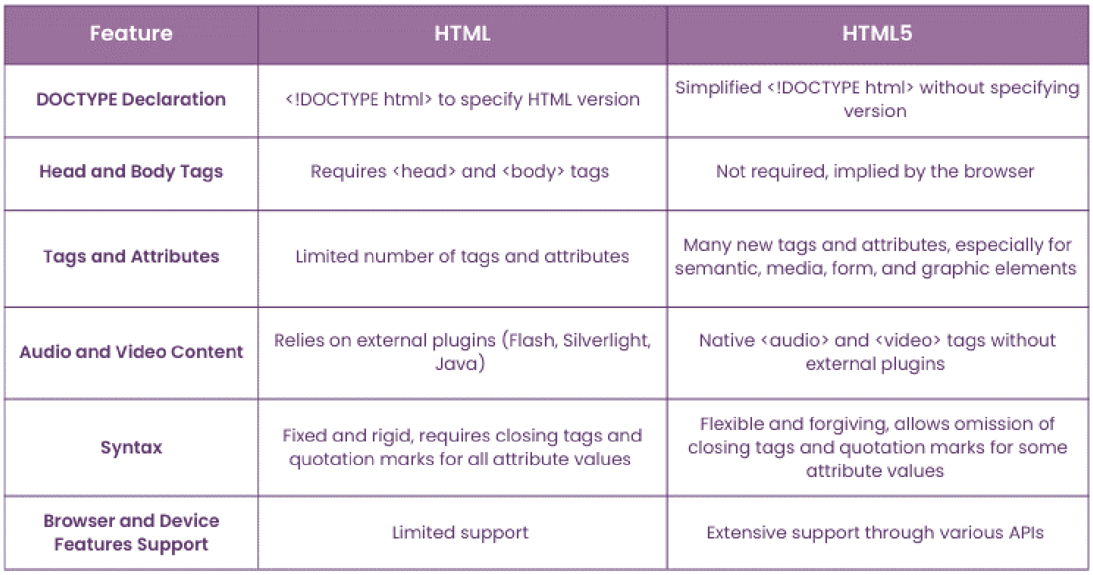
Module-5 Mernstack – HTML5

Theory Assignment:

Question 1: Difference b/w HTML & HTML5?



Feature	HTML	HTML5
DOCTYPE Declaration	<!DOCTYPE html> to specify HTML version	Simplified <!DOCTYPE html> without specifying version
Head and Body Tags	Requires <head> and <body> tags	Not required, implied by the browser
Tags and Attributes	Limited number of tags and attributes	Many new tags and attributes, especially for semantic, media, form, and graphic elements
Audio and Video Content	Relies on external plugins (Flash, Silverlight, Java)	Native <audio> and <video> tags without external plugins
Syntax	Fixed and rigid, requires closing tags and quotation marks for all attribute values	Flexible and forgiving, allows omission of closing tags and quotation marks for some attribute values
Browser and Device Features Support	Limited support	Extensive support through various APIs

Benefits of HTML5 vs HTML

HTML5 introduces certain new elements and attributes that enhance the way we handle media content, making it a significant upgrade over traditional HTML. Here are some key benefits:

a) **Multimedia Integration:** HTML5 includes native support for audio and video elements, allowing for seamless embedding of multimedia content without the requirement for external plugins like Flash.

b) **Improved Semantics:** New semantic elements such as article, section, and nav provide better structure and meaning to web pages, improving accessibility and Search Engine Optimisation (SEO).

c) Enhanced Graphics: The canvas element in HTML5 enables dynamic, scriptable rendering of 2D shapes and images, which is ideal for creating interactive graphics and animations.

d) Offline Capabilities: HTML5's application cache and local storage features allow web applications to function offline, providing a better user experience even without an internet connection.

e) Mobile-friendly: HTML5 is designed with mobile devices in mind, offering features like responsive design support and touch event handling, which are crucial for modern Web Development.

Question 2: What are the additional tags used in HTML5?

1. Semantic Elements:

- **`<header>`**: Represents introductory content or navigational links.
- **`<nav>`**: Represents a section of navigation links.
- **`<section>`**: Defines a section in a document, used for thematic grouping of content.
- **`<article>`**: Represents self-contained content that can be independently distributed (e.g., blog posts).
- **`<aside>`**: Represents content tangentially related to the content around it, often used for sidebars.
- **`<footer>`**: Represents footer content, often containing information about the author or related documents.
- **`<main>`**: Represents the main content of the document, unique and central to the document.
- **`<figure>`**: Represents self-contained content, often with a caption (`<figcaption>`).
- **`<figcaption>`**: Provides a caption for the `<figure>` element.

