

Docker and Containerization

A CLAO SWAT Project

M&T Tech

{ TABLE OF CONTENTS }

- 01**  **Tech Stack**
Layout the technologies used in the development of the project.
- 02**  **Requirements / Project Goals**
What were the initial goals of the project, what did we hope to accomplish
- 03**  **Project Details**
How is the project currently setup, goes into technical details on pipeline
- 04**  **Sneak Peek**
A demo of the current iteration of the project
- 05**  **Our Team / Mentors**
Who helped plan and work on the project throughout the summer
- 06**  **Takeaways**
What did we learn from this project? What skills / improvements to our tech stacks



{ ABOUT THE } PROJECT

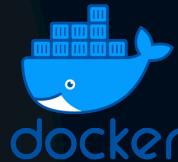
We need to automatically containerize our department's web applications, which would allow finely tuned environments tailored to those specific web applications to be portably deployed via *GitLab CI/CD* pipelines.

We seek to utilize Docker, a tool/platform that is used to automate the deployment of applications in lightweight containers so that applications can work efficiently in different environments in isolation.

{THE TECH STACK}



Web Application



CI/CD Pipeline



M&T Tech



{ CURRENT SOLUTION }



LONG MAINTENANCE TIMES



REQUIRES OFF PEAK HOURS



LACK OF ADAPTABILITY



M&T Tech

{ PROJECT GOALS }



Create a Docker container
that is capable of self
hosting a .NET 6 web-app



CI/CD to AUTOMATICALLY
deploy container to
designated environment



Migrate one real time
application to a Docker
container on OpenShift

{OUR GOALS}



Automation
Our containerization allows any new or existing application to be put through a pipeline, stored on the *Artifactory*, and deployed on *OpenShift* within a container without additional manual intervention.



Scalability
This process can easily be replicated or scaled across any web application by utilizing documentation that will be created following the completion of this project.



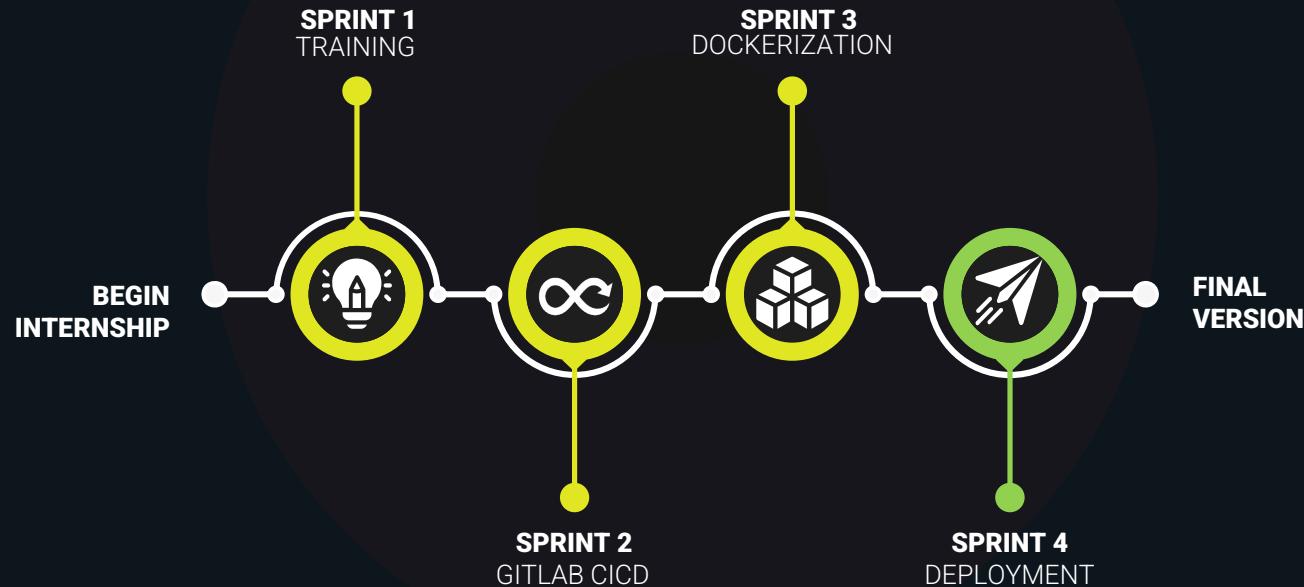
Cost Reduction

Assuming an average combined developer salary and operational cost of \$85, the cost for a resource that requires 10 hours of maintenance per month is:

