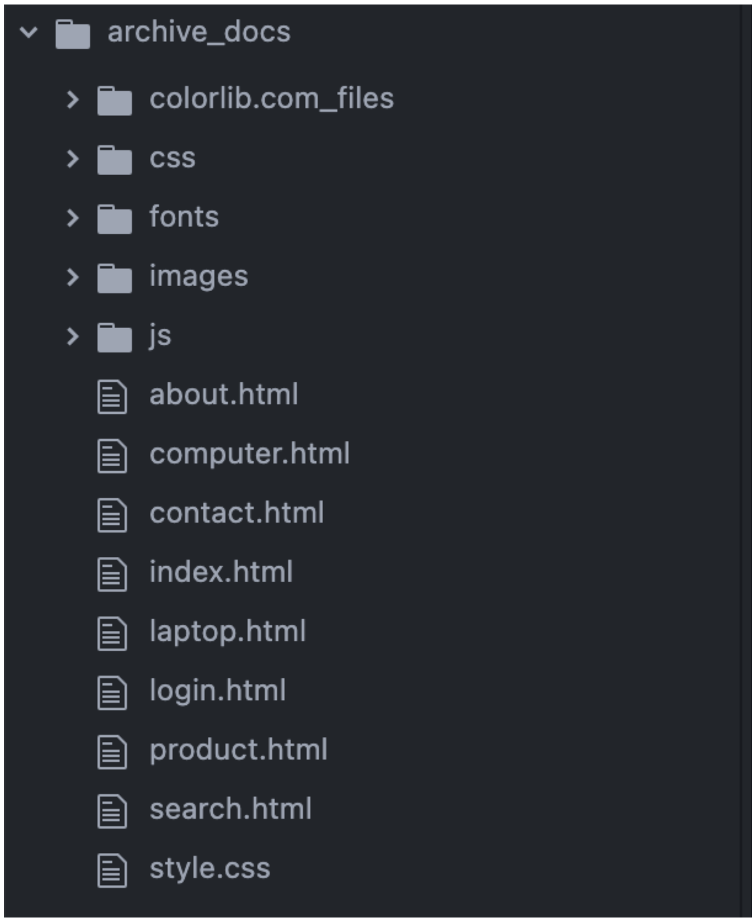
Week_5_wa
Focusing on the front-end development, design a proposal for the online store discussed in the previous unit's Written Assignment. Front-end engineering is also known as “client-side” or “client-facing” engineering which includes developing everything the user sees and interacts with. It may be helpful to focus on the front-end features present in the e-commerce site you chose in previous Units. This proposal should include a detailed description of the overall look and feel of the website including:

- a description of how the site will be navigated such as pages and any links or handles on the page

Navigation of e-commerce sites (futurama.com) is quite an important part of the web application. People prefer logical and easy navigational sites. It is important to understand that good navigation makes it easy to find our products, to pay for the order and ask for support if any. Basically we have to create an application that suits our customer very best. Anna Fitzgerald defines: “Website navigation is a collection of user interface components that allows visitors to find content and features on a site”.

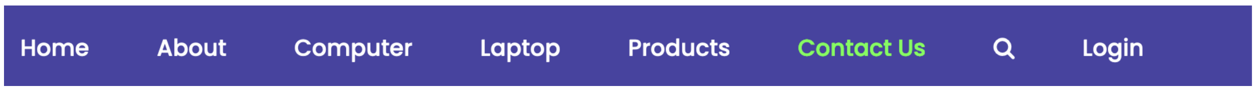


```
archive_docs
    colorlib.com_files
    css
    fonts
    images
    js
    about.html
    computer.html
    contact.html
    index.html
    laptop.html
    login.html
    product.html
    search.html
    style.css
```

Our website contains the following pages

Page name	Description
index.html	Main page, which contains links to other pages. Also means home, from the upper navigational tags.
about.html	Contains information about us, links to social accounts: twitter, facebook, linkedin and instagram. In addition, information on how to contact us.
computer.html	A place where our deals on computers are shown. Starting point for buying computers.
laptop.html	A place where our deal on laptops are shown. Starting point for buying laptops.
products.html	A place where computer/laptop accessories are shown. Starting point for buying accessories.
contact.html	If a visitor has a problem, question, suggestion or any other idea, this page serves as a messenger.
search.html	This page is a search for our products and accessories.
login.html	If a user has an account, it is an entry point to our site. This page also allows creating a new account.

