Objective:

Study the basics of HTML and CSS to build trivial site layouts.

Theory:

1. Read about HTML.
2. Read about CSS.
3. Read about codestyle
4. Use https://developer.mozilla.org/en-US/ as documentation to get information about tags, CSS properties, etc.
5. Read about responsive/adaptive design.
6. Read about *display: flex*.
7. Read about *CSS media queries*.

Task:

Deadline: 4 days

Intro:

Responsive design involves adapting a web page for all possible screen resolutions. Differences between mobile/tablet/desktop resolutions can include:

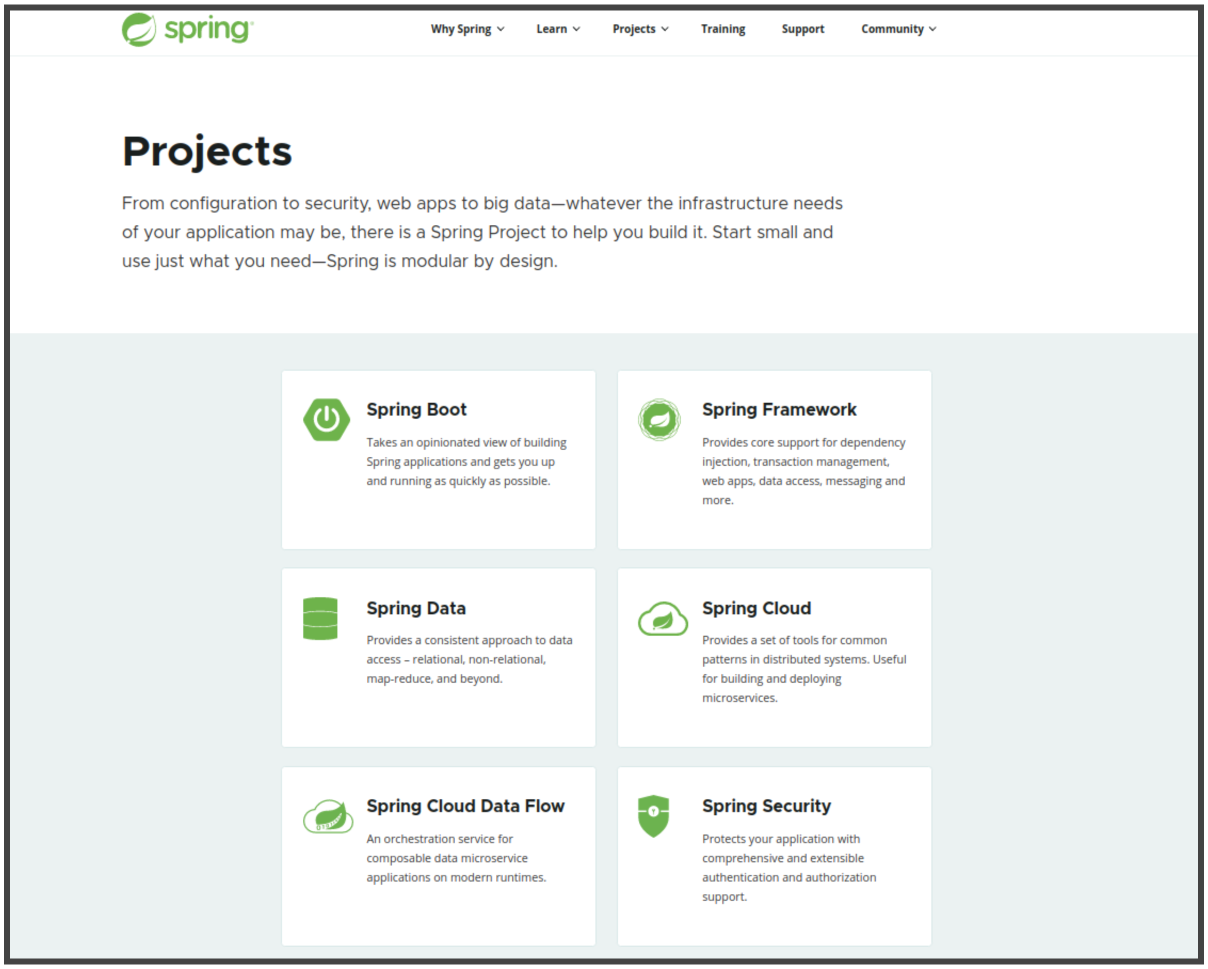
- different dimensions;
- displaying certain components for desktop but hiding them for mobile and vice versa;
- significant layout changes (compare YouTube.com in desktop and mobile).