$$10 \text{ hours} \times \$85 \\ = \$850 \text{ per developer}$$

If we continue to scale this with more developers working on a resource and multiple resources being maintained by different teams, the cost rises quite substantially.

{OUR TIMELINE}



M&T Tech

{ CHALLENGES }



Project Structure

Seeing that the project requires lots of stages and moving parts, understanding the structure of the project was a major challenge.

In order to overcome this, we spent time diagramming our idea of the structure and got verification from our lead, Anthony Steiner.



Tech Stack

As our team had very little experience with the technologies needed to complete this project, we spent a significant amount of time learning on Pluralsight.

Pluralsight allowed us to quickly overcome the learning curve of trying to understand new technologies and allowed us to make quick advances once we started the project.



Controller Endpoints

While reconfiguring a sample application into the current state, data from the .NET controller never reached the frontend.

Addressing this issue required collaboration with our lead, Anthony Steiner and co-lead Rishi Joshi, immense research into C# controllers, and the use of swagger to ensure that the endpoints indeed existed.

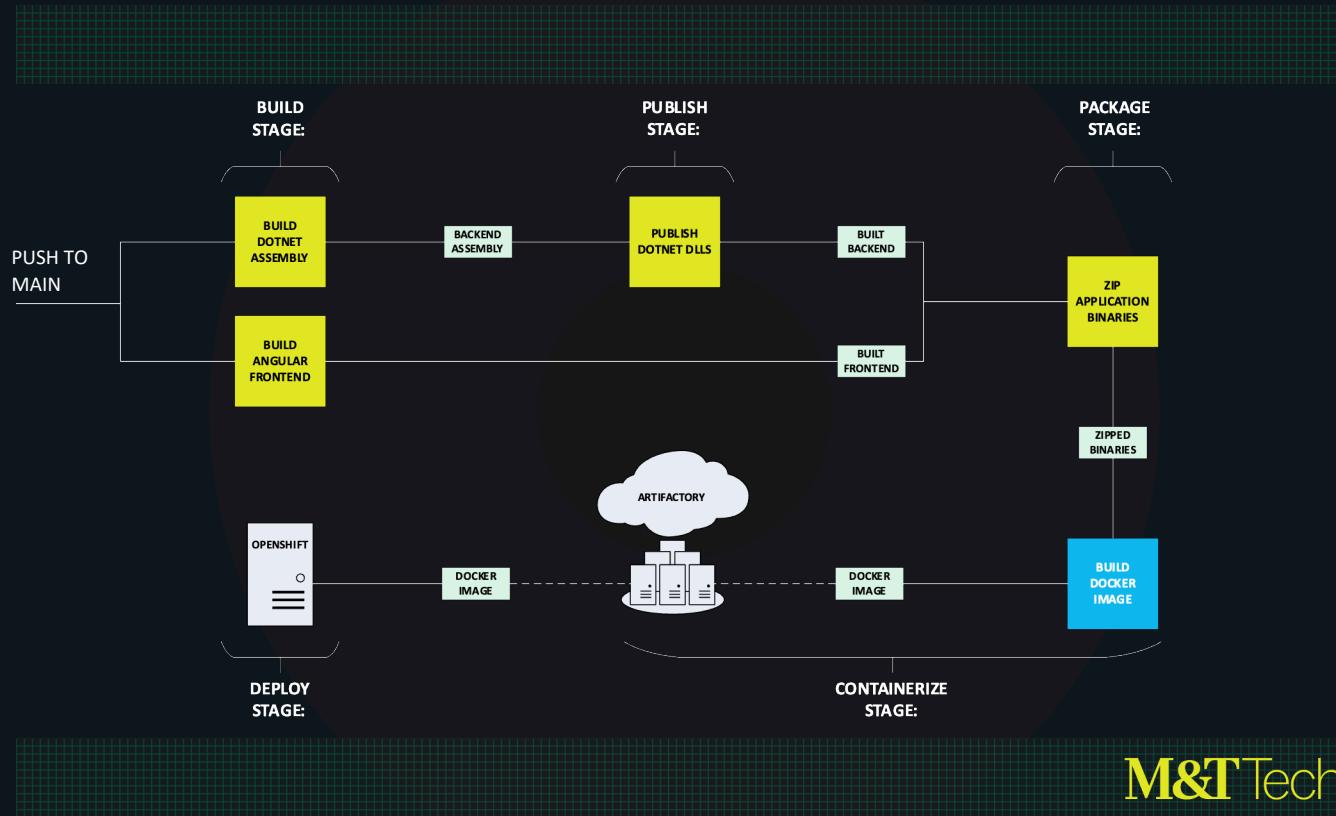


Initial Docker Build

Converting the application to run inside the docker container was challenging. As currently M&T uses windows specific dependencies for authentication. Whereas OpenShift only allows Linux images.

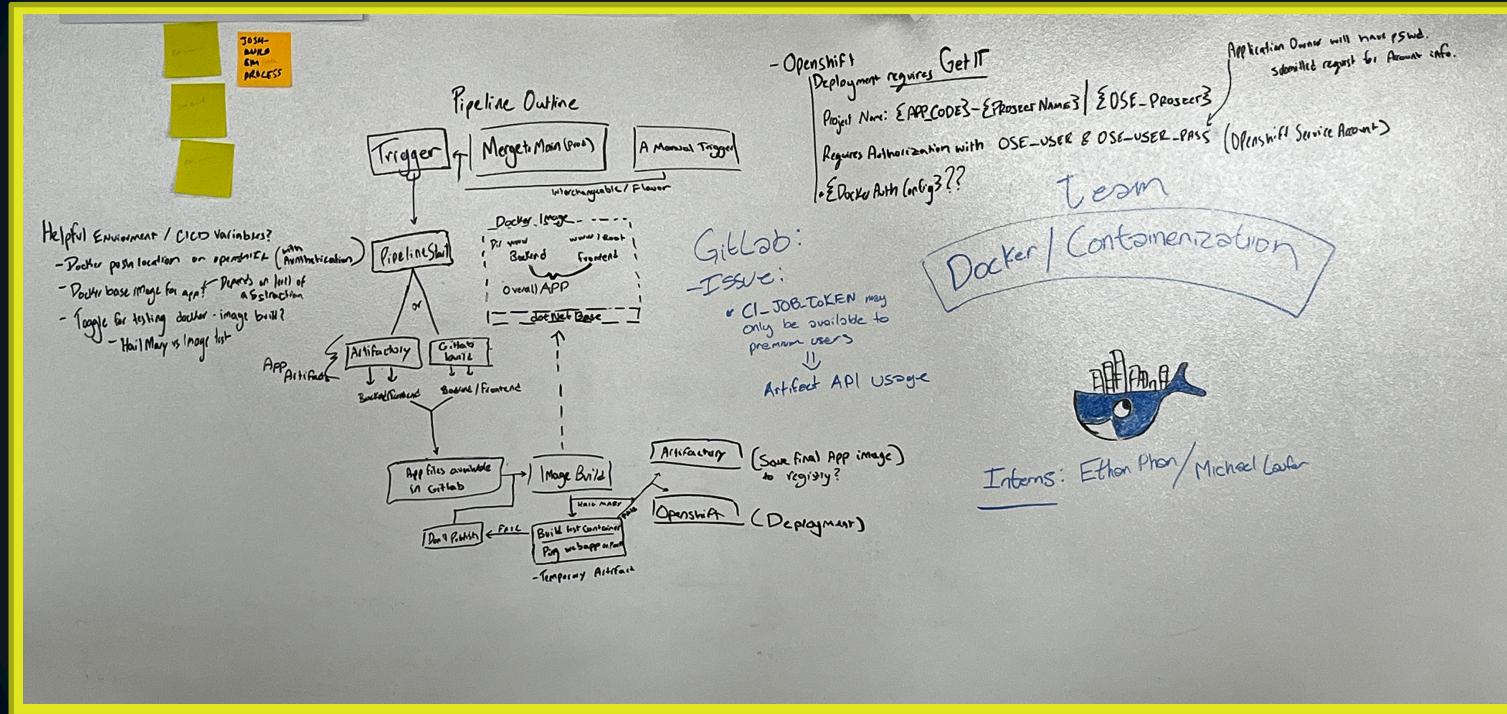
To address this, we will offload the Windows authentication to an external application and make use of Kestrel for HTTPS.

