Single-page-application

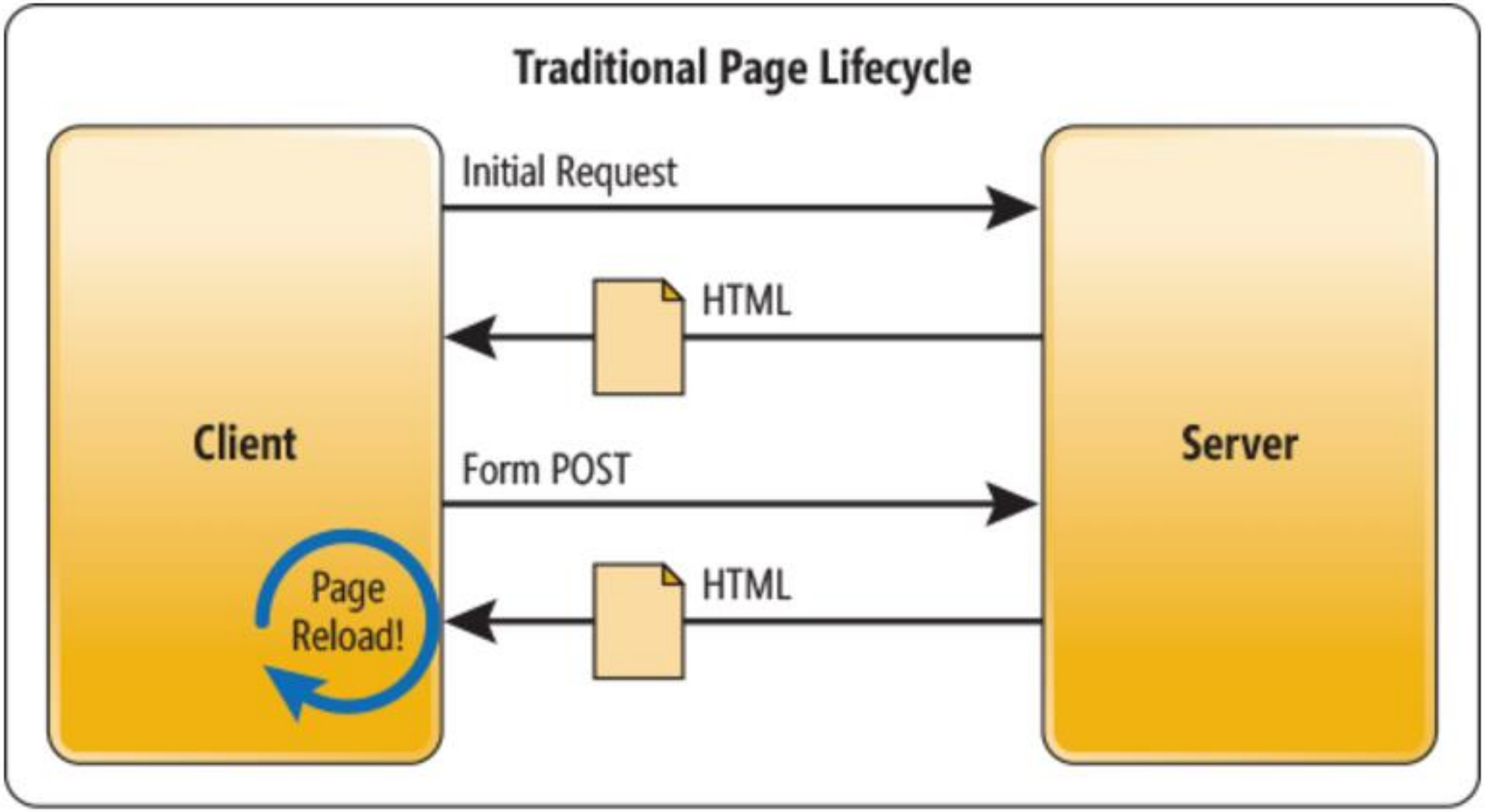
What?

