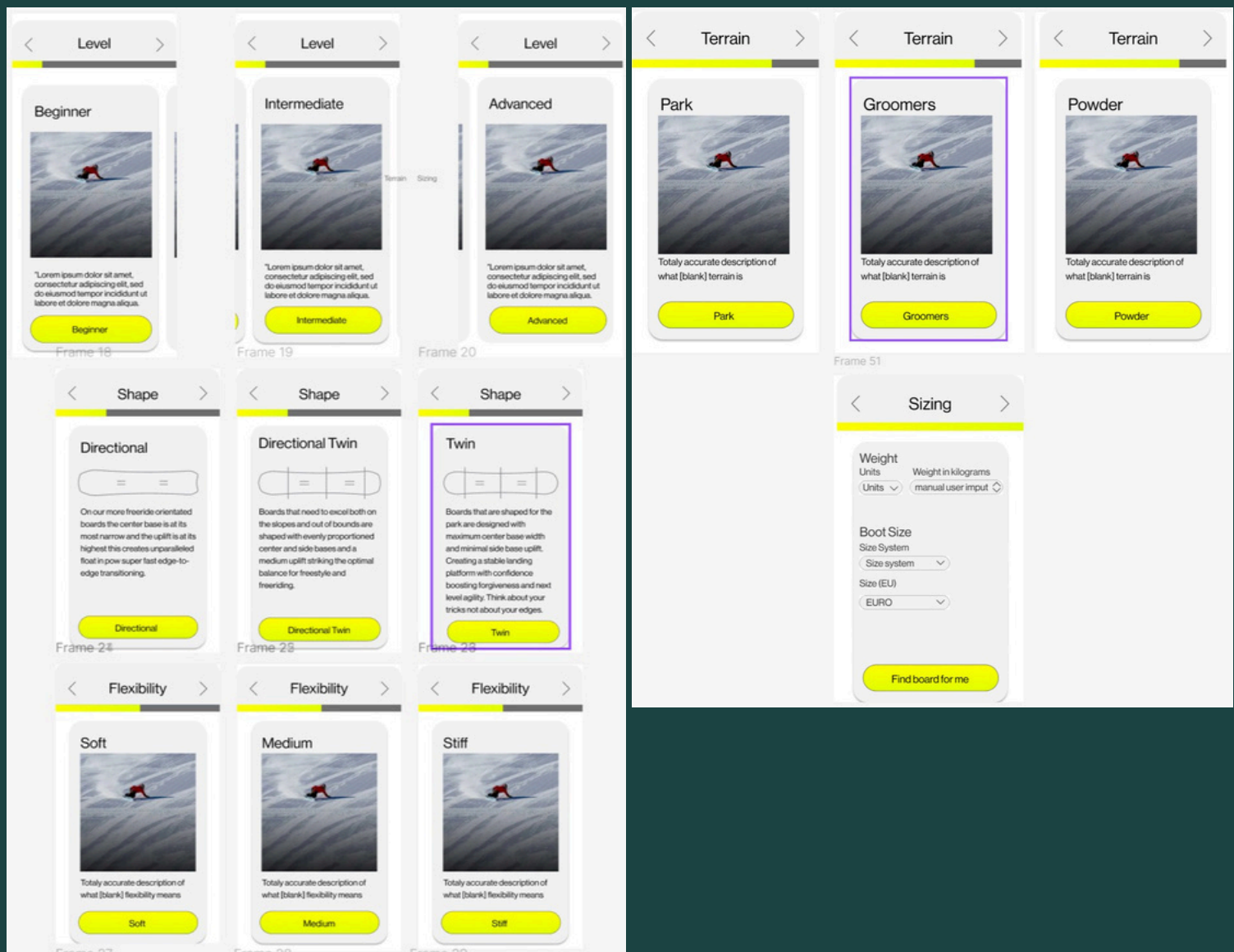
C

Appendix

D



Appendix E



Appendix F

```
function QuestionShape(){
document.getElementById("TemplateLvl").style.display="none";
document.getElementById("TemplateShape").style.display="block";
document.getElementById("TemplateCamber").style.display="none";
document.getElementById("TemplateFlex").style.display="none";
document.getElementById("TemplateTerrain").style.display="none";
document.getElementById("TemplateSizing").style.display="none";
console.log("Chosen_path should be at " + Chosen_path);
console.log("QuestionShape");}
```

