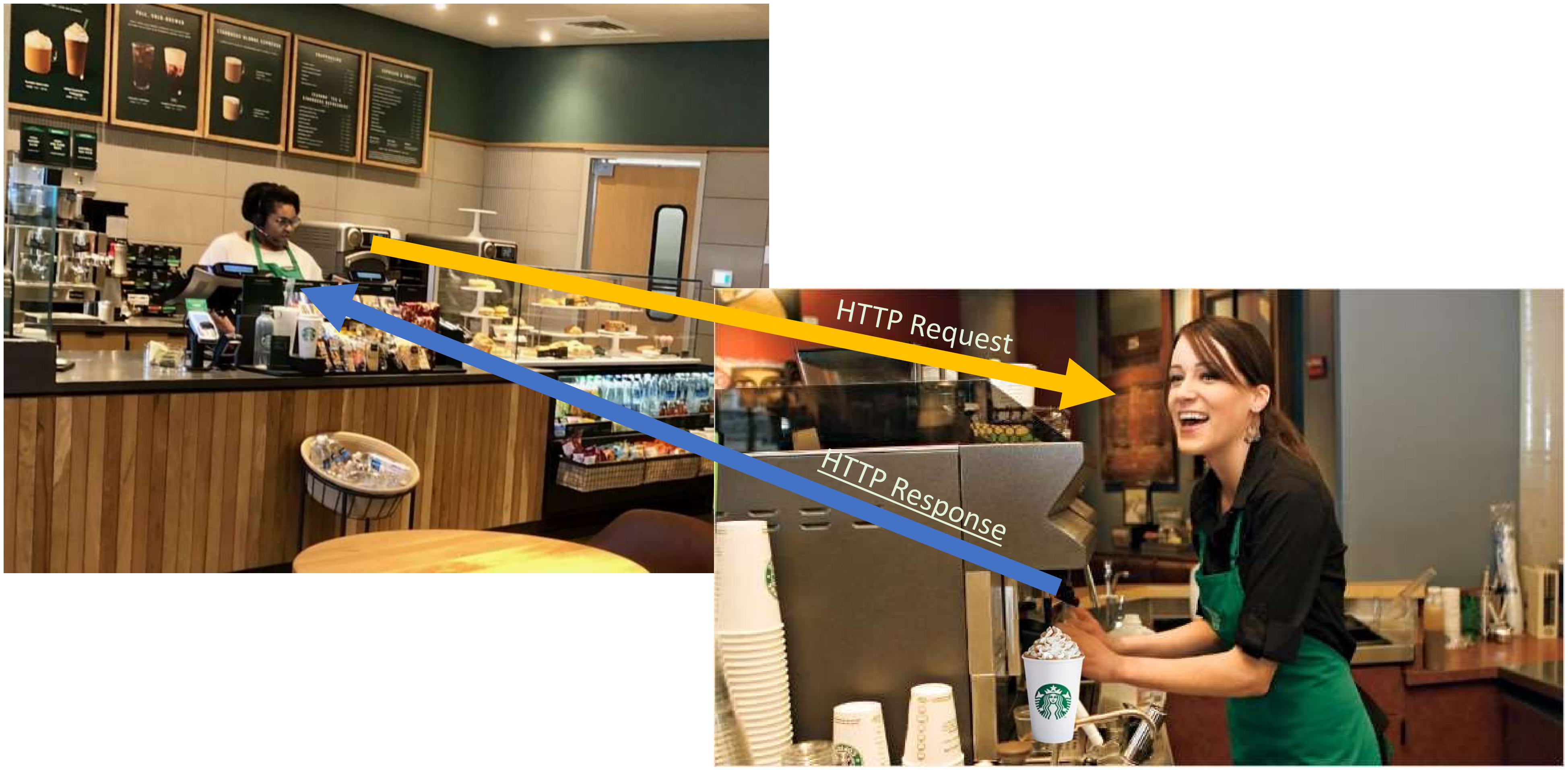
MODULE 3