2. Multimedia Elements:

- **`<audio>`**: Used for embedding audio content.
- **`<video>`**: Used for embedding video content.
  `<source>`: Defines multiple media resources for `<audio>` and `<video>`.
- **`<track>`**: Provides text tracks (subtitles, captions) for `<video>` and `<audio>`.

3. Graphics and Interactive Elements: ▯ **`<canvas>`**: Used for drawing graphics on the fly via scripting (usually JavaScript).

- **`<svg>`**: Supports Scalable Vector Graphics.
- **`<details>`**: Used as a disclosure widget that shows or hides additional content.
- **`<summary>`**: Provides a summary, legend, or caption for the `<details>` element.

4. Form Elements: ▯ **`<datalist>`**: Provides an autocomplete feature for `<input>` elements.

- **`<keygen>`**: Generates a key-pair and a certificate request (now deprecated in favor of Web Crypto API).
- **`<output>`**: Represents the result of a calculation or user action. ▯ **`<progress>`**: Represents the completion progress of a task.
- **`<meter>`**: Represents a scalar measurement within a known range, or a fractional value.

5. Other Elements:

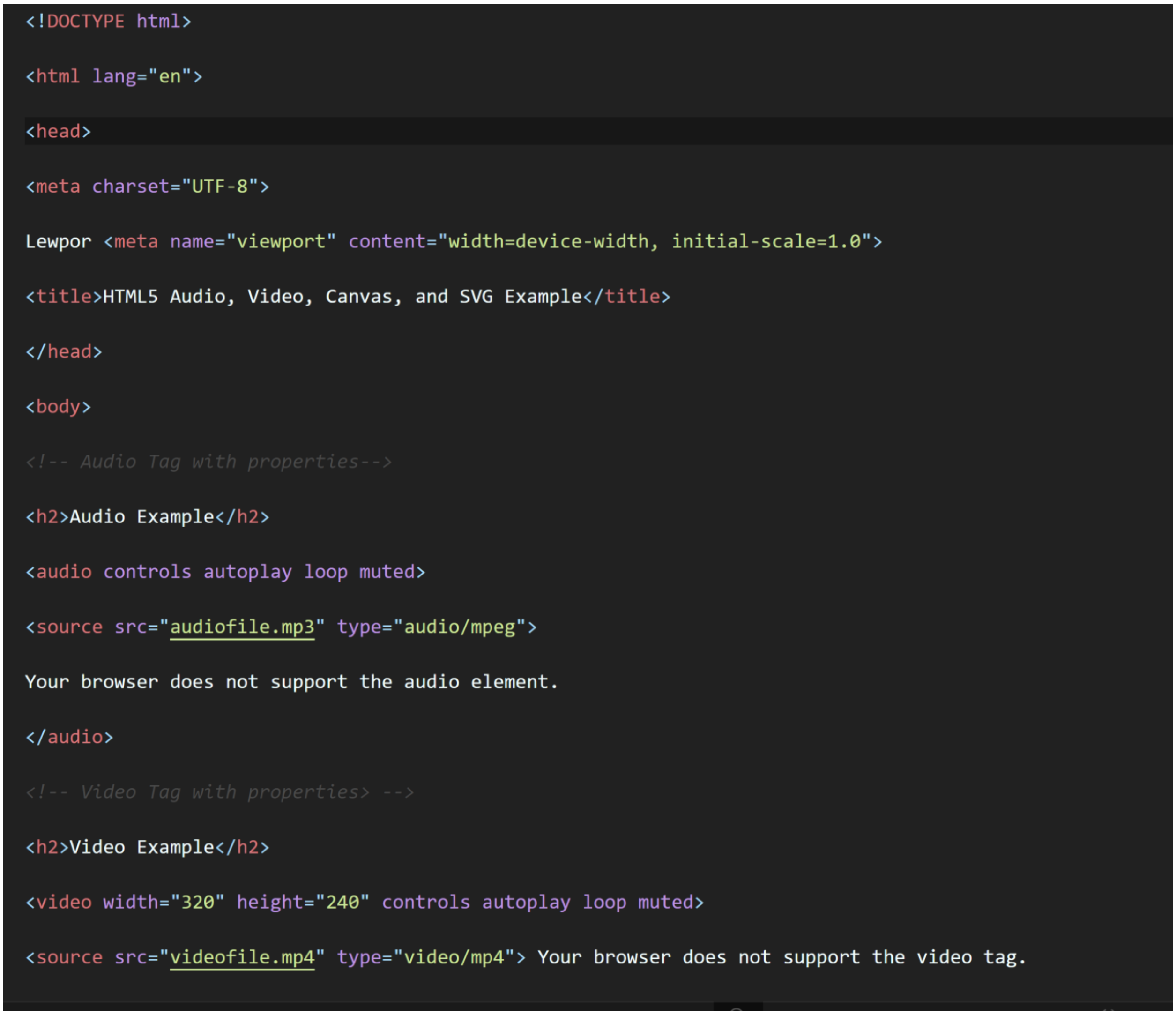
- **`<mark>`**: Highlights text for reference or importance.
- **`<time>`**: Represents a specific point in time or duration.

