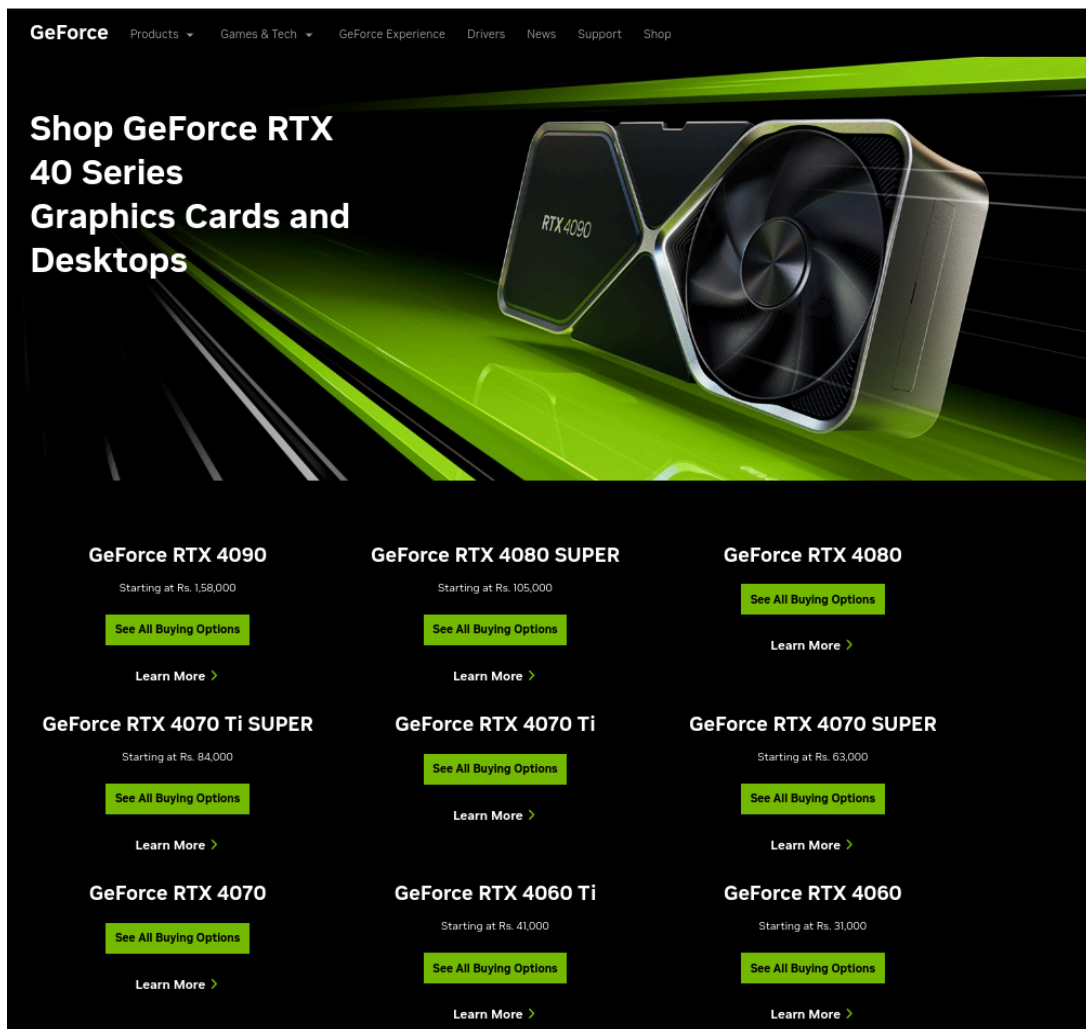


CC Backend - Task 2

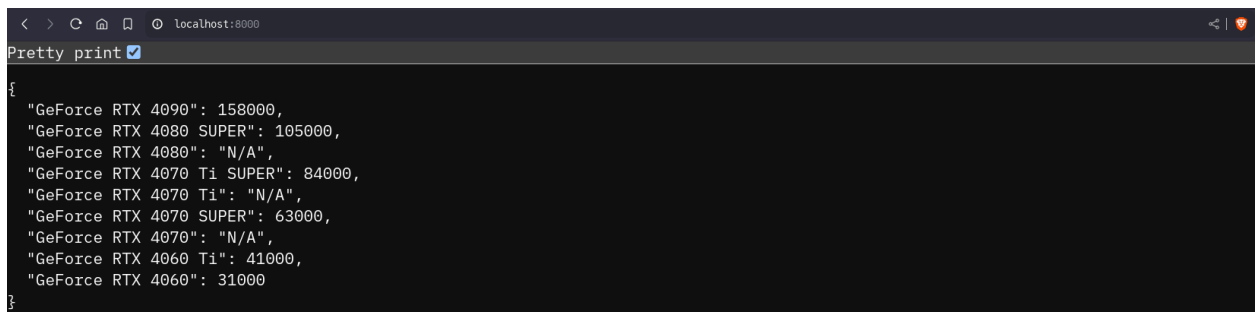
The objective of this task is to scrape data from a website that does not provide APIs for the same. You must then serve that data as JSON from a locally hosted server that will be read by a provided HTML file for verification.

