Office Hours 10/10

Monday, October 10, 2022

LOW-LEVEL DESIGN:

• Low-level design will only be logging and registration for milestone 2

DATA BASE AND ALTERNATE TECHNOLOGY:

- MariaDB vs Azure
- Alternative technologies for students with mixed OS
- Data store is SQL server which is relational
 - MariaDB is also relational
 - o Doesn't matter as long as its relational
 - Our infrastructure is mainly AWS (Azure not recommended due to added complexity)
- Web server and database are NOT cross platform which is why alternate technologies were listed
- Technology lecture will be done after networking lecture
 - Shows both windows environment and mac environment
 - o Probably don't want to rely solely on his technology lecture

RESUBMISSION OF DOCUMENTS

- High-level design should give help or pointers when coding
 - HLD will not have feedback given
- BRD has to be approved
- Project plan is onus on us
 - o If we think its acceptable/feasible then we can proceed forward
- Every project is constrained by budget, resources, and deadline/time
 - o All we have control over is the complexity of the project
 - Rework the scope if it seems too difficult to implement

EVERYTHING IS LOCKED IN BY THE END OF THIS SEMESTER

Need to run any changes by him NOW this semester

• Any changes afterwards will be -10% per deadline missed

Low-level Design

Should be sequence diagrams

High-level Design (Another Group's Feedback)

Did not mention which architecture will be used for front-end or back-end

- 1. Not labeled in diagram
 - What are the arrows? What do they mean?
 - o If a domain model doesn't extend to the front-end/back-end it should be a different color

Not enough information about how each of the different behaviors will work for each of the layers

- 2. What is security for each layer
 - They only have security in data store and entry method
 - Issue is: System only has 2 points which are being guarded
 - There are ways to get around a layer using something like buffer overflow
 - Rest of system will be exposed
 - o Security checks in back-end only mean every request will have a security check
 - Data store on a different environment will allow a network latency every single time
 - Should check before you even try to enter data store

- Don't wait until you get to data store to check, check before the call is even made
- o Typically want security checks throughout the system, will lead to a more robust solution
- o Errors can happen at any point: front-end or back-end
 - Web applications must make an initial request from a browser to another environment, meaning an error on the server end will
 return to the client
 - ☐ Breaks the app since nothing was handled on client end
- 3. Explain the decision behind why you are using this technology
 - Someone will come and read the document, must explain why you chose this specific technology
 - Don't include development tools
 - Can still take design and apply it to a different tool
 - o Moreso decisions for why you chose it

OTHER NOTES ON HLD

- Not elaborating on WHAT EACH PURPOSE OF EACH LAYER IS
- NOT DESCRIBING IN DETAIL WHAT DESIGN BEHAVIOR IS
- NOTE DOWN BOUNDARY BETWEEN FRONT-END AND BACK-END
- Flaws in each layer
- Need to address messaging in each layer (?)
 - o Couldn't really hear him
- Say specifically where logging will happen at each layer/feature
- Try to crash gracefully for each exception/error
 - Save data to drive which is full, handle exception by noticing error and tell user "file full"
 - o Must explain error handling at every level of the design
 - What happens if a user tries to get to a path which doesn't exist? UI LEVEL ERROR
 - User just changes the URL himself, describe what will happen if it doesn't exist or not allowed
 - General behavior in that specific layer
 - HTTP is entry point error, but incomplete by itself to describe entry point