Our site allows navigation using navigational menu



Navigational links to other pages are in page headers.

It is worth adding Google’s User Flow report in order to understand how users use the web application and in what parts.

It is also important to design for different types of screens, desktop, mobile phones or tablets.

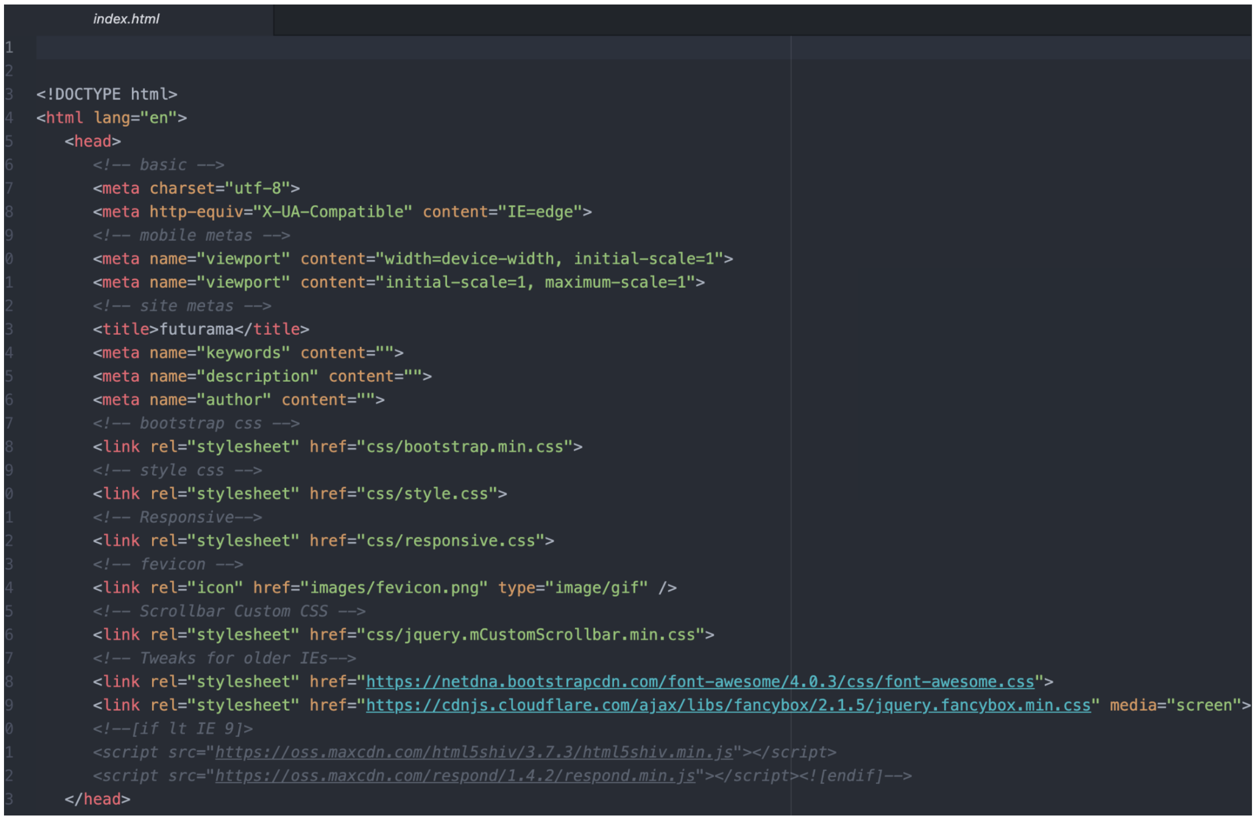
- a description of how then you will use HTML, CSS, JavaScript, and any other languages like Bootstrap, Sass, etc.

