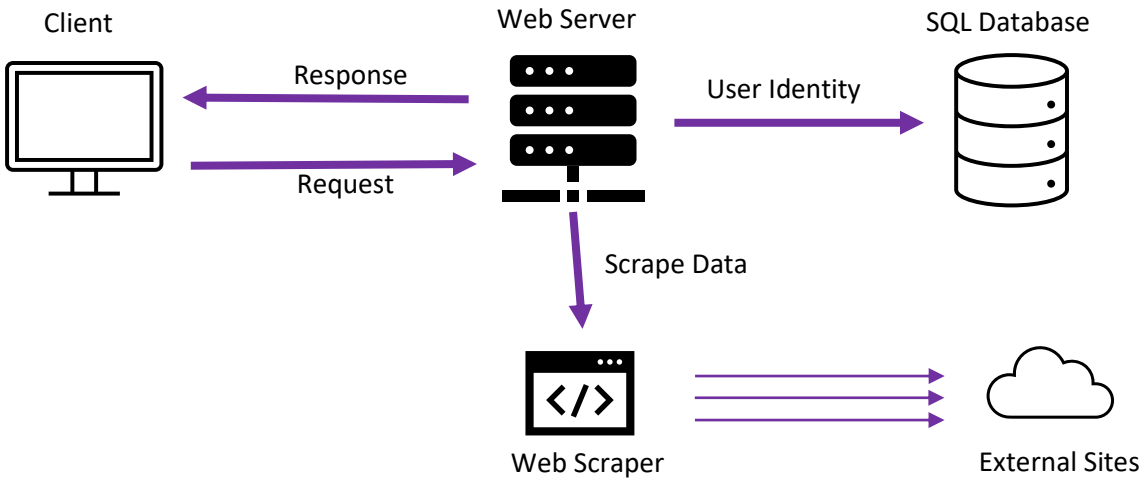
6/7/2020

Team20

Web Scraper System Documentation

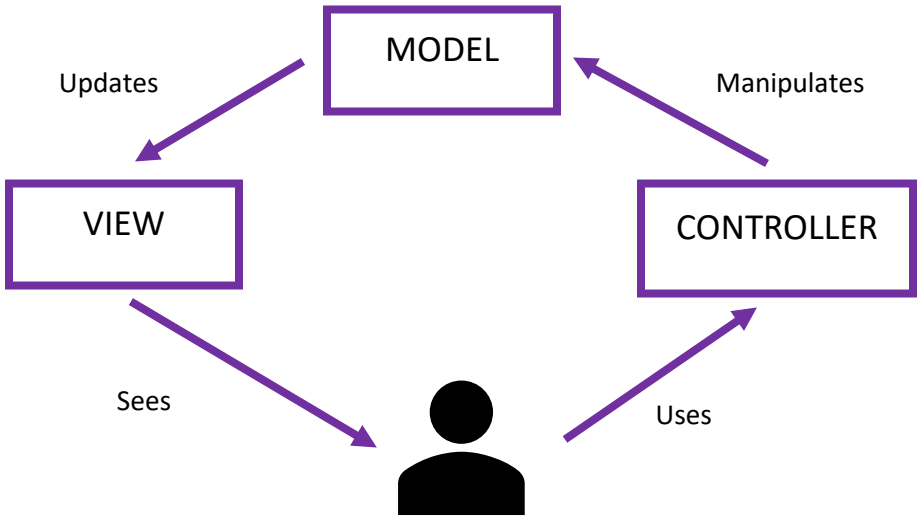
System Architecture Overview:

The Web Scraper application was built using ASP.NET Core 3.1. This framework has built in architecture types that it pushes the developers to implement. Since this is a basic web application, the users request a page, which is then retrieved or rendered by the server, and returned to the user. The application in its current state only makes these simple requests for pages, there is no usage of AJAX or any other client side request model. The user can request a page that lists the products of a certain category. The application receives this request, pulls data from outside websites and categorizes that data within a "Scraper" abstract class. The data is dynamically formatted in an html page and sent as a response to the user. The application is also connected to a SQL database to store and retrieve user login information. This database can be expanded to hold more user data such as search history. Below is a simple diagram outlining the overall system.



Design Patterns:

Within the application, we used a few different design patterns following an object oriented approach. The overall model we used to handle requests was the Model-View-Controller pattern (MVC). In this pattern, a controller class contains methods that handle route endpoints. The controller will interact with the database or other services to create a model. This model is then used to create the view (the html page).



ASP.NET Core is heavily based on Dependency Injection. Outside services can be registered within the Startup.cs file and injected into a controller to handle actions. In our application, we inject the database context and our in memory models in order to manipulate what is returned to the user. These individual dependencies can change in their implementation, and the application itself will be unaffected.

File Structure:

Since this application follows the MVC pattern, its folders also follow this pattern. From the root directory, the Controllers, Models, Views, and ViewModels folders make up the MVC pattern. There is an additional Areas folder, which is created when scaffolding the default user identity system. The Data folder holds all in memory data structures and database interaction classes. The wwwroot folder is an ASP.NET folder that houses all static files, including client side libraries, JavaScript files, icons, and images. There are three other important files in the root structure. Program.cs defines the "main" method and is the entry point for the application. Startup.cs holds all the services and configuration options for the application, including the request pipeline and any middleware added to the program. Finally, appsettings.json is a configuration file that holds connection strings and other user configurations.

Languages/Tooling/Frameworks:

The application is written in C# using ASP.NET Core and .NET Core 3.1. To run the application locally, ASP.NET Core 3.1 and .NET Core 3.1 runtimes will need to be installed on the system. The Scraper class uses the Html Agility Pack library to handle parsing data scraped from external websites. The application is connected to a SQL Server database

Deployment:

The application is hosted on an Azure App Service. Deployment to this service can be done using the Publish feature in Visual Studio or on the command line using the dotnet CLI. In

order for the application to talk to the database, the connection string must be saved as an environment variable on the system, or within the Azure configuration settings.

Code Base:

The source code for the project is currently maintained on GitHub. Any changes can be submitted via a pull request. Acceptance of the changes must be approved by two members of the core team.