{PIPELINE}



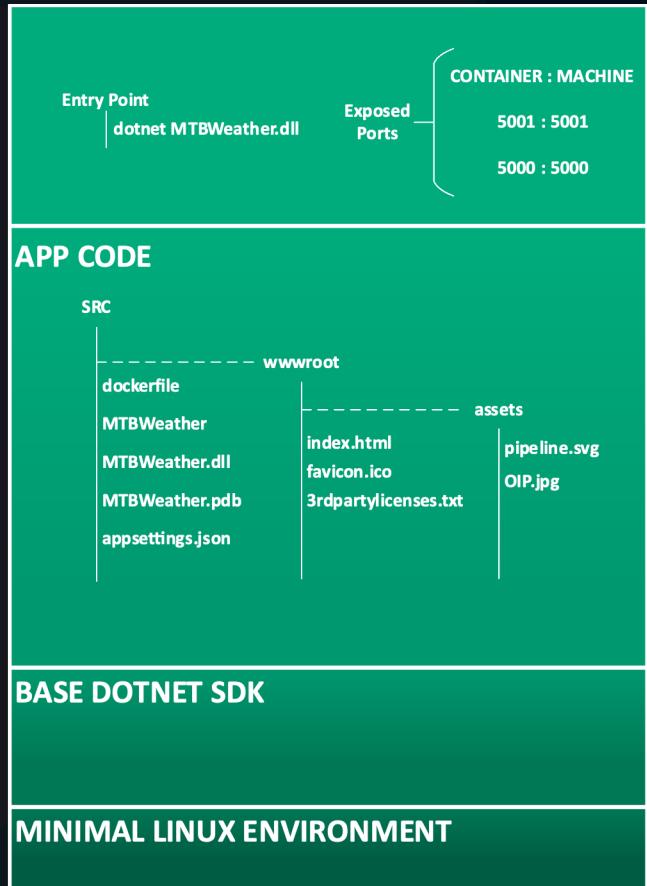
M&T Tech

{ ORIGINAL PIPELINE }



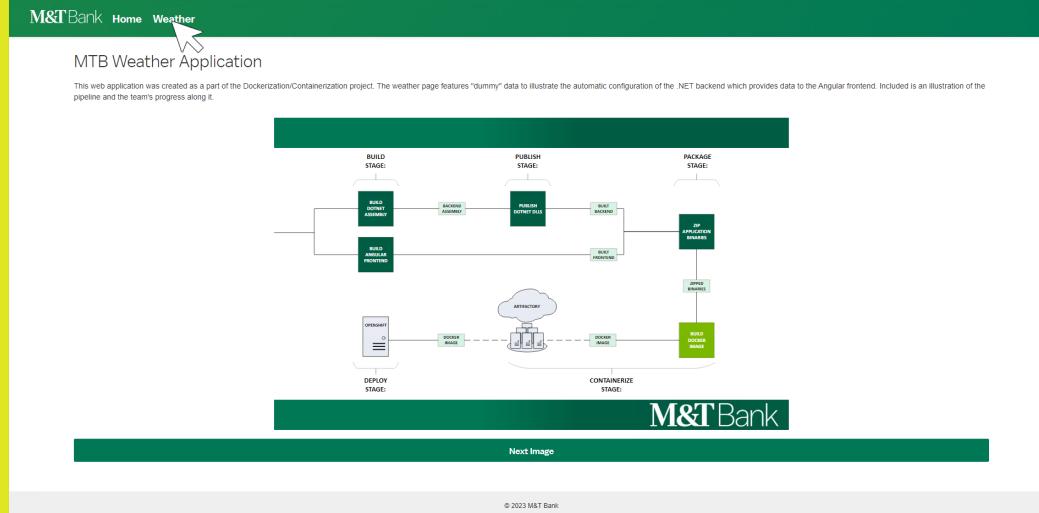
{ DOCKER IMAGE }

- OS (in this case a minimal Linux environment) is handled by *Docker Daemon*
- Base dotnet SDK handles the running of application (using an internal Artifactory docker image as the base)
- App code is the *application binaries* which is copied into the *Docker image's* working directory
- The image has ports 5000/5001 exposed and mapped to the same ports externally, for **http** and **https** respectively