Appendix G

```
function QuestionSizing(){
  if(Chosen_path===0){
    document.getElementById("RemoveIntermediate4").style.display="none";

    document.getElementById("RemoveBeginner3").style.display="none";
    document.getElementById("RemoveBeginner4").style.display="none";
    document.getElementById("TemplateLvl").style.display="none";
    document.getElementById("TemplateShape").style.display="none";
    document.getElementById("TemplateCamber").style.display="none";
    document.getElementById("TemplateFlex").style.display="none";
    document.getElementById("TemplateTerrain").style.display="none";
    document.getElementById("TemplateSizing").style.display="block";
    console.log("QuestionSizing");
  }
  else if(Chosen_path===1){
    document.getElementById("RemoveIntermediate4").style.display="none";
    document.getElementById("RemoveBeginner3").style.display="block";
    document.getElementById("RemoveBeginner4").style.display="block";
    document.getElementById("TemplateLvl").style.display="none";
    document.getElementById("TemplateShape").style.display="none";
    document.getElementById("TemplateCamber").style.display="none";
    document.getElementById("TemplateFlex").style.display="none";
    document.getElementById("TemplateTerrain").style.display="none";
    document.getElementById("TemplateSizing").style.display="block";
    console.log("QuestionSizing");
  }else{

document.getElementById("RemoveIntermediate4").style.display="block";
    document.getElementById("RemoveBeginner3").style.display="block";
    document.getElementById("RemoveBeginner4").style.display="block";
    document.getElementById("TemplateLvl").style.display="none";
    document.getElementById("TemplateShape").style.display="none";
    document.getElementById("TemplateCamber").style.display="none";
    document.getElementById("TemplateFlex").style.display="none";
    document.getElementById("TemplateTerrain").style.display="none";
    document.getElementById("TemplateSizing").style.display="block";
    console.log("QuestionSizing");}

}
```

Appendix H

```
<div class="finder-template" id="TemplateTerrain" style="display:none">
  <div class="card-row">
    <div class="Snowboard-finder-header">
      <div class="header-button" style="width:10%;"
onclick="QuestionLvl();">Advancement </div>
      <div id="RemoveIntermediate3" class="header-button remove"
onclick="QuestionShape();">Shape</div>
      <div id="RemoveBeginner1" class="header-button remove"
onclick="QuestionCamber();">Camber</div>

      <div id="RemoveBeginner2" class="header-button remove"
onclick="QuestionFlex();">Flexibility</div>
      <div class="header-button" onclick="QuestionTerrain();"
style="background-color: var(--darkest-grey); color: var(--lightest-grey);">Fav
terrain</div>

      <div class
="header-button"onclick="QuestionSizing();"style="display:none">Sizing</div>
    </div>
  </div>
  <div class="card-row" style="justify-content:start; background-color:
var(--darkest-grey);">
    <div class="progressbar5" > </div>
  </div>
  <div class="card-row" style="margin-top:2%;margin-bottom:2%;">
    <div class="card-column">
      <label class="checkbox"><div class="cardButton" name="style-button"
data-name="terrain" data-value ="Park"
style="height:100%"onclick="QuestionSizing();"><input type="checkbox" name="Terrain"
style="opacity:0;" onclick=TerrainPark();>
      <h4 style="margin-bottom:4%; margin-left:3%">Park</h4>
      <img class="fluid" width="100%" height="100%"
src=https://cdn.shopify.com/s/files/1/0553/2100/2151/files/product-finder-park.jpg?v=1
716291990/>
      <p style="margin-top: 5%;">Lorem ipsum dolor sit amet,
consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore
magna aliqua. </p>
      <div class="card-button-green"><p style="margin-top:
3%;">Park</p></div>
    </div></label>
  </div>
  <div class="card-column">
    <label class="checkbox"><div class="cardButton" name="style-button"
data-name="terrain" data-value ="Groomed"
style="height:100%"onclick="QuestionSizing();"><input type="checkbox" name="Terrain"
style="opacity:0;" onclick=TerrainGroomers();>
    <h4 style="margin-bottom:4%; margin-left:3%">Groomed</h4>
    <img class="fluid" width="100%" height="100%"
src=https://cdn.shopify.com/s/files/1/0553/2100/2151/files/product-finder-groomers.jpg
?v=1716291990/>
    <p style="margin-top: 5%;">Lorem ipsum dolor sit amet,
consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore
magna aliqua. </p>
    <div class="card-button-green"><p style="margin-top:
3%;">Groomed</p></div>
  </div></label>
  </div>
  <div class="card-column">
    <label class="checkbox"><div class="cardButton" name="style-button"
```



```

data-name="terrain" data-value ="Powder"
style="height:100%"onclick="QuestionSizing();" ><input type="checkbox" name="Terrain"
style="opacity:0;" onclick=TerrainPowder();>
    <h4 style="margin-bottom:4%; margin-left:3%">Powder</h4>
    <img class="fluid" width="100%" height="100%"
src=https://cdn.shopify.com/s/files/1/0553/2100/2151/files/product-finder-pow.jpg?v=17
16291989/>
    <p style="margin-top: 5%;">Lorem ipsum dolor sit amet,
consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore
magna aliqua. </p>
    <div class ="card-button-green"><p style="margin-top:
3%;">Powder</p></div>
    </div></label>
    </div>
    </div>
</div>