- **Web apps**
  - Load a single HTML page
  - Dynamically update the page as the user interacts with the app

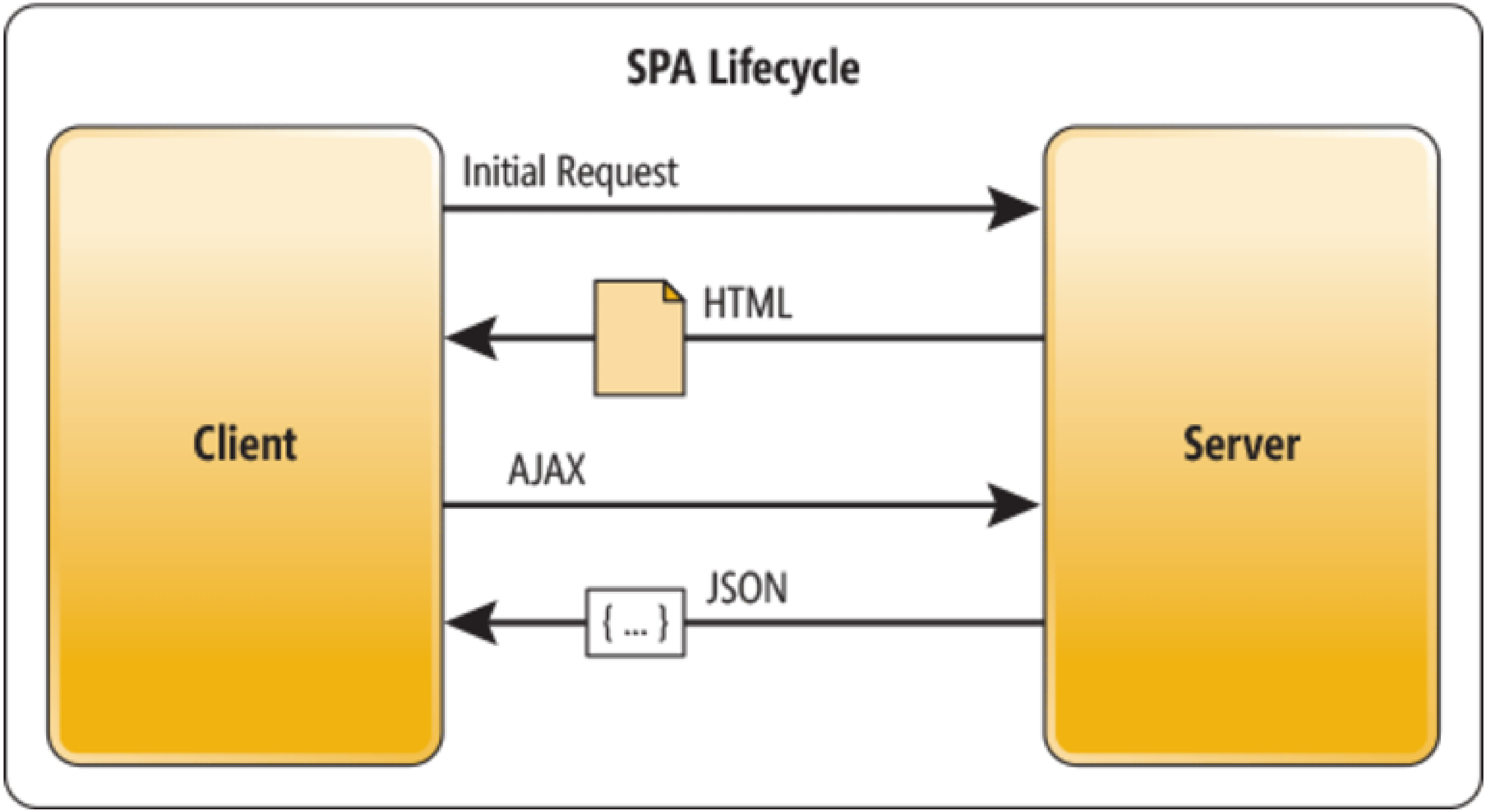
- Use **AJAX and HTML5** to create fluid and responsive Web apps
  - much of work happens on the client side, in JavaScript

- **Eg:** https://angular-university.io/home

Lifecycle


Traditional Page Lifecycle
Client
Initial Request
HTML
Form POST
HTML
Server
Page Reload!

Lifecycle


SPA Lifecycle
Client
Server
Initial Request
HTML
AJAX
JSON
{ ... }

Benefits

- **Applications are more fluid and responsive**
  - No more constant full page reloads
  - No more network back and forth trips to the server to fetch all that HTML

Benefits

- Creates a separation between the **presentation** (HTML markup) and **application logic** (AJAX requests and JSON responses)
  - easier to design and evolve each layer

Benefits

- **Client and service are independent**
  - Replace back-end
  - Replace frond-end

Benefits

- **Client:** super-simple to deploy
  - just one index.html file, with a CSS bundle and a Javascript bundle

- Can be uploaded to any static content server like Apache, Nginx, Amazon S3 or Firebase Hosting

