# Project Objectives

Implement a Feature-level component , inside of an NG App that exhibits lazy-loading, and has automated testing.

## UI objectives

1. Implement with the most modern version of NG (Angular 10.0.07)
2. Implement an App level solution with a TopMenu and LeftNav using Angular CLI
3. Feature component (*called* ‘Member’) leverages the NG [*element*](https://angular.io/guide/elements) to create a true Web Component. *We should be able to use / re-use with any HTML based CMS or framework*
4. Use[*webpack*](https://webpack.js.org/concepts/)for compaction (downline-load speed) < 250 kb is great!
5. Implement Routine / Lazy Loading for the Add Member component and dynamic import, an optimization and App architecture pattern
6. Implement Tests: Test coverage of the NG App and feature component with Jasmine
7. Implement only the CreateMember() API through the NG Service and API stack
8. Host the App on Azure with a T.B.D. hosting option

*Optional*

* Evaluate if Ivy is meaningful (esp. if small apps), [article here](https://indepth.dev/angular-with-ivy-build-performance-review/)

# SVC Objectives

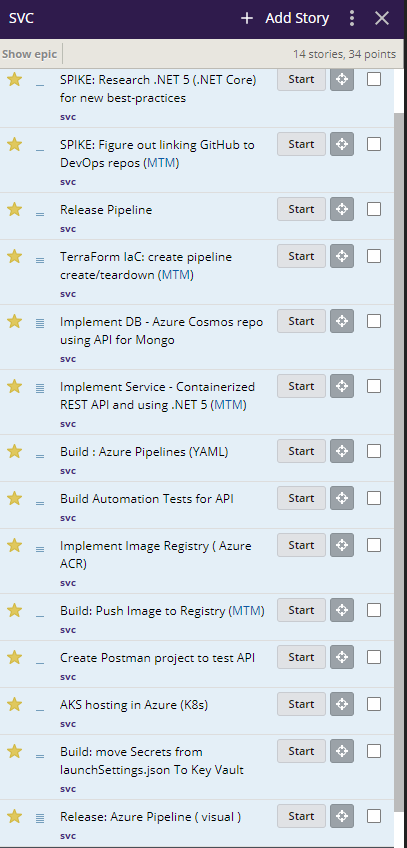
* Use .NET 5 ( successor to .NET Core and the merger with .NET Frk’s) for Service
* Containerize and have a Linux code base, tested locally in Docker For Windows
* Learn HashiCorp TerraForm for create IaC solution in Azure:
  + Azure Container Registry (ACR) image registry
  + AKS (Kubernetes Service)
  + SPIKE: Is there a Load Balancer in front of AKS? How to route to service hosted in K8s?
  + Create NS Record (DNS) name to refer to the service (LoadBalancer)
  + Hosted-Mongo (NoSQL) in Azure Cosmos, see [API for MongoDB](https://docs.microsoft.com/en-us/azure/cosmos-db/mongodb-introduction)
  + Both Provision and Tear-down with TerraForm (no charges after sandbox session)
    - Auto-tear down in Azure? Is that what [DevTestLabs is for – it can do this](https://azure.microsoft.com/en-us/services/devtest-lab/#overview)!

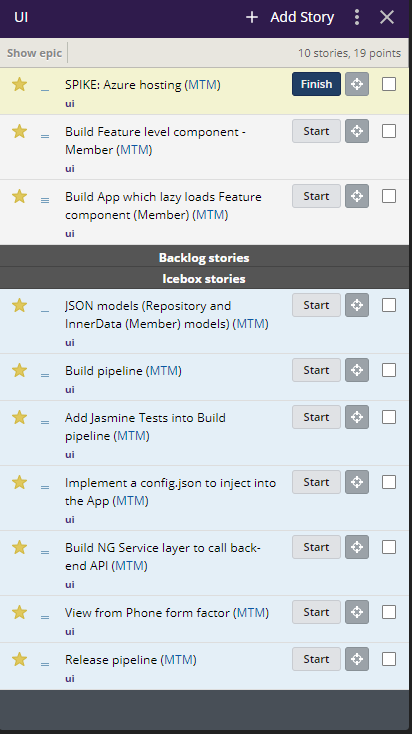
# Narrative

## Stories – from Objectives & Requirements

First, I wrote up the user stories in PivatolTracker for [project NG10E2E](https://www.pivotaltracker.com/n/projects/2463171). The UAT’s will be written or implied as I code. Tasks will also be added. GitHub check-ins will be tied to a PivotalTracker story.

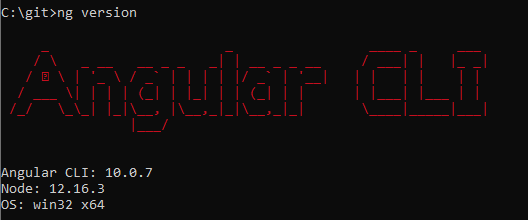
Initial Backlogs on 8/26/2020 are,

For the Microservice (SVC) epic:  


For the UI epic:  


## Environment Latest Tools

I’ll use Visual Studio 2019 Preview edition, which also installs the .NET 5 Preview. I’ll also install the latest Angular CLI (10.0.7).

1. I updated NG to 10.0.7  
   
2. I updated VS 2019 to v 16.8 Preview 2  
   
   1. Checked to verify the latest .NET 5 version is installed  
      

## Getting Started

### First story is a spike: How to host on Azure the NG App

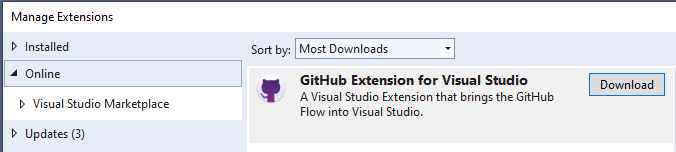
So here I guess I want to build the App with the CLI, then also learn the WebPack things I need to understand in one fell swoop.

Once I understand Webpack, I imagine the method of hosting will be easy to implement upon research of how to host on Azure.

Thesis: 1) WebApp is going to be easy or ok to use Webpack with  
2) There will be other ways, some obvious and some not. Blob URL has been mentioned, but it’s not a server (but can it act as one, given a URI to the Webpack? 3) smallest served Linux front-end server with Nginx on it, should also be considered.

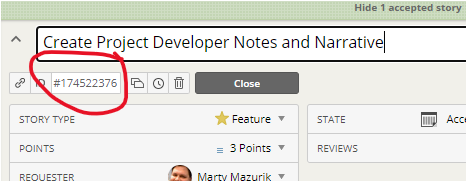
#### Sidebar: work to put my document into Github from a Pull Request (PR) into PivotalTracker

I found I didn’t have the ability to Connect in VS2019 to Github, looked for extension: bingo !



#### Branching strategy

I will create a branch, off of trunk (Main) using the Story ID (we will be at the story level on check-ins and merge requests; at issue is we should then ‘stash’ in between code (daily) for safekeeping. And with Stash is there any other work to do. (If I’m not religious about stash, will it kill me? Only if my local HD dies)



I create then a branch for the work **\174522376** gets created linked to the story.

And issue pull request for a merge. I’ll do this out of habit, although it is just me right now.