Basically the web is built by two main components: a web browser and a web server. The client sends requests to the server and the server serves the request. They share a common method/protocol which is called Application Programmable Interface.

HTML stands for Hyper Text Markup Language. Information shown in the browser in pages, pages contains different elements like: links, divs, titles and more. It is like an instruction set about style, fonts, context, format and so on.

Kingsley Ubah continues: “HTML helps you structure your page into elements such as paragraphs, sections, headings, navigation bars, and so on. ”

Our index.html looks like:

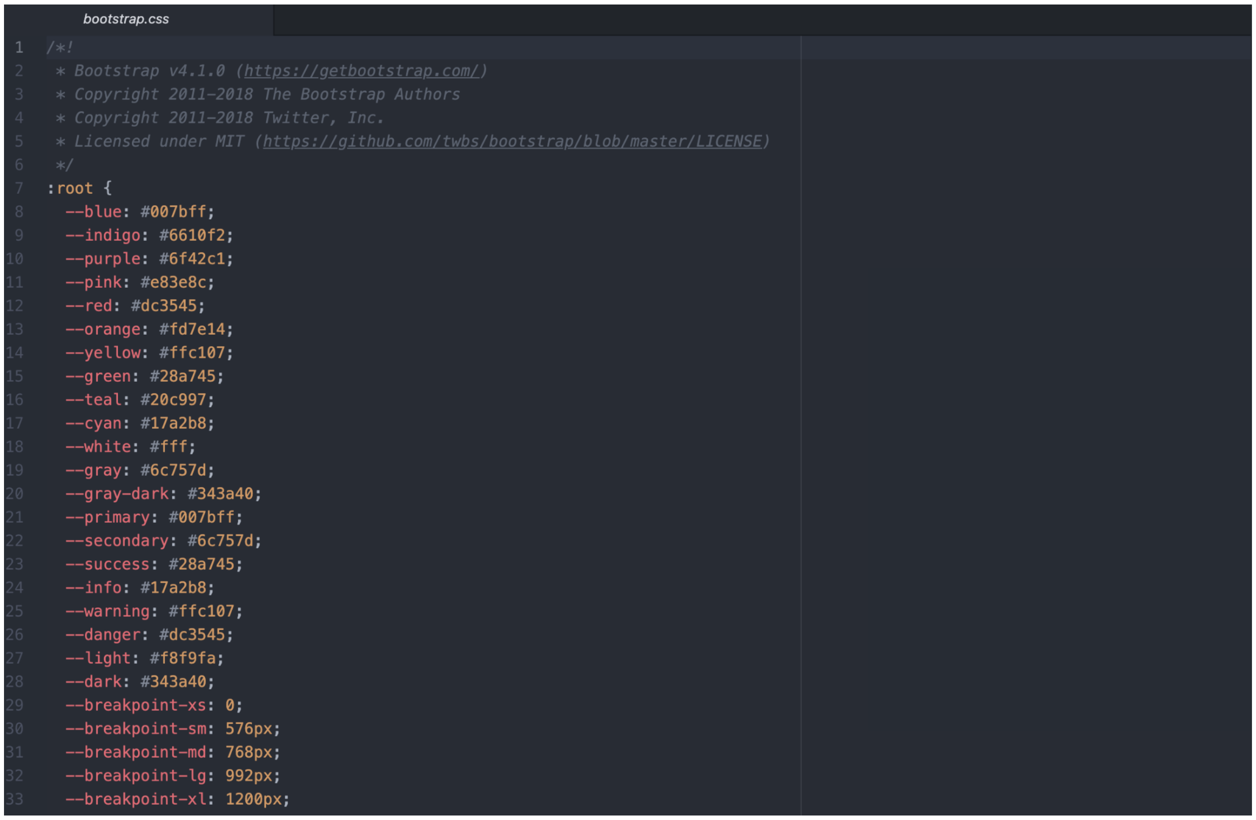


index.html

```


<!DOCTYPE html>
<html lang="en">
   <head>
      <!-- basic -->
      <meta charset="utf-8">
      <meta http-equiv="X-UA-Compatible" content="IE=edge">
      <!-- mobile metas -->
      <meta name="viewport" content="width=device-width, initial-scale=1">
      <meta name="viewport" content="initial-scale=1, maximum-scale=1">
      <!-- site metas -->
      <title>futurama</title>
      <meta name="keywords" content="">
      <meta name="description" content="">
      <meta name="author" content="">
      <!-- bootstrap css -->