- **Easy to versioning in Production**

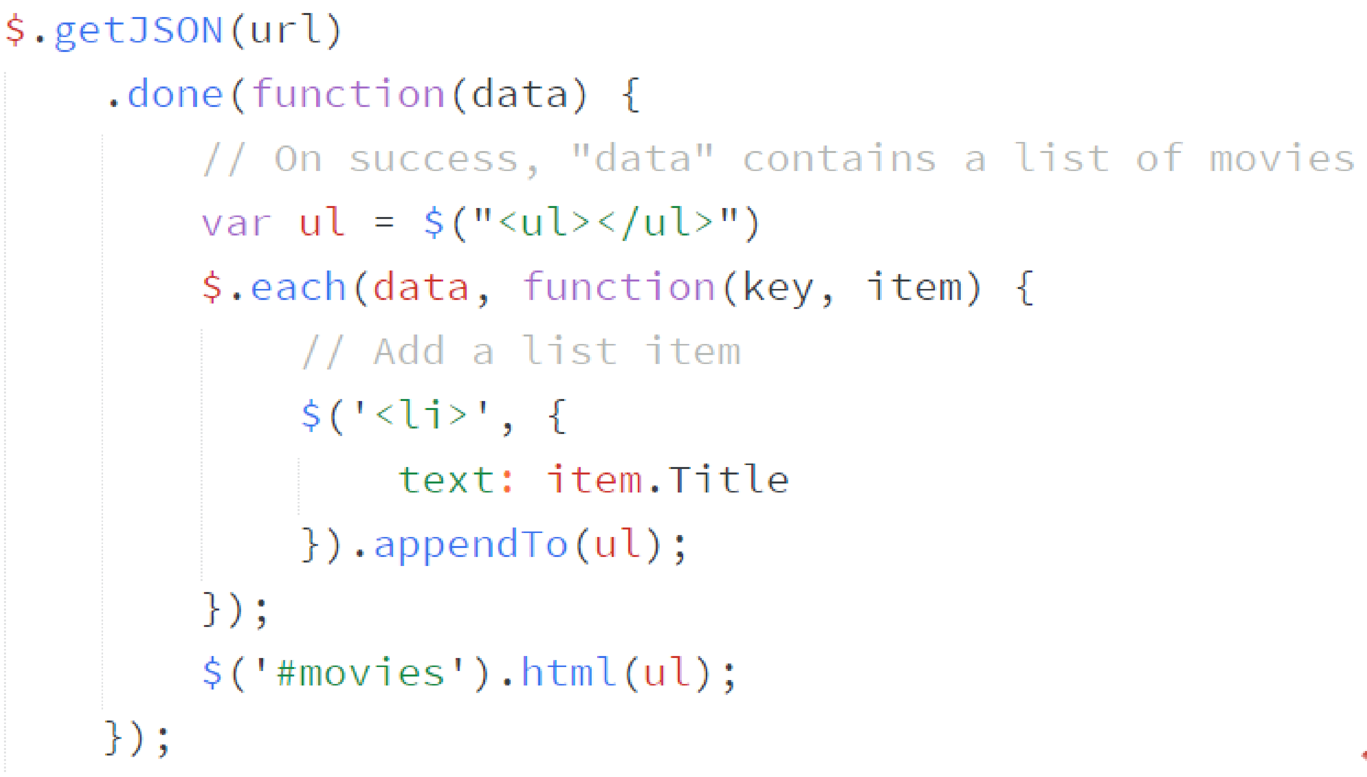
Architecture

- **Service/Backend**
  - RESTful API

- **Client**
  - HTML
  - CSS
  - Javascript
    - Pure JS
    - **Front-end framework**