Small changes between two resolutions can be made with CSS media queries, but for more serious changes, it's not enough to override some CSS properties to adapt the desktop version to mobile. In such cases, the only solution is to have two separate versions of your application and render them based on screen width. Feel free to read through this article to understand more.

Description:

The goal is to have HTML + CSS files to have the same result as the mockup below (link to the original).

Mockup:



Projects

From configuration to security, web apps to big data—whatever the infrastructure needs of your application may be, there is a Spring Project to help you build it. Start small and use just what you need—Spring is modular by design.

Spring Boot

Takes an opinionated view of building Spring applications and gets you up and running as quickly as possible.

Spring Framework

Provides core support for dependency injection, transaction management, web apps, data access, messaging and more.

Spring Data

Provides a consistent approach to data access – relational, non-relational, map-reduce, and beyond.

Spring Cloud

Provides a set of tools for common patterns in distributed systems. Useful for building and deploying microservices.

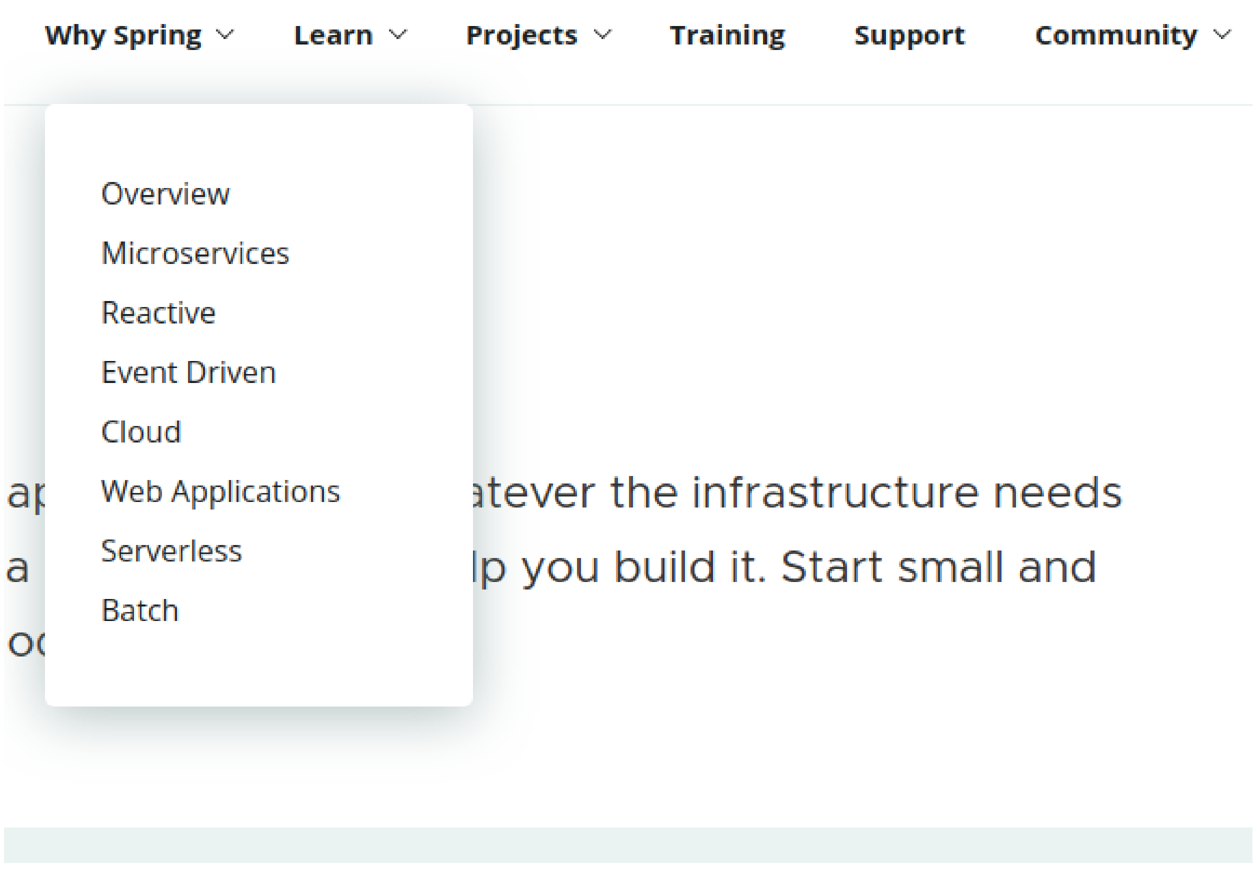
Spring Cloud Data Flow

An orchestration service for composable data microservice applications on modern runtimes.

Spring Security

Protects your application with comprehensive and extensible authentication and authorization support.

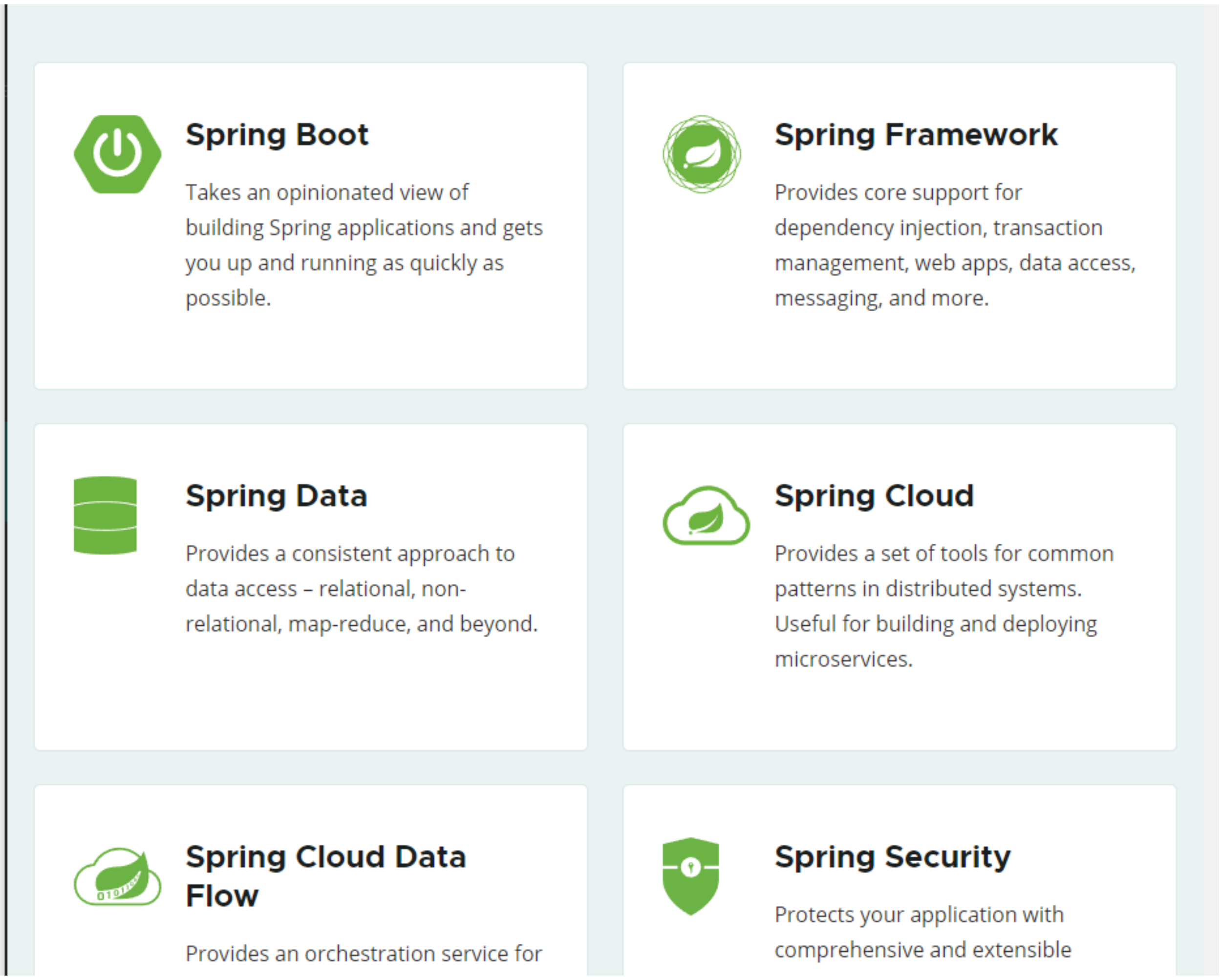
Please don’t forget about the dropdown menu, it’s part of the scope too.