```

Appendix I

```

const snowboards = [

    {% paginate collections['all-snowboards'].products by 59 %}
    {% for product in collections['all-snowboards'].products %}
    {% for variant in product.variants %}
    {
        name: "{{product.title | append: "-" | append:
variant.option1}}",
        level: {{product.metafields.global.level}},
        shape: "{{product.metafields.global.shape}}",
        camber: "{{product.metafields.global.camber}}",
        flexibility: {{product.metafields.global.flex-1}},
        ratingPark: {{product.metafields.global.rating-1}},
        ratingGroomers: {{product.metafields.global.rating-5}},
        ratingPowder: {{product.metafields.global.rating-4}},
        bootSizeRange:
[Number({{variant.metafields.global.recommend-boot}})-3,Number({{variant.met
afields.global.recommend-boot}})+3],
        weightRange:
[Number({{variant.metafields.global.recommend-weight}})-3,
Number({{variant.metafields.global.recommend-weight}})+3]
    },
    {% endfor %}
    {% endfor %}
    {% endpaginate %}
];

```


Appendix J

```
function findProduct() {
    const form = document.getElementById('product-finder-form');
    const level =
document.querySelector('.cardButton[data-name="level"].selected')?.getAttribute('data-value');
    const shape =
document.querySelector('.cardButton[data-name="shape"].selected')?.getAttribute('data-value');
    const camber =
document.querySelector('.cardButton[data-name="camber"].selected')?.getAttribute('data-value');
    const flexibility =
document.querySelector('.cardButton[data-name="flexibility"].selected')?.getAttribute('data-value');
    const terrain =
document.querySelector('.cardButton[data-name="terrain"].selected')?.getAttribute('data-value');
    const bootSize = document.getElementById('boot-size-value');
    const weight = document.getElementById('weight-value');

    const filteredSnowboards = snowboards.filter(board =>
        board.level == level &&
        board.shape == shape &&
        board.camber == camber &&
        board.flexibility == flexibility &&
        board.terrain == terrain &&
        bootSize >= board.bootSizeRange[0] && bootSize <=
board.bootSizeRange[1] &&
        weight >= board.weightRange[0] && weight <= board.weightRange[1]
    );
    console.log(terrain + 'is saved terrain now HERE!!!');

    const bestMatches = filteredSnowboards.slice(0, 3);

    const resultList = document.getElementById('result-list');
    resultList.innerHTML = '';

    if (bestMatches.length === 0) {
        // Find the closest match
        let closestMatch = null;
        let highestScore = 0;

        snowboards.forEach(board => {
            let score = 0;
```

```

        if (board.level == level) score++;
        if (board.shape == shape) score++;
        if (board.camber == camber) score++;
        if (board.flexibility == flexibility) score++;
        if (board.terrain == terrain) score++;
        if (bootSize >= board.bootSizeRange[0] && bootSize <=
board.bootSizeRange[1]) score++;
        if (weight >= board.weightRange[0] && weight <=
board.weightRange[1]) score++;

        if (score > highestScore) {
            highestScore = score;
            closestMatch = board;
        }
    });

    resultList.innerHTML = `

No exact matches found. Showing the
closest match:</p>`;

    if (closestMatch) {
        const boardDiv = document.createElement('div');
        boardDiv.classList.add('card');
        boardDiv.innerHTML = `
            <h3>${closestMatch.name}</h3>
            <p id = "target">Level: ${closestMatch.level}</p>
            <p>Shape: ${closestMatch.shape}</p>
            <p>Camber: ${closestMatch.camber}</p>
            <p>Flexibility: ${closestMatch.flexibility}</p>
            <p>Terrain: ${closestMatch.terrain}</p>
            <p>Boot Size Range: ${closestMatch.bootSizeRange[0]} -
${closestMatch.bootSizeRange[1]}</p>
            <p>Weight Range: ${closestMatch.weightRange[0]} -
${closestMatch.weightRange[1]}</p>
        `;
        resultList.appendChild(boardDiv);
        console.log('There were no direct matches but it should give match
regardless')
    }
} else {
    bestMatches.forEach(board => {
        const boardDiv = document.createElement('div');
        boardDiv.classList.add('card');
        boardDiv.innerHTML = `
            <h3>${board.name}</h3>
            <p>Level: ${board.level}</p>
            <p>Shape: ${board.shape}</p>
            <p>Camber: ${board.camber}</p>
            <p>Flexibility: ${board.flexibility}</p>