      <link rel="stylesheet" href="css/bootstrap.min.css">
      <!-- style css -->
      <link rel="stylesheet" href="css/style.css">
      <!-- Responsive-->
      <link rel="stylesheet" href="css/responsive.css">
      <!-- fevicon -->
      <link rel="icon" href="images/fevicon.png" type="image/gif" />
      <!-- Scrollbar Custom CSS -->
      <link rel="stylesheet" href="css/jquery.mCustomScrollbar.min.css">
      <!-- Tweaks for older IEs-->
      <link rel="stylesheet" href="https://netdna.bootstrapcdn.com/font-awesome/4.0.3/css/font-awesome.css">
      <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/fancybox/2.1.5/jquery.fancybox.min.css" media="screen">
      <!--[if lt IE 9]>
      <script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>
      <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script><![endif]-->
   </head>
```

The author also explains what is css: “While HTML is a **markup language** used to format/structure a web page, CSS is a **design language** that you use to make your web page look nice and presentable. ” CSS also helps to layout our elements on the page. In addition css defines colors, background, typefaces, spaces, padding and even more.

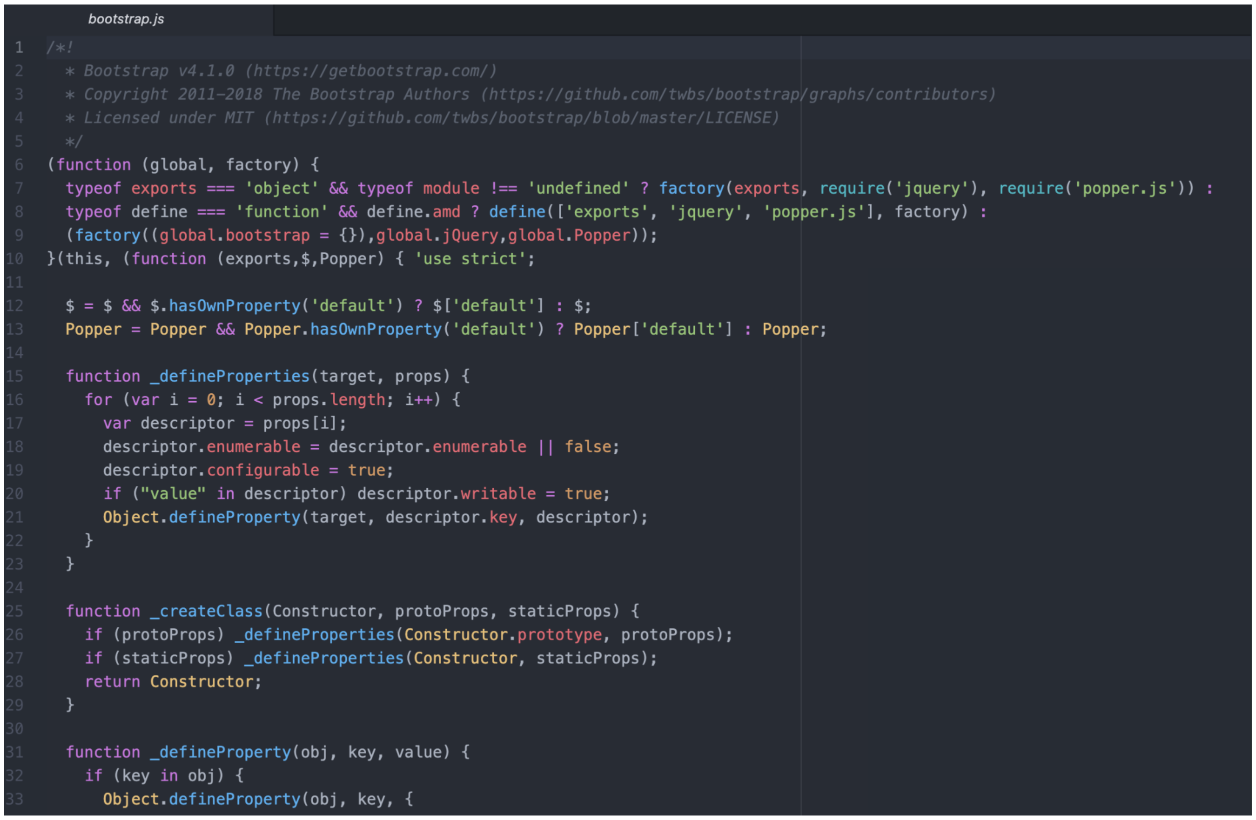
Our CSS (one of them) looks like this one:


bootstrap.css

```css
/*!
 * Bootstrap v4.1.0 (https://getbootstrap.com/)
 * Copyright 2011-2018 The Bootstrap Authors
 * Copyright 2011-2018 Twitter, Inc.
 * Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)
 */
:root {
  --blue: #007bff;
  --indigo: #6610f2;
  --purple: #6f42c1;
  --pink: #e83e8c;
  --red: #dc3545;
  --orange: #fd7e14;
  --yellow: #ffc107;
  --green: #28a745;
  --teal: #20c997;
  --cyan: #17a2b8;
  --white: #fff;
  --gray: #6c757d;
  --gray-dark: #343a40;
  --primary: #007bff;
  --secondary: #6c757d;
  --success: #28a745;
  --info: #17a2b8;
  --warning: #ffc107;
  --danger: #dc3545;
  --light: #f8f9fa;
  --dark: #343a40;
  --breakpoint-xs: 0;
  --breakpoint-sm: 576px;
  --breakpoint-md: 768px;
  --breakpoint-lg: 992px;
  --breakpoint-xl: 1200px;
```

What do we have until now? A markup, design language. How do we use API? Right, we need a programming language, called JavaScript. Native language of the web.

Example of JS code:

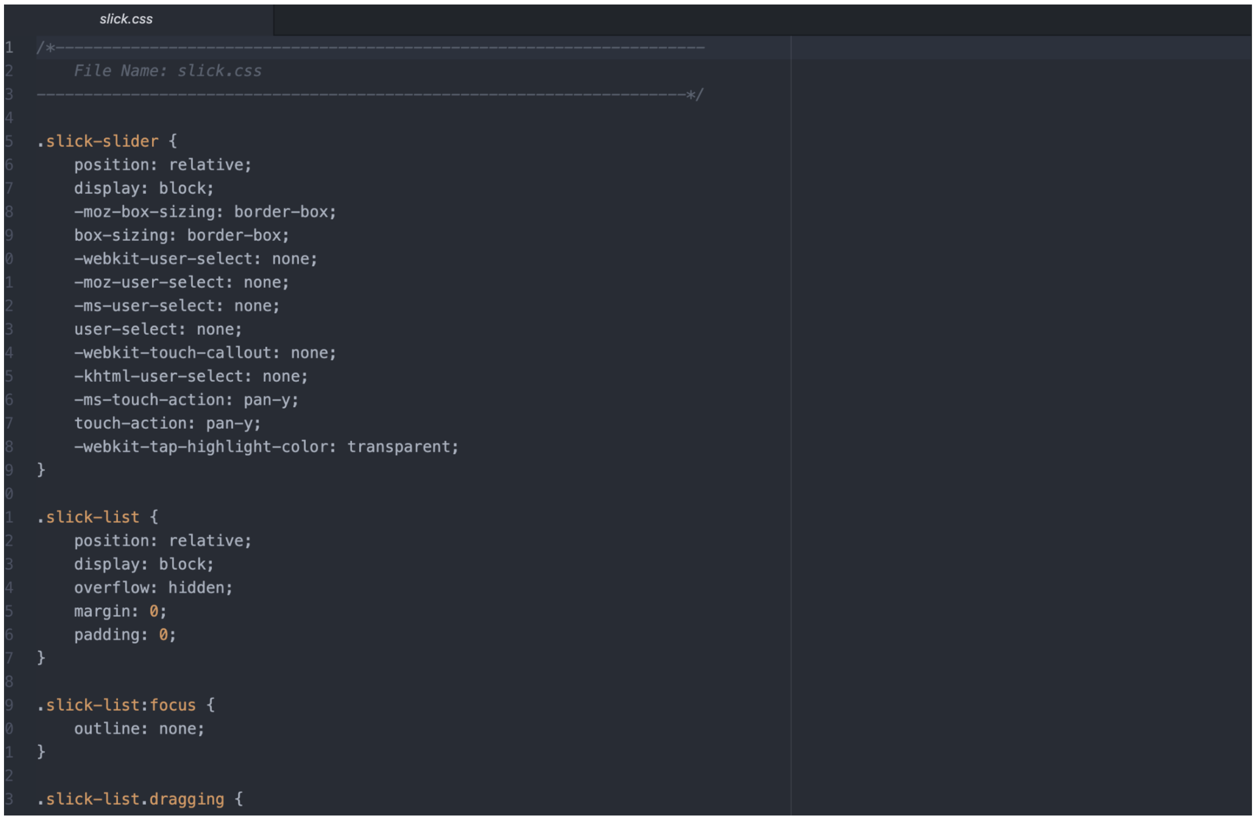


```javascript
/*!
  * Bootstrap v4.1.0 (https://getbootstrap.com/)
  * Copyright 2011-2018 The Bootstrap Authors (https://github.com/twbs/bootstrap/graphs/contributors)
  * Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)
  */
(function (global, factory) {
  typeof exports === 'object' && typeof module !== 'undefined' ? factory(exports, require('jquery'), require('popper.js')) :