ap atever the infrastructure needs a lp you build it. Start small and o

Requirements:

1. No need to copy the whole page from the link above, only what is on the mockup image.
2. Setup prettier
3. All dimensions, paddings, etc. - at your choice, don’t try to copy these values from the original; it’s a waste of time.
4. The site should look like on the mockup for screen width >= 1000px.
5. For width >= 800px and < 1000px, project blocks take almost the whole page width:



Spring Boot

Takes an opinionated view of building Spring applications and gets you up and running as quickly as possible.

Spring Framework

Provides core support for dependency injection, transaction management, web apps, data access, messaging, and more.

Spring Data

Provides a consistent approach to data access – relational, non-relational, map-reduce, and beyond.

Spring Cloud

Provides a set of tools for common patterns in distributed systems. Useful for building and deploying microservices.

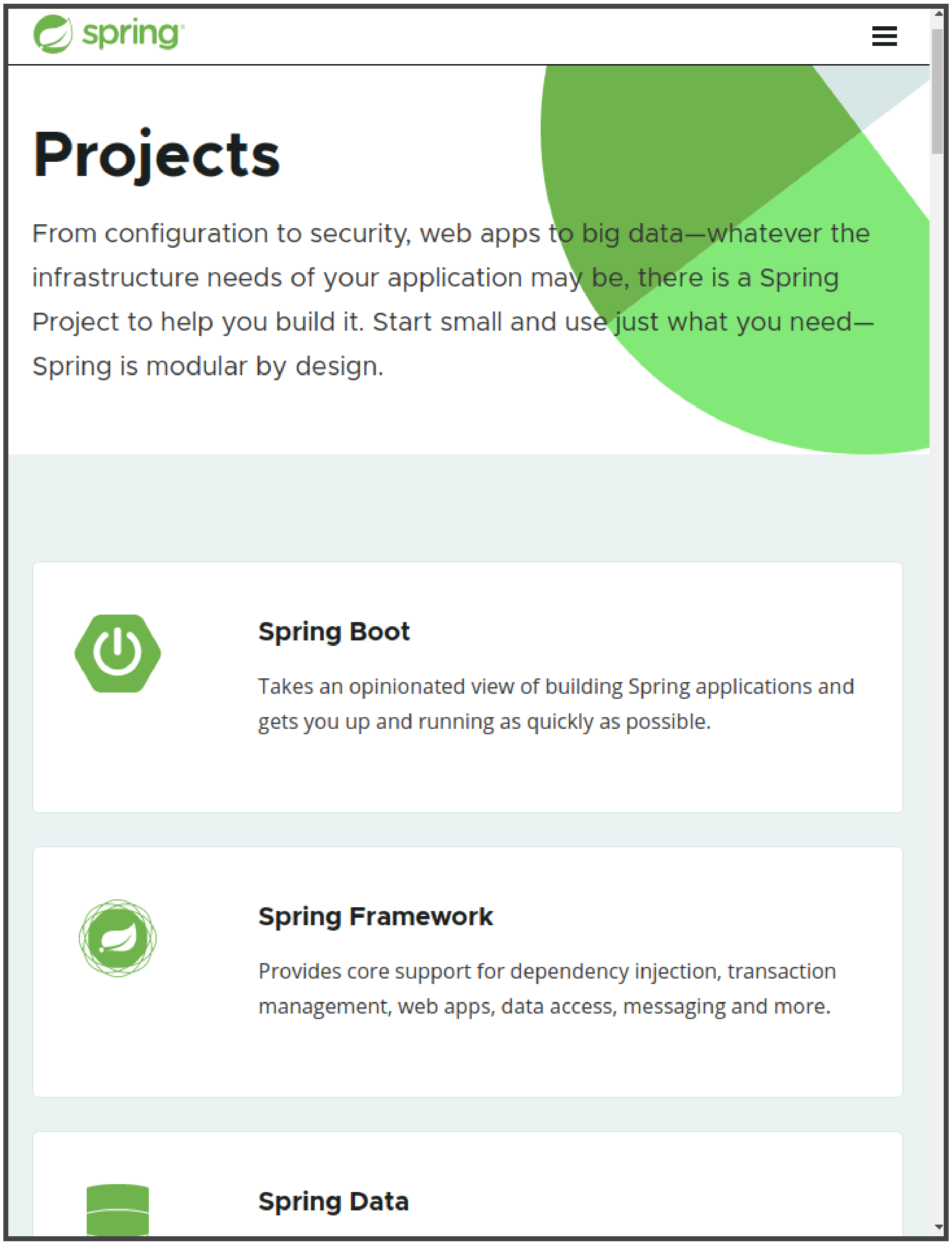
Spring Cloud Data Flow

Provides an orchestration service for

Spring Security

Protects your application with comprehensive and extensible

6. For width < 800px:


spring

Projects

From configuration to security, web apps to big data—whatever the infrastructure needs of your application may be, there is a Spring Project to help you build it. Start small and use just what you need—Spring is modular by design.

Spring Boot

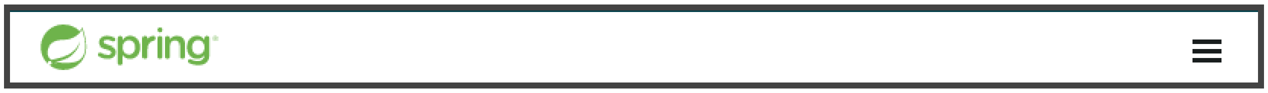
Takes an opinionated view of building Spring applications and gets you up and running as quickly as possible.

Spring Framework

Provides core support for dependency injection, transaction management, web apps, data access, messaging and more.