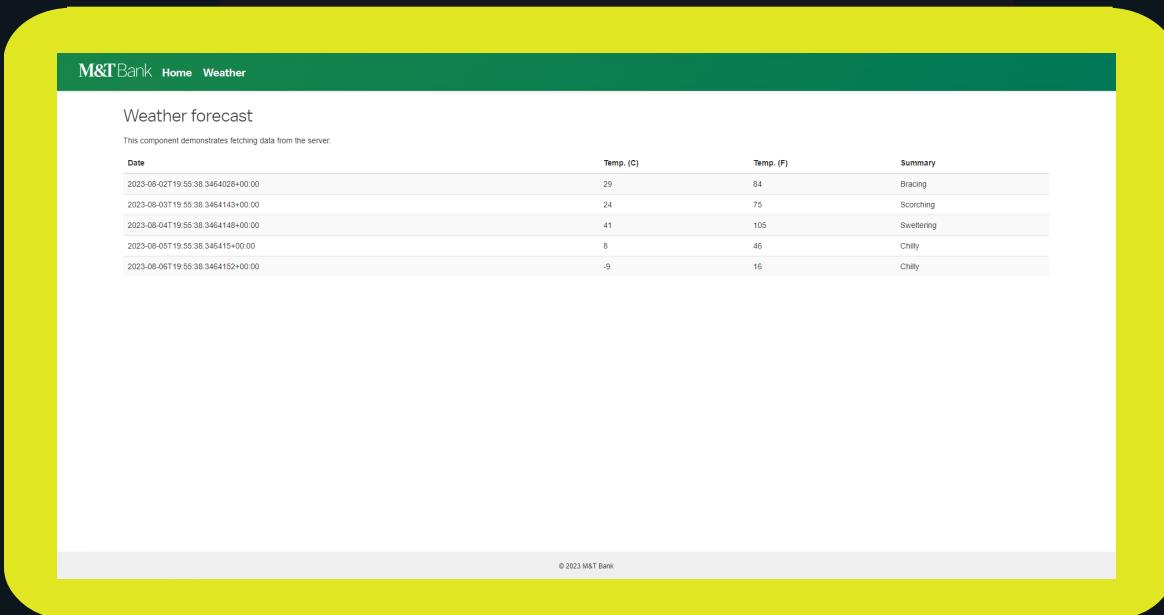
{ SNEAK PEEK }

Angular
frontend built
from *Docker*
container



M&T Tech

{ SNEAK PEEK }

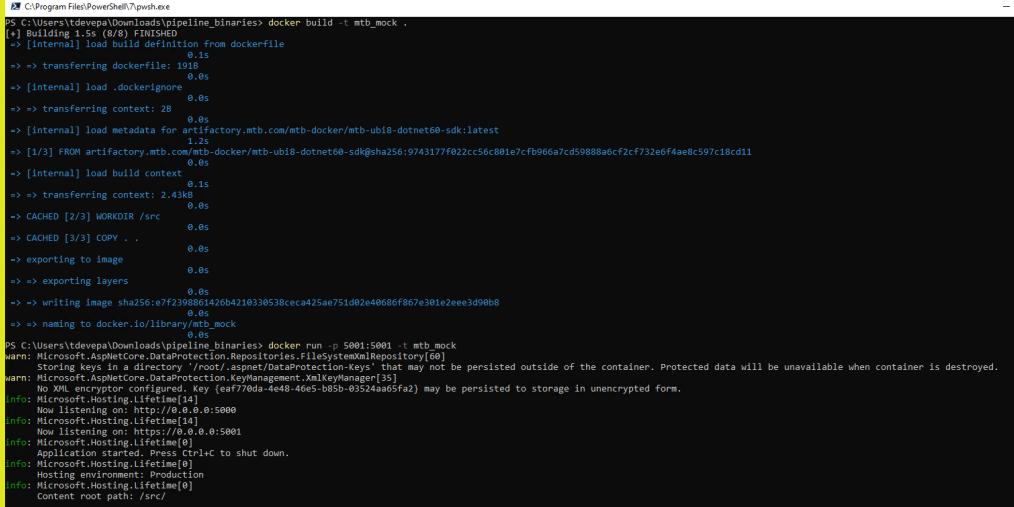


Working .NET
backend
sending “data”
within *Docker*
container

M&T Tech

{ SNEAK PEEK }

Running
Docker
container

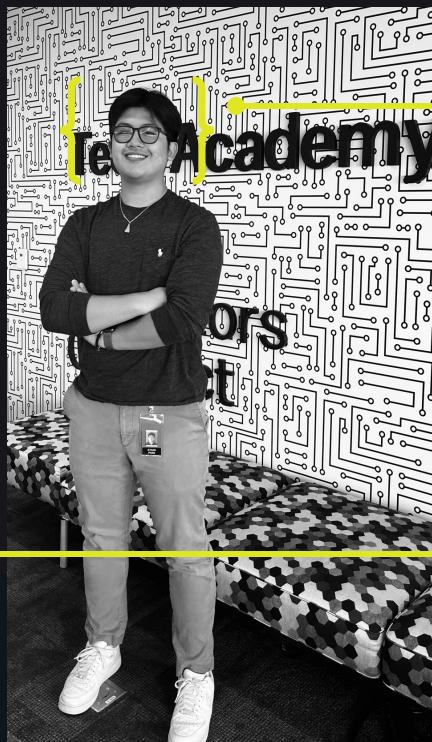


A screenshot of a Windows PowerShell window titled 'C:\Program Files\PowerShell\7\pwsh.exe'. The window displays two commands being run:

```
PS C:\Users\devepa\Downloads\pipeline_binaries> docker build -t mtb_mock .
[*] Building 1.5s (8/8) FINISHED
-> [internal] load build definition from dockerfile
    0.1s
-> => transferring dockerfile: 191B
    0.0s
-> [internal] load .dockerignore
    0.0s
-> => transferring context: 28
    0.0s
-> [internal] load metadata for artifactory.mtb.com/mtb-docker/mtb-ub18-dotnet60- sdk:latest
    0.0s
-> [1/3] FROM artifactory.mtb.com/mtb-docker/mtb-ub18-dotnet60- sdk@sha256:9743177f022cc56c801e7cfb960a7cd59888a6cfc2f732e6f4ae0c597c18cd11
    0.0s
-> [internal] load build context
    0.1s
-> => transferring context: 2.43kB
    0.0s
-> CACHED [2/3] WORKDIR /src
    0.0s