  typeof define === 'function' && define.amd ? define(['exports', 'jquery', 'popper.js'], factory) :
  (factory((global.bootstrap = {}),global.jQuery,global.Popper));
}(this, (function (exports,$,Popper) { 'use strict';

  $ = $ && $.hasOwnProperty('default') ? $['default'] : $;
  Popper = Popper && Popper.hasOwnProperty('default') ? Popper['default'] : Popper;

  function _defineProperties(target, props) {
    for (var i = 0; i < props.length; i++) {
      var descriptor = props[i];
      descriptor.enumerable = descriptor.enumerable || false;
      descriptor.configurable = true;
      if ("value" in descriptor) descriptor.writable = true;
      Object.defineProperty(target, descriptor.key, descriptor);
    }
  }

  function _createClass(Constructor, protoProps, staticProps) {
    if (protoProps) _defineProperties(Constructor.prototype, protoProps);
    if (staticProps) _defineProperties(Constructor, staticProps);
    return Constructor;
  }

  function _defineProperty(obj, key, value) {
    if (key in obj) {
      Object.defineProperty(obj, key, {
```

Next a bootstrap. Bootstrap is an open source web framework which facilitates web development in a more fast, robust and I would say more controllable manner. Its objective is to create responsive web applications which will be working the same on different platforms like: desktops, mobile phones and tablets. Responsive means to react to a user's interaction, to give feedback for actions.

