Project Development Phase

Code-Layout reusability, readability

Team ID	NM2023TMID06140
Project Name	Creating an sponsored post for Instagram

To ensure code reusability and readability in an Instagram Reel application, you can follow modular programming principles, use meaningful variable and function names, and organize your code into separate modules. Below is an example of how you can structure your code for an Instagram Reel application using HTML, CSS, and JavaScript while focusing on reusability and readability:

HTML (index.html):

```
<!DOCTYPE html>
<html lang="en">


<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="styles.css">
    <title>Instagram Reel</title>
</head>


<body>
    <div class="reel-container" id="reelContainer">
```

```
        <!-- Reel content goes here -->
    </div>
    <script src="script.js"></script>
</body>

</html>
```

CSS (styles.css):

```
body {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    margin: 0;
    background-color: #f0f0f0;
}
.reel-container {
    display: flex;
    flex-wrap: wrap;
    justify-content: center;
}
.reel-item {
    margin: 10px;
    padding: 10px;
    background-color: #fff;

    border: 1px solid #ccc;

    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

    text-align: center;

}


// Utility function to create a reel item

function createReelItem(videoUrl, caption) {

    const reelItem = document.createElement("div");

    reelItem.classList.add("reel-item");


    const videoElement = document.createElement("video");

    videoElement.src = videoUrl;

    videoElement.controls = true;

    reelItem.appendChild(videoElement);


    const captionElement = document.createElement("p");

    captionElement.textContent = caption;

    reelItem.appendChild(captionElement);

    return reelItem;

}


// Function to add a new reel item to the container
```

```
function addReelItem(videoUrl, caption) {
    const reelContainer = document.getElementById("reelContainer");
    const newReelItem = createReelItem(videoUrl, caption);
    reelContainer.appendChild(newReelItem);
}

// Example usage
const videos = [
    {
        url: "https://example.com/reel1.mp4",
        caption: "Amazing Reel 1"
    },
    {
        url: "https://example.com/reel2.mp4",
        caption: "Awesome Reel 2"
    }
];
// Adding reel items dynamically
videos.forEach(video => {
    addReelItem(video.url, video.caption);
});
```

In this code structure:

The createReelItem function creates a new reel item with a video player and a caption. This function is reusable and can be used to create multiple reel items.

The addReelItem function adds a new reel item to the container. It takes a video URL and a caption as parameters, making it easy to add new items with different content.

The example usage section demonstrates how you can use the addReelItem function to add multiple reel items dynamically by iterating over an array of video objects.

By following this structure, your code remains modular, making it easier to maintain and extend in the future. Variable and function names are descriptive, enhancing readability, and the logic is organized into functions, promoting reusability.