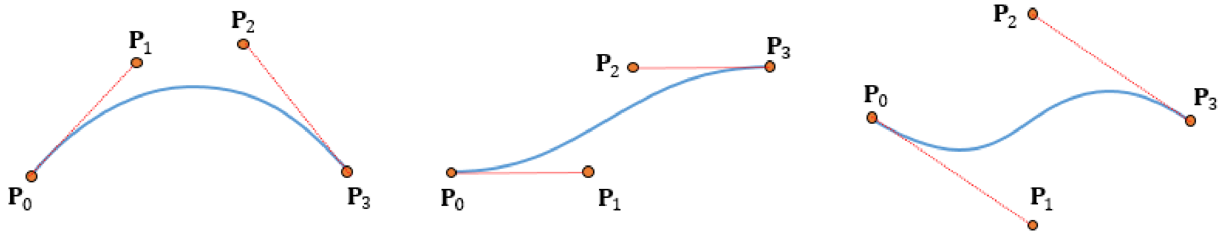
Main exercises:

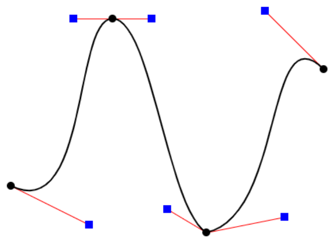
1. Curves, Classes, and Modules (coding required)

A Bezier curve is defined by a set of control points $\mathbf{P}_0$ through $\mathbf{P}_n$, where n is called its order (n = 1 for linear, 2 for quadratic, 3 for cubic).



$$\mathbf{B}(t) = (1-t)^3\mathbf{P}_0 + 3(1-t)^2 t\mathbf{P}_1 + 3(1-t)t^2\mathbf{P}_2 + t^3\mathbf{P}_3 \qquad (0 \le t \le 1)$$

A Bezier path is a line defined by several Bezier curve segments joined together and a flag to indicate if the path is closed. Each segment independently can be of order 1, 2, or 3.

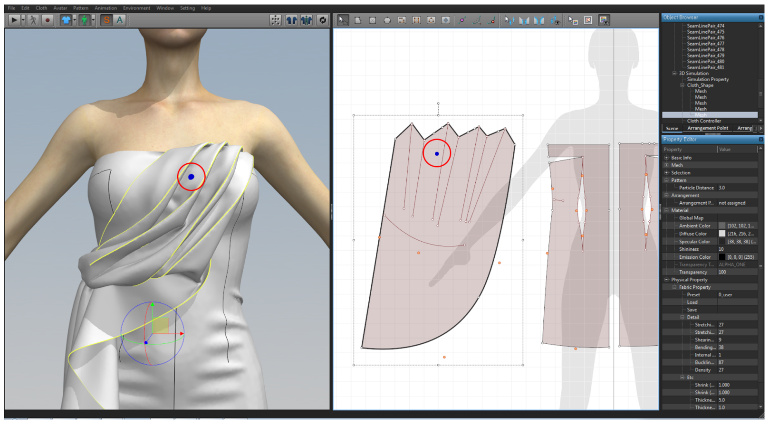


Create a ***BezierCurve2*** class representation using Typescript. The 2 means that all control points are 2D. The class should provide the implementation for an *eval* method in which given a parameter t between 0 and 1 returns the corresponding point in the curve.
Create a ***BezierPath2*** class representation in another file. The class should provide the implementation for a *getSegment* method in which given a segment index it returns the corresponding Bezier curve.

How would your representation handle the support of editing path control points in our application? Which members would you expose to do that?
How would your representation handle converting one of the path curved segments to a straight line? Which members would you expose to do that?

2. Picking Geometry (no coding required)

We have a 3D view where we can visualize all the garment pieces (triangle meshes). We want to have interactive feedback for selection and draw an outline of the piece the mouse is over at every moment.

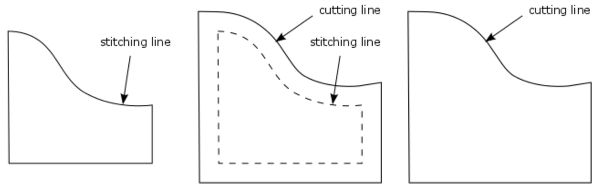


What flow of operations would you do to know which piece is below the cursor?

Each piece could have tens of thousands of polygons. What data structures can we use to accelerate geometry intersection queries and avoid framerate dropping?