The website you will be obtaining data from is here - <https://www.nvidia.com/en-in/geforce/buy/>. The website contains information about GPUs and their prices from the popular american MNC Nvidia Corporation.



Unlike in the previous task, many web-services do NOT provide APIs for convenient use, especially when the data is proprietary. In such cases, an (usually) illegal way of obtaining data is through the use of automated software to scrape their respective websites and collect information from them.

Additionally, you have to create a (very simple) server to serve up the collected data. By doing so, you will effectively have created an API of your own, and you can access the information exactly like you did in the first task. If you visit your endpoint, you should be able to see JSON data:

A screenshot of a web browser window with the address bar showing 'localhost:8000'. The page content displays a JSON object with various GeForce RTX model names and their corresponding prices. The JSON is formatted with syntax highlighting. A 'Pretty print' button is visible in the top left corner of the browser window.

```
{
  "GeForce RTX 4090": 158000,
  "GeForce RTX 4080 SUPER": 105000,
  "GeForce RTX 4080": "N/A",
  "GeForce RTX 4070 Ti SUPER": 84000,
  "GeForce RTX 4070 Ti": "N/A",
  "GeForce RTX 4070 SUPER": 63000,
  "GeForce RTX 4070": "N/A",
  "GeForce RTX 4060 Ti": 41000,
  "GeForce RTX 4060": 31000
}
```

Fig (2)

A file will be provided on the WhatsApp group (source code at the end of the document). This is a simple HTML file that does the job of making a request to your server, and displays the result. Use this to verify your results:

A screenshot of a web browser window showing the title 'JSON Data from http://localhost:8000'. The main content area displays the same JSON object as in Figure 2, with syntax highlighting.

```
{
  "GeForce RTX 4090": 158000,
  "GeForce RTX 4080 SUPER": 105000,
  "GeForce RTX 4080": "N/A",
  "GeForce RTX 4070 Ti SUPER": 84000,
  "GeForce RTX 4070 Ti": "N/A",
  "GeForce RTX 4070 SUPER": 63000,
  "GeForce RTX 4070": "N/A",
  "GeForce RTX 4060 Ti": 41000,
  "GeForce RTX 4060": 31000
}
```

Fig (3)

For the completion of this task, it is recommended that you use either the [Django REST framework](#), or [FastAPI](#) (python) / [Express.js](#) (JS), although you are free to use any other web framework as well.

Rules:

- (a) This task should be completed in either Python or JavaScript. Scraping information from these websites should be done using [Selenium](#) (respective wrappers for Python and JS). You will need to install a driver for the browser you use (Chromedriver for Chromium based browsers, GeckoDriver for Firefox, MS Edge WebDriver for MS Edge, etc).
- (b) The server should serve up a map of the GPU names and their prices (N/A if the price is not listed), as shown in Fig (2). The server must NOT be hosted on another provider like Vercel or Render, but should only be made available on localhost for the verification process.
- (c) The only modification to the HTML file allowed is to change the value of the constant 'URL' in the code for the script. Change the port (currently at 8000) to where your server is hosted, and specify the route (if needed). If successful, the website will display information from your API. You can use the console to debug any errors during the fetch process.
- (d) You **cannot** resolve errors using any middleware provided by the library you are using to create the server. Find other ways to do it.

Note:

- (a) Submit the 'index.html' file along with the rest of the code in your final submission. Any modifications to the code apart from the URL will invalidate your submission. In the project's README, attach a picture of the website displaying the information (like Fig (3))
- (b) To display the webpage, you can download the index.html file, and then visit this URL on your browser:
file:///absolute path to the file>. Alternatively, you can use a third party package for a HTTP server and host it on a different port simultaneously with your server.

HTML Source code:

```

<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>CC-Test</title>
    <style>
      body {
        margin: 0;
        padding: 20px;
        background-color: #f0f0f0;
        color: #333;
      }
      h1 {
        color: #0056b3;
        text-align: center;
      }
      pre {
        background-color: #fff;
        padding: 20px;
        border: 1px solid #ccc;
        border-radius: 8px;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        overflow: auto;
        max-height: 70vh;
      }
    </style>
  </head>
  <body>
    <h1 id="title">JSON Data from Localhost</h1>
    <pre id="jsonData">Loading...</pre>

    <script>
      const URL = "http://localhost:8000"; // CHANGE THIS URL TO THAT OF THE SERVER
      document.getElementById("title").textContent = "JSON Data from " + URL;
      fetch(URL)
        .then((response) => {
          if (!response.ok) {
            throw new Error("Network response was not ok " + response.statusText);
          }
          return response.json();
        })
        .then((data) => {
          document.getElementById("jsonData").textContent = JSON.stringify(data, null, 2);
        })
        .catch((error) => {
          document.getElementById("jsonData").textContent = "There has been a problem with trying to
fetch: \n" + error;
        });
    </script>
  </body>
</html>

```