Without using bootstrap we could not develop our navigation, grid systems, UI controls and buttons.

Sass extends CSS functionality. And it looks:


slick.css

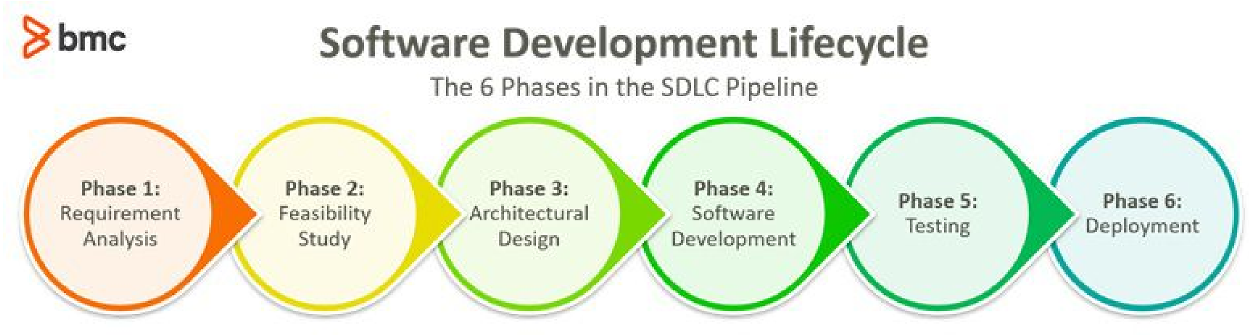
```css
/*---------------------------------------------------------------------
    File Name: slick.css
---------------------------------------------------------------------*/

.slick-slider {
    position: relative;
    display: block;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
    -webkit-user-select: none;
    -moz-user-select: none;
    -ms-user-select: none;
    user-select: none;
    -webkit-touch-callout: none;
    -khtml-user-select: none;
    -ms-touch-action: pan-y;
    touch-action: pan-y;
    -webkit-tap-highlight-color: transparent;
}

.slick-list {
    position: relative;
    display: block;
    overflow: hidden;
    margin: 0;
    padding: 0;
}

.slick-list:focus {
    outline: none;
}

.slick-list.dragging {
```

Next thing is packaging. There is a package manager called webpack.

The last important thing is web hosting. We used github and it allows us to deploy our app as github pages. If we need a functional application without worrying about domain name, it is the perfect option so far.

- an outline of how you will apply the SDLC to the development in this front-end

The SDLC as flow looks


bmc
Software Development Lifecycle
The 6 Phases in the SDLC Pipeline
Phase 1: Requirement Analysis
Phase 2: Feasibility Study
Phase 3: Architectural Design
Phase 4: Software Development
Phase 5: Testing
Phase 6: Deployment

taken from this source

We started with requirement gathering, what our web application serves for, what are our users, what we have to sell and a lot of other requirements. Then we analyzed technical, legal, economical, operational factors. Architectural design added some restrictions and made a path where we go and how. Then we started to divide our web application into functional modules, drawn figmas and started to develop. During the development, for each executable entry we also write tests. We work with Sprints. The longevity of the Sprint is two weeks. At the end of each Sprint we show our customers what has been done. We use github Actions. Each commit builds our web application with recent changes and deploys.

References:

1. Website navigation https://blog.hubspot.com/website/main-website-navigation-ht
2. Kingsley Ubah, 2020. Building block of web https://www.freecodecamp.org/news/html-css-and-javascript-explained-for-beginners/
3. https://www.hostinger.com/tutorials/what-is-bootstrap/
4. What is Sass - https://www.w3schools.com/sass/sass_intro.asp
5. Webpack - https://github.com/jantimon/html-webpack-plugin