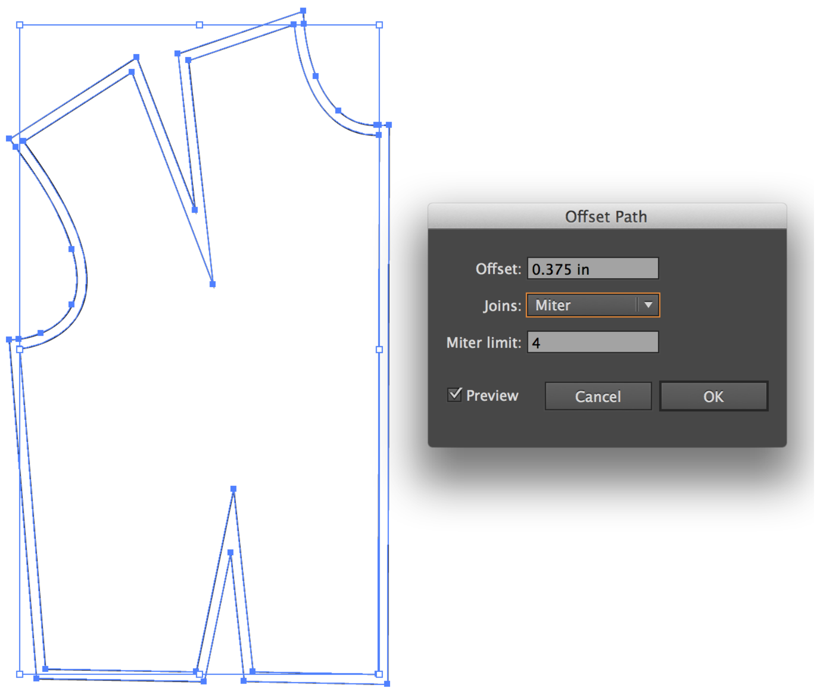
3. Seam Allowance (no coding required)

Once a design has been approved in our digital replica, in order to prepare the pattern for production and cutting we need to add seam allowances. Seam allowances give the extra offset to the base stitching lines needed to sew the pieces together, following specifications of the necessary bias to add on each side of a garment piece.



Could you think of an algorithm to generate the complete cutting lines automatically from the stitching lines?

Look at the following picture, is your algorithm able to handle all corners correctly?



Offset Path

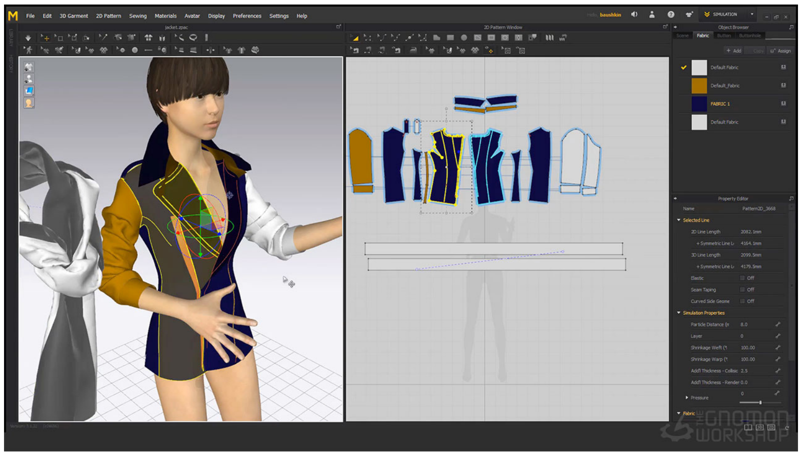
Offset: 0.375 in

Joins: Miter

Miter limit: 4

4. Reactive Components (no coding required)

We want a single-page web application with different 2D/3D views and a properties editor, similar to any Desktop modeling or game engine package.



✓	Default Fabric
	Default_Fabric
	FABRIC 1
	Default Fabric

Property Editor

Name	Pattern2D_3668
Selected Line	
2D Line Length	2082.1mm
+ Symmetric Line L	4164.1mm
3D Line Length	2099.5mm
+ Symmetric Line L	4179.5mm
Elastic	Off
Seam Taping	Off
Curved Side Geome	Off
Simulation Properties	
Particle Distance (r	8.0
Layer	0
Shrinkage Weft (%	100.00
Shrinkage Warp (%	100.00
Add'l Thickness - Collisic	2.5
Add'l Thickness - Render	0.0
Pressure	0
Fabric	

How would you structure the web application code to keep all the web components reactive to the user input?

