3) What other information would you add to health endpoint json object in step 2? Explain what would be the use case for that extra information?

The service uses the standard Nuget package, Microsoft.Extensions.Diagnostics.HealthChecks which enable standard health checks and allows custom health checks to be implemented. The health checks making use of this approach target middleware services, the UI, and system-level services, networks, etc.:

- Entity Framework (ORM) such as valid connection string
- External URL validation
- SQL Server
- MySQL
- SQLite
- RabbitMQ
- Elasticsearch
- Redis
- System: Disk Storage, Private Memory, Virtual Memory, Process, Windows Service
- Azure Storage: Blob, Queue and Table
- Amazon S3
- Network: Ftp, SFtp, DNS, TCP port, Smtp, Imap
- MongoDB
- Kafka
- Kubernetes
- UI
- InMemoryStorage

Most deployments utilize a log consolidator (Splunk, Sumo, etc.) but any health check should facilitate access to the application's logs which might require implementing a custom health check.

5) How would you automate the build/test/deploy process for this application? (a verbal answer is enough. installation of CICD is bonus, not required)

What branching strategy would you use for development?
What CICD tool/service would you use?
What stages would you have in the CICD pipeline?
What would be the purpose of each stage in CICD pipeline

This has been a one person project so Github Flow works. If I was scaling (team/hosts/versions) then Git Flow or Gitlab Flow. There were no PRs and no ticket so there was no need to create a develop branch and go with Git Flow or Gitlab Flow.

Most environments are not greenfield so you do not pick the CI/CD platform (ASML/Specific Diagnostics was Bamboo, Ozzy/DocuSign was Jenkins legacy and Groovy based, etc., Ozzy moved to CircleCI). For AWS applications I lean towards Elastic Beanstalk with Jenkins. For Azure I lean towards ADO pipelines. If there was time to build out the CI/CD, I'd use Azure DevOps because I have a $150 per-month credit. I am more comfortable in Jenkins but ADO is free given the monthly credit. If a company was funding the project, I'd be with AWS: Elastic Beanstalk with Jenkins.

For stages there would be a Source Stage to detect a change to main (yes, it defaults to main not master). The source stage triggers the build on a change to main as this is Github Flow. As this is a solo effort and I don't have unreliable team members, I am not concerned with gated check in.

I'd need a Build Stage broken down into code and Docker.

The test state is typically smoke, unit, and integration. If the is project scaled each health check would be implemented by specific interface so I could use IOC to test each individual health check and test the service with MOQed health checks. An Integration test would require an integration environment. Given this is a solo effort the closest I'd get are unit tests with IOC/MOQ and no smoke/integration tests.

For the Deploy Stage, I'd go to production as this is a one person project. For a larger team/project I'd use a development, QA, integration, staging, and production environments if the project merited it. If it made sense, changes would go straight to production but I would need to understand the business case.