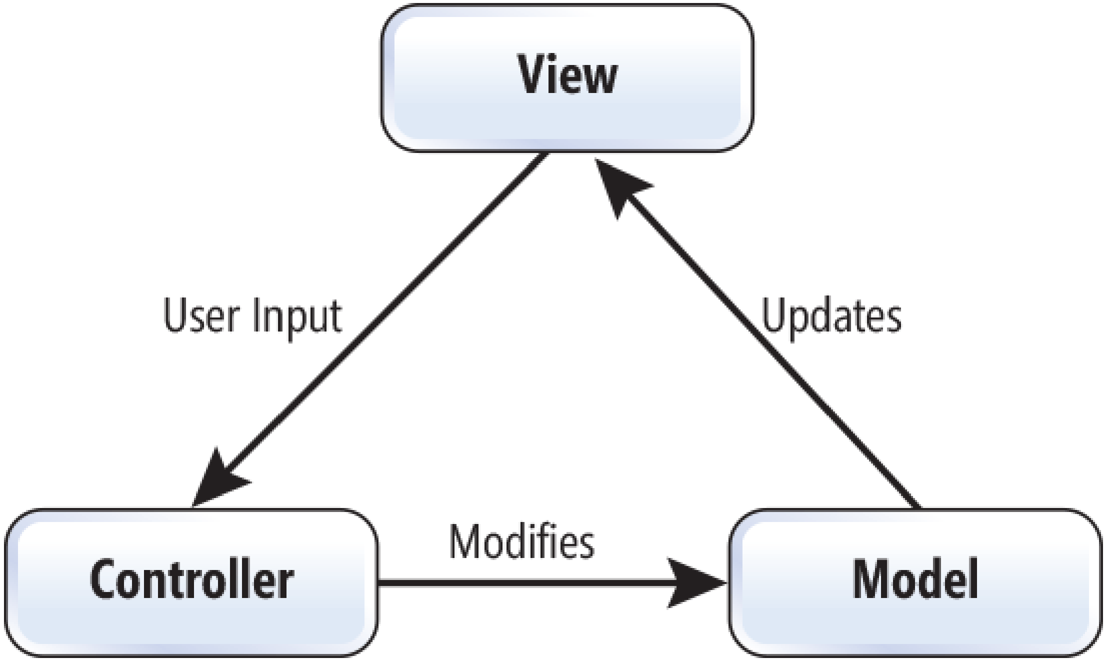
Client

- UI triggers an AJAX request
- Update the HTML to display the response payload
- Handle AJAX errors



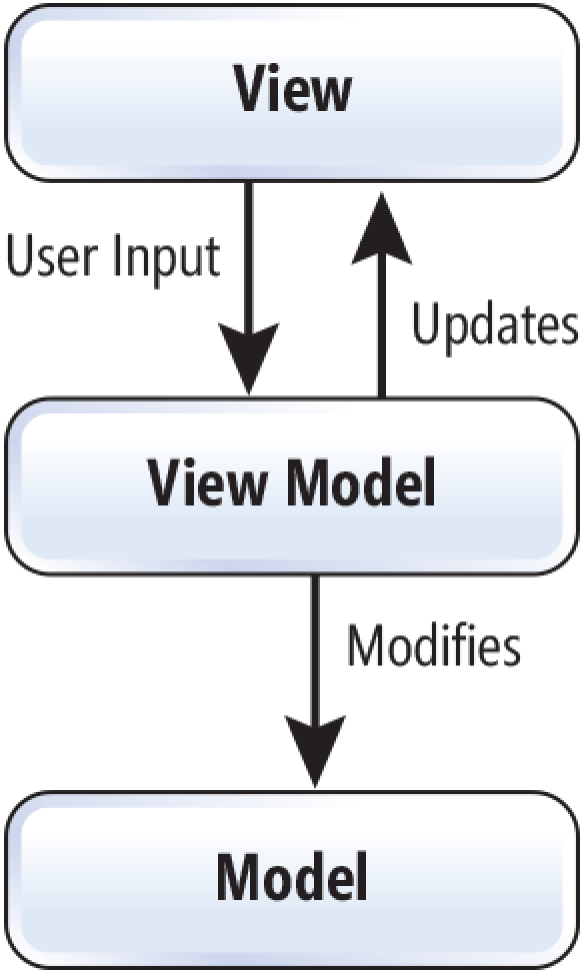
```
$.getJSON(url)
    .done(function(data) {
        // On success, "data" contains a list of movies
        var ul = $("<ul></ul>")
        $.each(data, function(key, item) {
            // Add a list item
            $('<li>', {
                text: item.Title
            }).appendTo(ul);
        });
        $('#movies').html(ul);
    });
```

MVC Pattern



- Model represents the domain data and business logic
- View displays the model
- Controller receives user input and updates the model

MVVM Pattern



- Model represents the domain data and business logic
- View-Model is an abstract representation of the view
- View displays the view-model and sends user input to the view-model

- **In a JavaScript MVVM framework, the view is markup and the view-model is code**

Javascript framework

- Backbone
- Angular JS
- Ember
- Knockout
- JavaScriptMVC
- React JS
- **Vue.js**
- …

Summary

- SPA benefits
  - bring a much-improved experience to the user
  - App is faster because less bandwidth is being used & no full page refreshes
  - App will be much easier to deploy in production, at least certainly the client part:
    - static server
  - Client is simple to version in production, allowing for simplified deployment and rollbacks to previous version

References

- Single-Page Applications: Build Modern, Responsive Web Apps with ASP.NET
  - https://msdn.microsoft.com/en-us/magazine/dn463786.aspx

- Why a Single Page Application, What are the Benefits ? What is a SPA ?
  - https://blog.angular-university.io/why-a-single-page-application-what-are-the-benefits-what-is-a-spa/

- Making a Single Page App Without a Framework
  - https://tutorialzine.com/2015/02/single-page-app-without-a-framework

Thank You !