Project Intermediate Report
Project Intermediate Report

• Due next week.
  • No late day/grace period.
  • Slightly longer, 2-3 pages.
  • Reasonably well formatted, e.g., ACM template.
Project Intermediate Report, cont’d

• Opinion, how will I write the report?
  • Introduction:
    • Incentive: How does the application solve your problem? Why is it important to you? Why do you want to do it?
    • Status quo: Are there similar products? How do they compare to mine?
  • Architecture/Specification:
    • How does users use your application?
    • How do you design your control flow and data flow?
    • How do microservices work together?
Project Intermediate Report, cont’d

• Technology:
  • Describe key algorithms.
  • Describe how you will put your design into real code, with library/framework/etc...
  • Name what Azure components you’ll use and how you’ll use them.

• Implementation
  • Be specific about what is done and what will be done. Give a timetable.
  • Where’s your data? ML, IoT, etc.

• Evaluation:
  • What is deliverable? In what format?
  • What will you demo?
  • How can we evaluate your implementation?
• TA Comment Feedback:
  • (If not answered in previous sections) Please answer the questions TA raises.
  • Be specific, especially for the technically trivial issues.
How will we grade your report?

• Do you address our concerns as raised in the initial report?
• Have you really started with your project?
• ...
Some tools you might use
Docker

• Everything starts from a docker file.
  • https://docs.docker.com/engine/reference/builder/
  • Start from some base “images”.
  • Set up your own environment with very simple terminologies.
Docker

• Docker file -> docker image
  • Using docker build command.
  • You can maintain different versions for management.
  • You can also publish your image using docker hub.
  • This is a time-taking process and sometimes caching by layers saves your time.
Docker

• Docker image -> docker container
  • Using docker run command.
  • You can feed more parameter, like port mapping, to the container.

• Many other commands like
  • docker start
  • docker container rm
  • docker logs
Docker

• Practices:
  • Docker is not a VM, think of `sysctl` command.
  • Nested containers usually is not a good idea, e.g., I want my database "live" together with the service?
  • You need to design how containers talk to each other, thinking about proxy servers.
Proxy server

- Forward proxy vs. reverse proxy
Proxy server

• Does it support the “protocol” you use?
• Does it work with http2 (TLS/SSL)?
• Can it work with microservices, service mesh?
• My guess for your pick:

### envoyproxy / envoy

- Watch: 608
- Fork: 3.9k
- Star: 20.6k

### nginx / nginx

- Watch: 979
- Fork: 6k
- Star: 17.3k
How does your application communicate?

• REST APIs:
  • REpresentational State Transfer
  • Uniform Interface, Client-Server, Stateless, Cacheable, ...
  • Doesn’t necessarily be http request. But http request can be restful.

• Easy to implement using popular server frameworks.
  • Python Flask
  • Java Spring Boot
  • You define the “access point” and the framework help you with the rest of work.
RPC
Demo

https://grpc.io/docs/platforms/web/basics/