-> CACHED [3/3] COPY .
    0.0s
-> exporting to image
    0.0s
-> => exporting layers
    0.0s
-> => writing image sha256:e7f2398861426b4210330538ceca425ae751d02e40686f867e301e3eee3d90b8
    0.0s
-> => naming to docker.io/library/mtb_mock
PS C:\Users\devepa\Downloads\pipeline_binaries> docker run -p 5001:5001 -t mtb_mock
warn: Microsoft.AspNetCore.DataProtection.Repositories.FileSystemXmlRepository[68]
  Storing keys in a directory '/root/.aspnet/DataProtection-Keys' that may not be persisted outside of the container. Protected data will be unavailable when container is destroyed.
warn: Microsoft.AspNetCore.DataProtection.KeyManagement.XmlKeyManager[35]
  No XML key found for application key (eaF70da-4e48-46e5-b85b-03524aad65fa2) may be persisted to storage in unencrypted form.
Info: Microsoft.Hosting.Lifetime[14]
  Now listening on: http://[::]:5000
Info: Microsoft.Hosting.Lifetime[14]
  Now listening on: https://[::]:5001
Info: Microsoft.Hosting.Lifetime[14]
  Application started. Press Ctrl+C to shut down.
Info: Microsoft.Hosting.Lifetime[8]
  Hosting environment: Production
Info: Microsoft.Hosting.Lifetime[6]
  Content root path: /src/
```