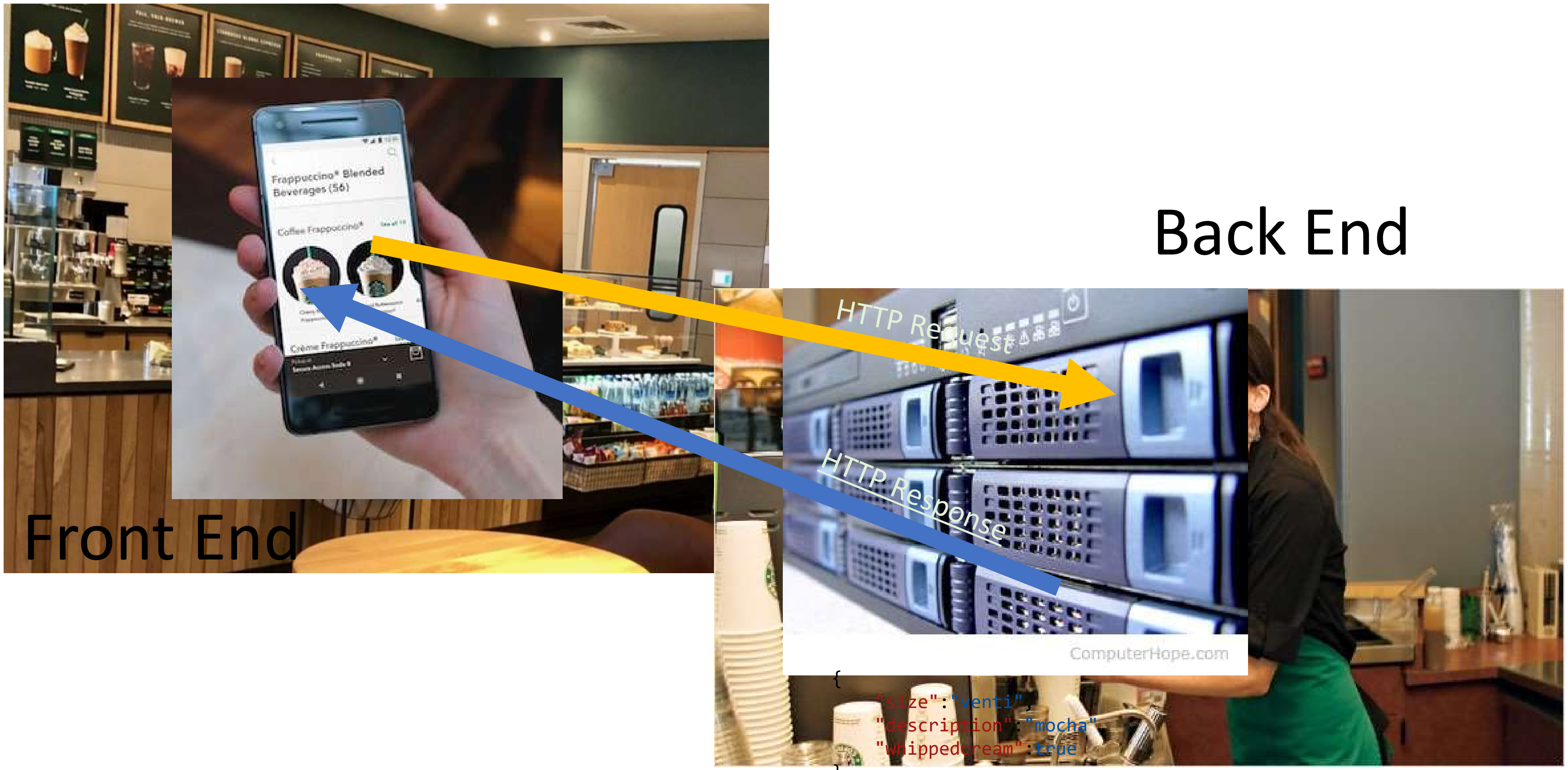
Review – Web Services

Let’s get some coffee!