```

```

        <p>Terrain: ${board.terrain}</p>
        <p>Boot Size Range: ${board.bootSizeRange[0]} -
        ${board.bootSizeRange[1]}</p>
        <p>Weight Range: ${board.weightRange[0]} -
        ${board.weightRange[1]}</p>
    `;
    resultList.appendChild(boardDiv);
    console.log('It should be giving matches now')
  });
}

//document.getElementById('result').classList.remove('hidden');
}

```

Appendix K

Boards that are provided as closest match

```

{
  name: "Party Wave-164",
  level: ["Intermediate","Expert"],
  shape: "Directional",
  camber: "medium",
  flexibility: 4.0,
  ratingPark: 5,
  ratingGroomers: 9,
  ratingPowder: 9,
  bootSizeRange: [Number()-3,Number()+3],
  weightRange: [Number()-3, Number()+3]
}

{
  name: "Bataleon 20Y Board-156",
  level: ["Intermediate","Expert"],
  shape: "Directional Twin",
  camber: "medium",
  flexibility: 6.0,
  ratingPark: 9,
  ratingGroomers: 9,
  ratingPowder: 9,
  bootSizeRange: [Number()-3,Number()+3],
  weightRange: [Number()-3, Number()+3]
}

{
  name: "board finder product 2-156",

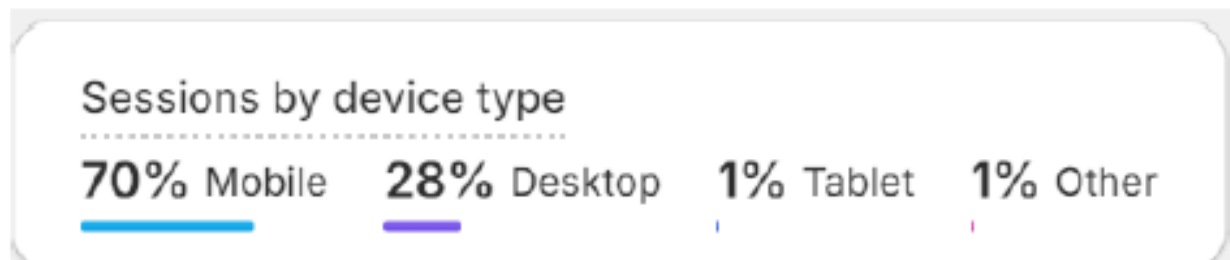
```

```
level: ["Intermediate","Expert"],
shape: "Directional Twin",
camber: "",
flexibility: 6.0,
ratingPark: 9,
ratingGroomers: 9,
ratingPowder: 9,
bootSizeRange: [Number()-3,Number()+3],
weightRange: [Number()-3, Number()+3]
},
```

One of the other boards for comparison

```
{
  name: "Whatever-138",
  level: ["Beginner","Intermediate","Expert"],
  shape: "Directional Twin",
  camber: "medium",
  flexibility: 5.0,
  ratingPark: 10,
  ratingGroomers: 9,
  ratingPowder: 8,
  bootSizeRange: [Number(23.0)-3,Number(23.0)+3],
  weightRange: [Number(51.0)-3, Number(51.0)+3]
}
```

Appendix L



7. References

[1]GilPress. (2024, February 4). Internet Traffic from Mobile Devices Stats (2024).

What'S the Big Data? <https://whatsthebigdata.com/mobile-internet-traffic/>