M&T Tech

{ THE TEAM }



Ethan Le Phan

TIP Intern, rising junior at University at Buffalo

Michael Laufer

TIP Intern, rising senior at University at Buffalo

M&Tech

{ OUR MENTORS }



Gurdarshan Singh
Bhogal, Manager



Anthony Steiner,
Lead



Rishi Joshi,
Co-Lead

{TAKEAWAYS}

ETHAN PHAN

From building the mock Angular/.NET applications to designing GitLab pipelines, working on this project allowed me to learn about various aspects of software development. I am grateful to be given the opportunity to expand my tech stack, obtain relevant experience, and connect with others working to achieve the same goals.

TESTIMONIES

MICHAEL LAUFER

This project was my first time experiencing the full suite of DevOps in a professional setting. I gained experience creating my own CI/CD pipelines to not only build an application, but to completely automate the construction of a docker image, and to research processes within the bank to utilize docker for deployment purposes.



{ PROJECT RESOURCES }

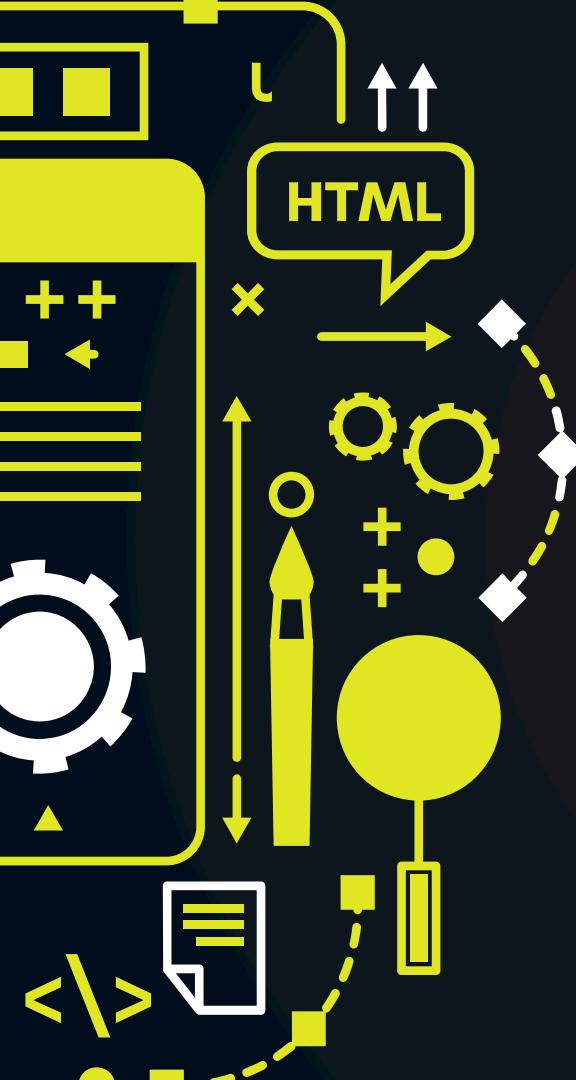


 Confluence



 GitLab

M&T Tech



THANKS!

Feel free to ask questions! And Connect with us!



Ethan Phan

<https://www.linkedin.com/in/ethanlphan/>



Michael Laufer

<https://www.linkedin.com/in/michaellauf2/>