Coffee process



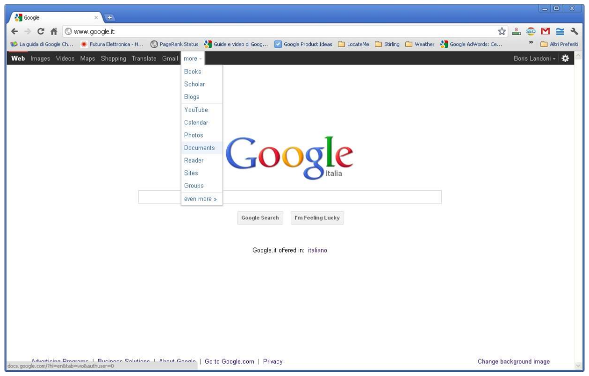
Coffee process

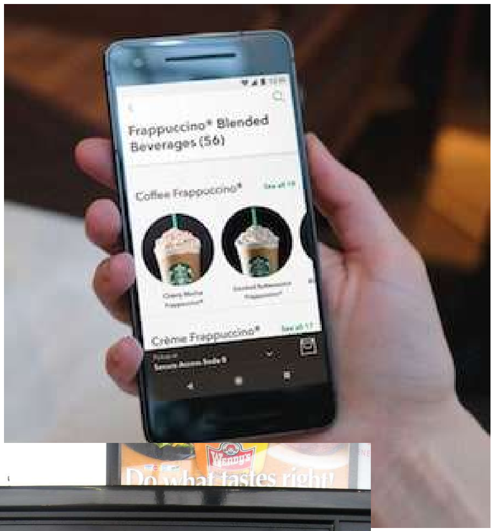


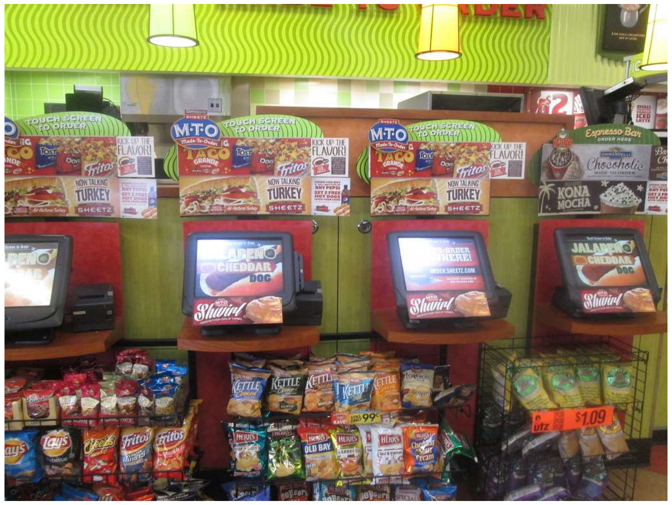
```
{
    "size":"venti",
    "description":"mocha"
    "whippedcream":true
}
```