Spring Data

7. If the screen width is less than 900px, then header links aren't shown at all. The header is shown like this (sidepane open/hide functionality will be added in future tasks, so the burger menu is not clickable for now):



8. The only place where you can use “position: absolute” in this task is the dropdown menu. Using it in another place is a bad practice.
9. All dimensions, paddings are at your discretion.
10. Do not use any CSS libraries.
11. Browser support: Google Chrome (latest).

Questions:

1. What are the differences between *display: inline* | *block* | *inline-block?*
2. What is the ‘display’ CSS property for *<div>, <p>, <span>, <li>, <a>*?
3. What are the differences between *position: absolute | relative | fixed*.
4. When would you use *position: absolute*?
5. If an element has *position: absolute* and *top: 0* it is going to the top of the window. How can we make this element go to the top of the other element, not the window?
6. How to make a sidebar attached to the left of the screen even during scroll?
7. What CSS properties would you use to implement show/hide of the sidepane (example, open mobile version and click on the menu icon in the top left of the screen)? What CSS properties would you set for the sidepane in open state? What CSS properties would you set for the sidepane in hidden state?
8. Read about *flex-grow*, *flex-basis* and *flex-shrink*.
9. Given a div (container) with 3 inner divs (items):
   a. Container has a **display: *flex***.
   b. Every item has ***flex-grow: 1***
   c. First item has ***flex-basis: 100px***

What will be the width of every item?

10. What is code style and why do we need it?