- **`<wbr>`**: Suggests a word break opportunity.
- **`<template>`**: Defines a template for client-side content that is not rendered immediately.
- **`<dialog>`**: Represents a dialog box or interactive component, such as a popup.

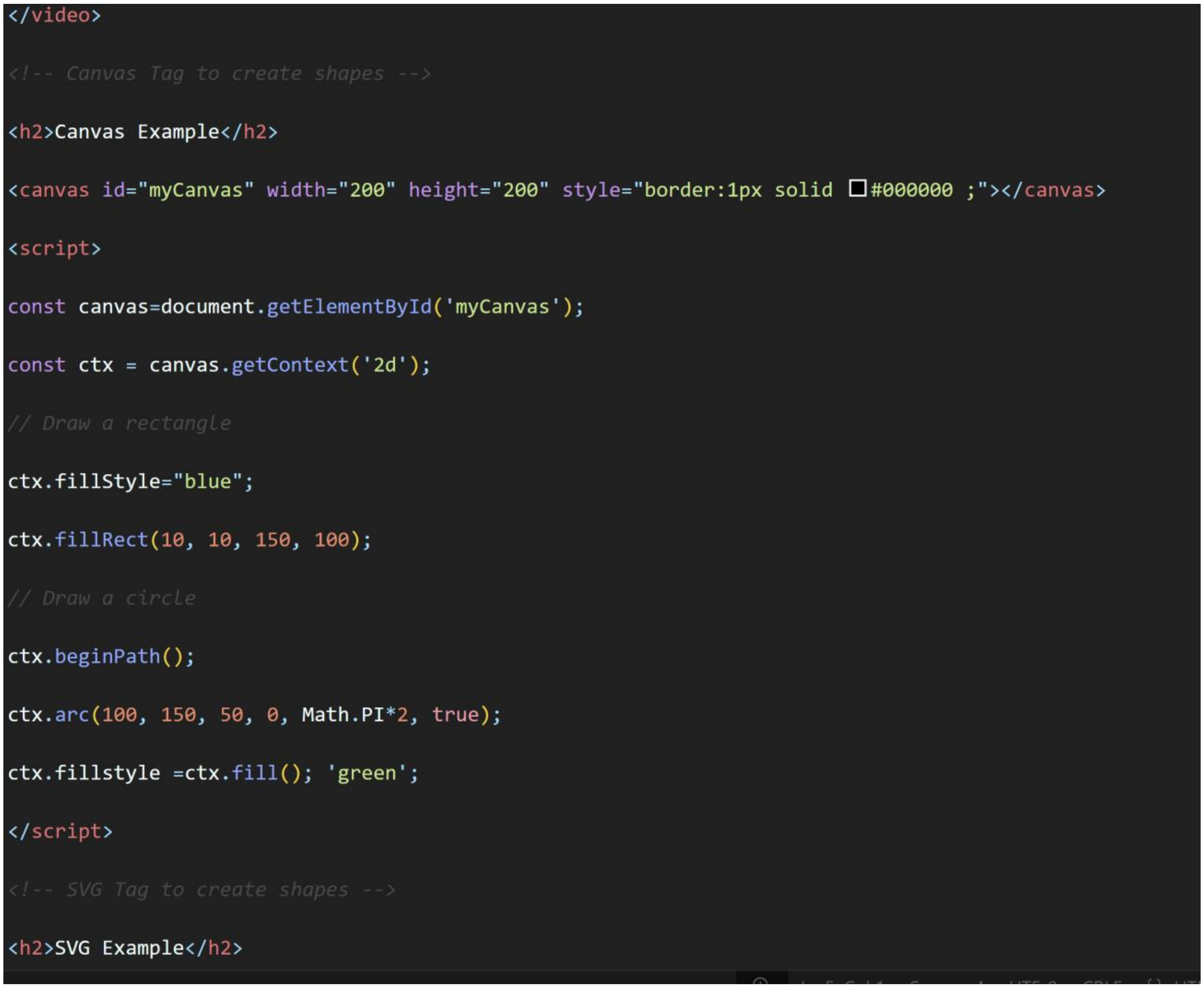
Lab Assignment:

Task:

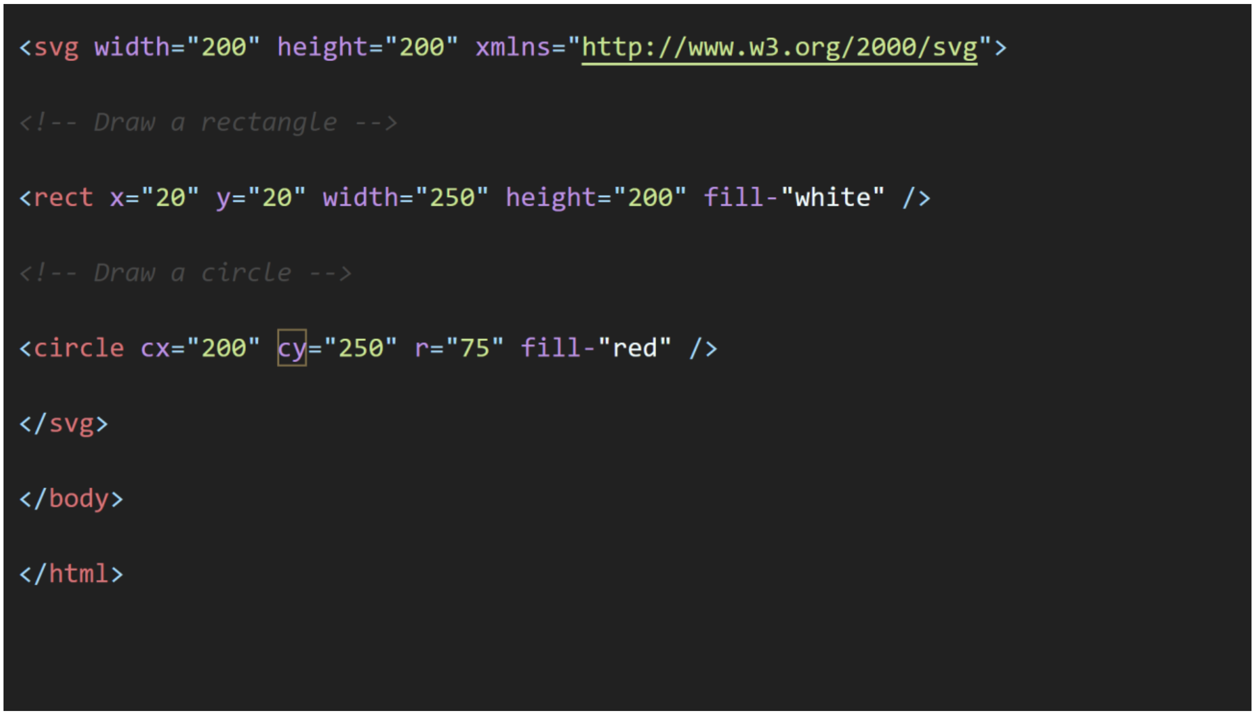
- Create a audio video tag
- Also applied properties like muted loop autoplay
- Create some shape using canvas tag in html
- Create some shape using svg tag in html



```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

Lewpor <meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>HTML5 Audio, Video, Canvas, and SVG Example</title>

</head>

<body>

<!-- Audio Tag with properties-->

<h2>Audio Example</h2>

<audio controls autoplay loop muted>

<source src="audiofile.mp3" type="audio/mpeg">

Your browser does not support the audio element.

</audio>

<!-- Video Tag with properties> -->

<h2>Video Example</h2>

<video width="320" height="240" controls autoplay loop muted>

<source src="videofile.mp4" type="video/mp4"> Your browser does not support the video tag.
```



```
</video>

<!-- Canvas Tag to create shapes -->

<h2>Canvas Example</h2>

<canvas id="myCanvas" width="200" height="200" style="border:1px solid #000000 ;"></canvas>

<script>

const canvas=document.getElementById('myCanvas');

const ctx = canvas.getContext('2d');

// Draw a rectangle

ctx.fillStyle="blue";

ctx.fillRect(10, 10, 150, 100);

// Draw a circle

ctx.beginPath();

ctx.arc(100, 150, 50, 0, Math.PI*2, true);

ctx.fillstyle =ctx.fill(); 'green';

</script>

<!-- SVG Tag to create shapes -->

<h2>SVG Example</h2>
```



```
<svg width="200" height="200" xmlns="http://www.w3.org/2000/svg">

<!-- Draw a rectangle -->

<rect x="20" y="20" width="250" height="200" fill-"white" />

<!-- Draw a circle -->

<circle cx="200" cy="250" r="75" fill-"red" />

</svg>

</body>

</html>
```