Front Ends – You’ve used

- Web Browser
- Mobile App
- ATM Machine
- Sheetz MTO Kiosk
- Gas Pump





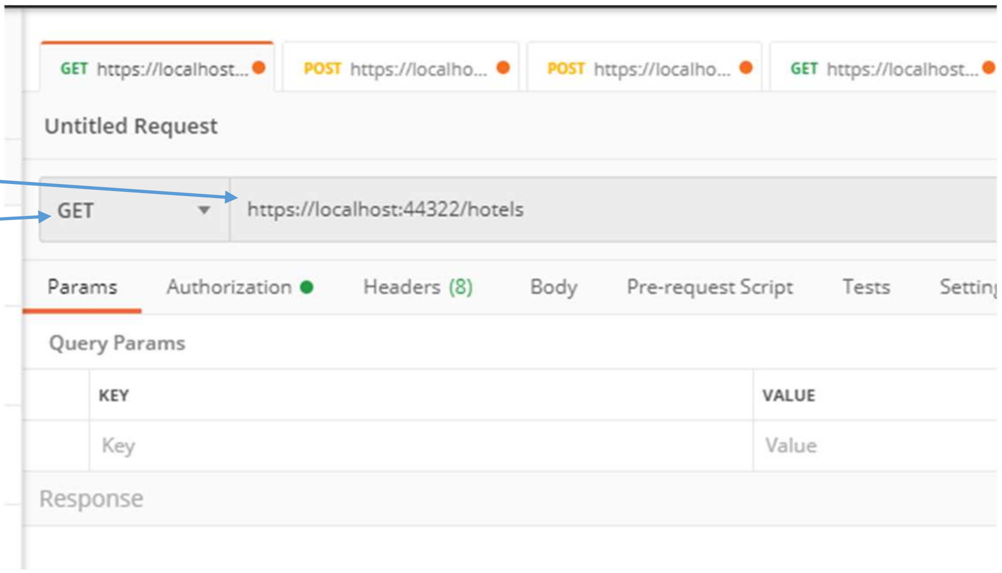


Front End Needs

- Reference to Back End
  - URL (end point)
- Method (or verb)
  - HttpGet – retrieve information
  - HttpPost – add information
  - HttpPut – update information
  - HttpDelete – delete information
- How to talk to the back end
  - Client

Front End Needs – Postman

- Reference to Back End
  - URL (end point)
- Method (or verb)
  - HttpGet – retrieve information
  - HttpPost – add information
  - HttpPut – update information
  - HttpDelete – delete information
- How to talk to the back end
  - Client



Front End Needs – CLI

- Reference to Back End
  - URL (end point)
- Method (or verb)
  - HttpGet – retrieve information
  - HttpPost – add information
  - HttpPut – update information
  - HttpDelete – delete information
- How to talk to the back end
  - Client

```
private readonly string API_URL = "https://localhost:44322/";
private readonly RestClient client = new RestClient();

public List<Hotel> GetHotels()
      {
          RestRequest request = new RestRequest(API_URL + "hotels");
          IRestResponse<List<Hotel>> response = client.Get<List<Hotel>>(request);
          if (response.ResponseStatus != ResponseStatus.Completed)
          {
              Console.WriteLine("Error occurred - unable to reach server.");
          }
          else if (!response.IsSuccessful)
          {
              Console.WriteLine("Error occurred - received non-success response");
          }
          else
          {
              return response.Data;
          }
          return null;
      }
```

Back End Needs – Process the Request

- Instantiate Controller
- Execute Action
- Return Data

Back End Needs

- Instantiate Controller
  - `https://localhost:44322/`**`hotels`**
- Execute Action
  - Determined by the method and the route

  HttpGet -- `https://localhost:44322/hotels`
- Return Data
  - Send back JSON

Back End Needs

- Instantiate Controller
  - `https://localhost:44322/`**`hotels`**
- Execute Action
  - Determined by the method and the route
  - **HttpGet** -- `https://localhost:44322/hotels`
- Return Data
  - Send back JSON

```
[Route("/[controller]")]
  [ApiController]
  public class HotelsController : ControllerBase
  {

     [HttpGet]
     public ActionResult<List<Hotel>> ListHotels()
     {
        return hotelDao.List();
     }
```

What about a specific hotel?

Specific Hotel – CLI

```
private readonly string API_URL = "https://localhost:44322/";
private readonly RestClient client = new RestClient();

public Hotel GetHotel(int hotelId)
      {
          RestRequest request = new RestRequest(API_URL + "hotels");
          request.AddParameter(“id”,hotelId);
          IRestResponse<Hotel> response = client.Get<Hotel>(request);
          if (response.ResponseStatus != ResponseStatus.Completed)
          {
              Console.WriteLine("Error occurred - unable to reach server.");
          }
          else if (!response.IsSuccessful)
          {
              Console.WriteLine("Error occurred - received non-success response”);
          }
          else
          {
              return response.Data;
          }
          return null;
      }
```

Specific Hotel – Back End

```
[Route("/[controller]")]
  [ApiController]
  public class HotelsController : ControllerBase
  {

    [HttpGet]
    public ActionResult<List<Hotel>> ListHotels()
    {
      return hotelDao.List();
    }

    [HttpGet("{id}")]
    public ActionResult<Hotel> GetHotel(int id)
    {
        Hotel hotel = hotelDao.Get(id);

        if (hotel != null)
        {
            return hotel;
        }
        else
        {
            return NotFound();
        }
    }
```

Defines the route:
`https://localhost:44322/hotels/1`

LET’S CODE!

ELEVATE YOURSELF