BITS in Motion – AR

In this procedure, you can export their finished Blender/Unity model to a lightweight GLB / USDZ format and embed it into a single-page WebAR viewer powered by Google's <model-viewer> component. Using nothing more than a static web host and QR code, you create an experience where you can **1)** scan the code, **2)** choose "View in AR" or "View in 3-D," and **3)** drop the perceptual-shift BITS logo into your own environment—no app installation required. The exercise covers mobile-optimized mesh preparation, cross-platform AR links (Scene Viewer for Android, Quick Look for iOS), and best-practice UX (instruction overlay, fallback desktop viewer).

The procedure

1. Design your 3D model
    a. Create the BITS logo in Blender.
2. After finishing your design, export the file as .glb file
    a. File —> Export —> gITF binary 2.0 (.glb)
3. Convert .glb file to .usdz file for the ios devices.
    a. Use online convertor or Apple`s Reality Convertor(macOS)
4. Create a Web folder
    a. There would be one html file, .glb file and .usdz file in the same folder.
5. Write the html viewer
    a. There would be two different options to see the design (view in 3D, and in AR mode).
    b. See the explanation of the html below
6. Upload all the files into the online host and make the html file as a main file.
7. Generate the QR code by using QR generator from the University.
8. Now, you will have QR code for the 3D and AR version of the model.

HTML code explanation

- **Document and Head set up**
  - Declares HTML-5 page - <!DOCTYPE html>
  - <html lang="en"> helps accessibility

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="utf-8" />
 <title>BITS Perceptual-Shift Viewer</title>
```

- **Load <model-viewer> and Mobile Viewport**
  - Imports Google`s *<model-viewer>* the engine that renders 3D and trigger the AR on phones.
  - *viewport* tag makes the page fill the entire phone screen.

```
 <!-- Google <model-viewer> web component -->
 <script type="module"
       src="https://unpkg.com/@google/model-viewer/dist/model-viewer.min.js"></script>
 <meta name="viewport" content="width=device-width, initial-scale=1.0" />
 <style>
  html,body{margin:0;height:100%;font-family:sans-serif;overflow:hidden;}
  model-viewer{width:100%;height:100%;}
  /* Overlay styling */
  #overlay{
   position:absolute;inset:0;display:flex;
   flex-direction:column;justify-content:center;align-items:center;
   gap:1rem;padding:0 2rem;text-align:center;
   background:rgba(0,0,0,.65);color:#fff;
```

```
  }
  #overlay h1{font-size:1.25rem;margin:0 0 .5rem;}
  #overlay button{
    padding:.8rem 1.4rem;border:none;border-radius:6px;
    font-size:1rem;cursor:pointer;font-weight:600;
  }
  #arBtn{background:#ff9a00;color:#000;}
  #viewBtn{background:#ffffff;color:#000;}
 </style>
</head>
```

- **3D and AR viewer**
  - Loads *bits.glb* on Android/desktop + *bits.usdz* on iOS
  - *ar* + *ar-modes* = one-tap launch into Scene Viewer (Android) or Quick Look (iOS)
  - *camera-controls, auto-rotate* = smooth fallback viewer if user picks 3D only.

```
<body>
 <!-- 3-D model -->
 <model-viewer id="bitsAR"
               src="bits.glb"
               ios-src="bits.usdz"
               ar
               ar-modes="scene-viewer quick-look webxr"
               camera-controls
               auto-rotate
               rotation-per-second="15deg"
               alt="Perceptual-shift BITS logo">
 </model-viewer>
```

- **Choice Overlay**
    - Front-and-Center instructions with two clear CTAs
    - Overlay hides once the user chooses either option.

```
<!-- Choice overlay -->

<div id="overlay">

  <h1>Explore the floating BITS logo</h1>

  <button id="arBtn">📱 View in AR</button>

  <button id="viewBtn">👀 View in 3-D</button>

</div>
```

- **Interaction Script**
    - *activateAR()* calls the phone`s native AR viewer.
    - Overlay is hidden once a choice is made
    - *exit-ar* listener (Android/WebXR) restores the overlay so users can relaunch AR

```
<script>

  const viewer   = document.getElementById('bitsAR');

  const overlay  = document.getElementById('overlay');

  const arBtn    = document.getElementById('arBtn');

  const viewBtn  = document.getElementById('viewBtn');



  // --- OPTION 1: Launch AR ---

  arBtn.addEventListener('click', () => {

    viewer.activateAR();          // open AR session

    overlay.style.display = 'none';

  });

  // --- OPTION 2: Stay in 3-D viewer ---

  viewBtn.addEventListener('click', () => {

    overlay.style.display = 'none';
```

```
    });
    // Bring overlay back if user exits AR (Android/WebXR)
    viewer.addEventListener('exit-ar', () => {
      overlay.style.display = 'flex';
    });
  </script>
</body>
</html>
```