Example use cases:

a) The user selects a garment piece in 3D, and all the selected piece attributes like textile material appear in the properties editor.

b) The user changes the textile material in the properties editor and automatically 3D view reflects the new material.

Optional exercises

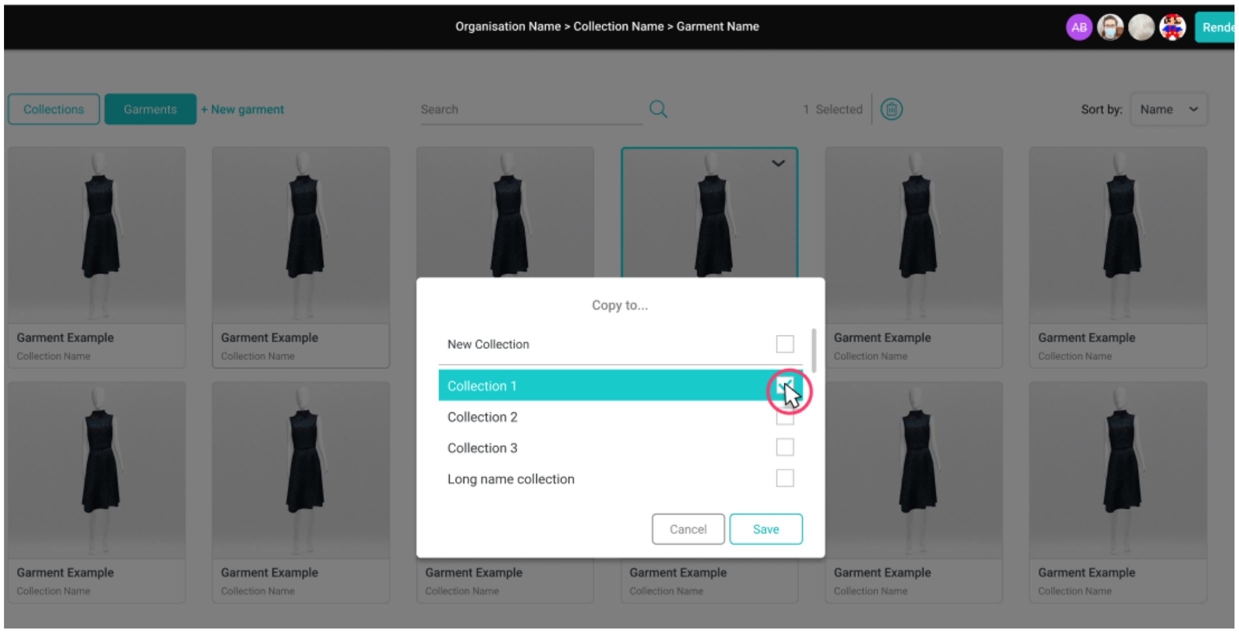
5. React & styled-components (coding required)

Given the following UI mock-up implement the foreground dialog. The idea is to develop something similar to it in style and composition, with a couple of simple reusable React Components using hooks and styled-components.

Efforts should be centered around understanding the UI hierarchy of components, style, and layout. A static state should be defined outside the component and no event treatment is needed.

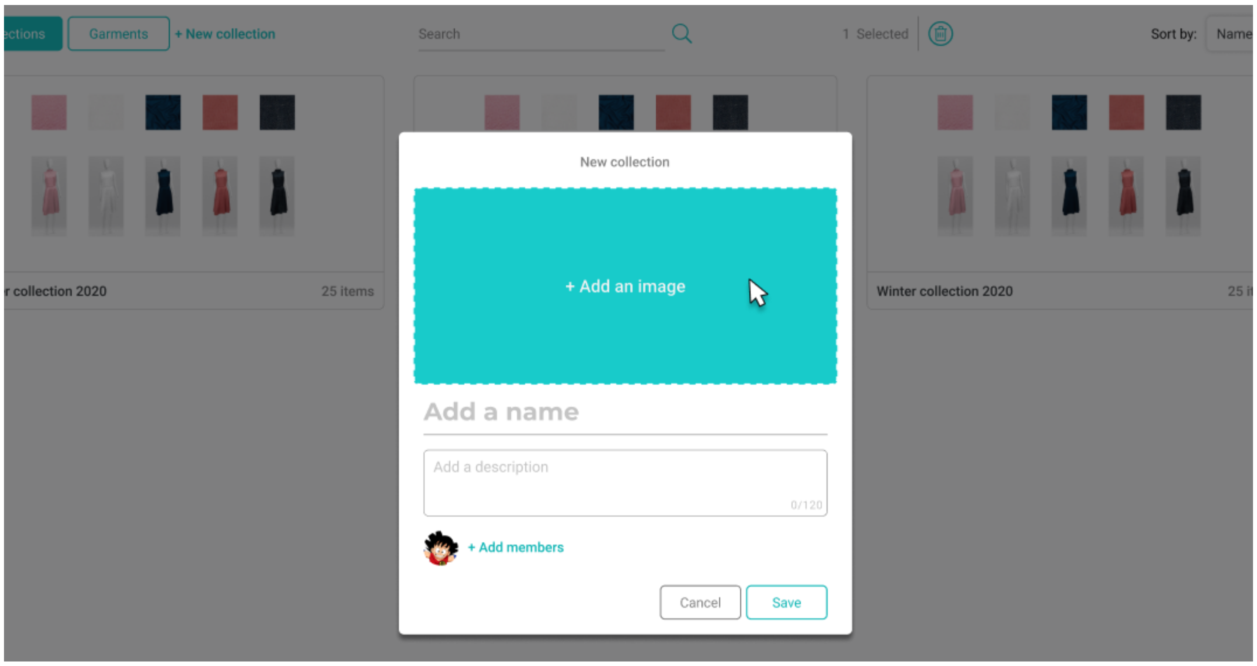
What props each of the components are receiving?
How did you approach the composition of them together?

You are free to use either JavaScript ES6+ or TypeScript.



6. React Components & UI Reusability (no coding required)

Now we have a different composition that our application needs to provide.

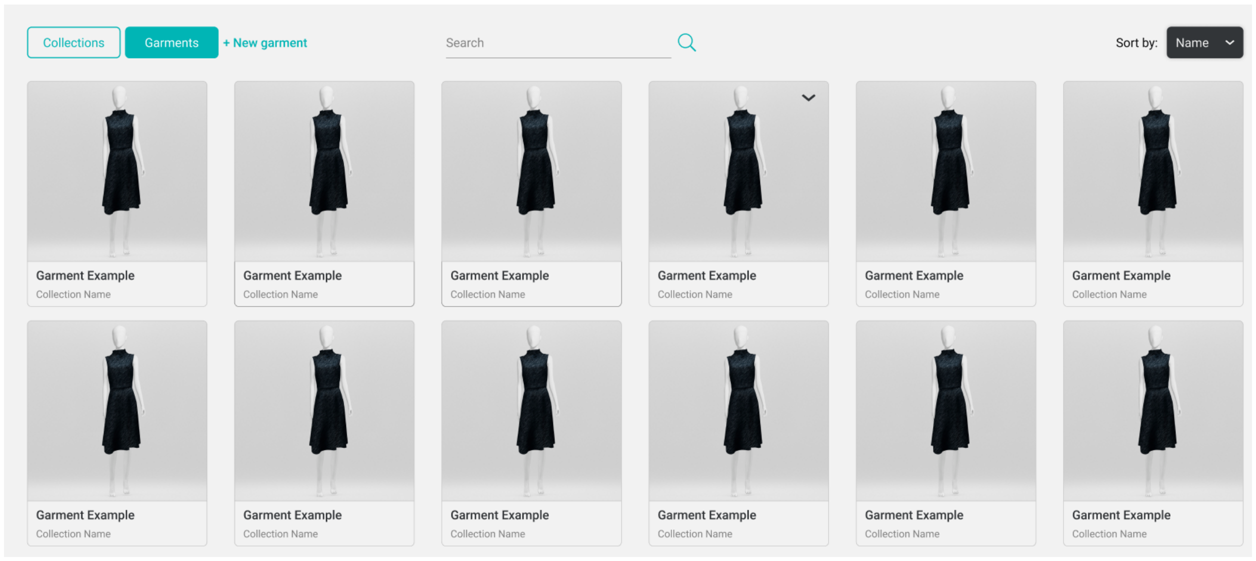


How much code from the previous answer would you be able to reuse when implementing the new requirements?

Describe the hierarchy of components in this new dialog and identify what components are new and what components will require to be implemented.

7. Listings and Pagination (no coding required)

In our application, we need to provide responsive listings to visualize all available resources. We need to provide an infinite scrolling listing that requests progressive chunks and aggregates them as the user scrolls instead of bringing them all at once.



Requirements include the possibility of retrieving the listings sorted by different properties like name, creation date, or last modification date.

Explain your solution to implement an infinite scrolling component. Explain some details about the interface of the API endpoints.

Do you foresee any problems with your solution when dealing with different screen sizes or screen aspect ratios?