Architecture Design: Lewis Instructional Software Architecture

Part 1 – Identifying the Architecturally Significant Requirements (ASRs)

1) Assembling an architecture that is low-cost and supported by students
    a. Category: Operational and Cost
    b. Significance: This architecture should be able to be accessed at a no/low-cost for both students and professors and allow them to have more control over their work instead of relying on university resources and access which could pose a potential problem if servers or other resources are unable to be accessed or maintained for periods of time that could slow down learning. This instead could benefit students by enabling them to learn cloud-based tools such as Microsoft Azure to manage their projects, which is a skill needed in many tech jobs.
2) Support for NoSQL databases with ability to eventually support SQL databases
    a. Category: Data Management and Modifiability
    b. Significance: This architecture should be able to handle NoSQL databases and resources that are easy to access by students and professors. It should also be able to eventually support SQL databases as well and allow for enough storage for medium-sized projects for classes to illustrate how these query languages work. It also should be able to handle various database functions and not delay very long when making any modifications to the database.
3) Additional resources, reference implementations, and documentation available
    a. Category: Teachability and Reusability
    b. Significance: Resources should be plentiful on this platform that can be accessed by both students and professors. It should be able to have a range of applications from building a basic web app like a “Hello World!” project or to create something more intensive with a team to handle APIs, servers, databases, authentication, etc. The resources and reference implementations provided should be somewhat interactive to help supplement what the professors plan to teach in class while also being broken down in a way that is easy to understand.
4) Support for web-based clients using React, HTML, CSS, or JavaScript
    a. Category: Usability
    b. Significance: This platform should be able to help students learn how to develop and deploy web applications with React, HTML, CSS, and/or JavaScript. It needs to be usable in the way that a student or professor should be able to create a web app of their own and then have support to help run in the background to test it and teach them about these languages by creating something. It also has to be useful in that this

platform should be relatively easy to access and create web applications with as long as there is some kind of device provided, maybe an IDE, and WiFi.

5) Accessible through a sign-in with Microsoft, GitHub, or Google
    a. Category: Security and Usability
    b. Significance: I think this ASR can fit under these categories because most CS students have an account with one or more of these systems or are required to make one in class with their professors since they are so common in the industry. By having support to sign in or make a separate account with all of these systems, it can help verify that only Lewis students and professors can access this platform to avoid any unexpected users who might cause harm or use too much of the limited storage space if there is any provided.

Part 2 – Defining Quality Attribute Scenarios (for 3 ASRs)

Selected ASRs:

- ASR 1: Assembling an architecture that is low-cost and supported by students
- ASR 2: Support for web-based clients using React, HTML, CSS, or Javascript
- ASR 3: Additional resources, reference implementations, and documentation available

Scenario Format: [Stimulus] causes the system to respond [response] while in [context]

Scenario for ASR 1: A first-year CS student is looking to deploy their Hello World HTML, CSS, and JavaScript project on Azure with a free-tier plan. In response, the system will provide a guide to set up the appropriate environment and provide a template for the student to set up their project. The context is that this can all be done with only a no/low-cost hosting tier on Azure and maintained by the student without any support from the university.

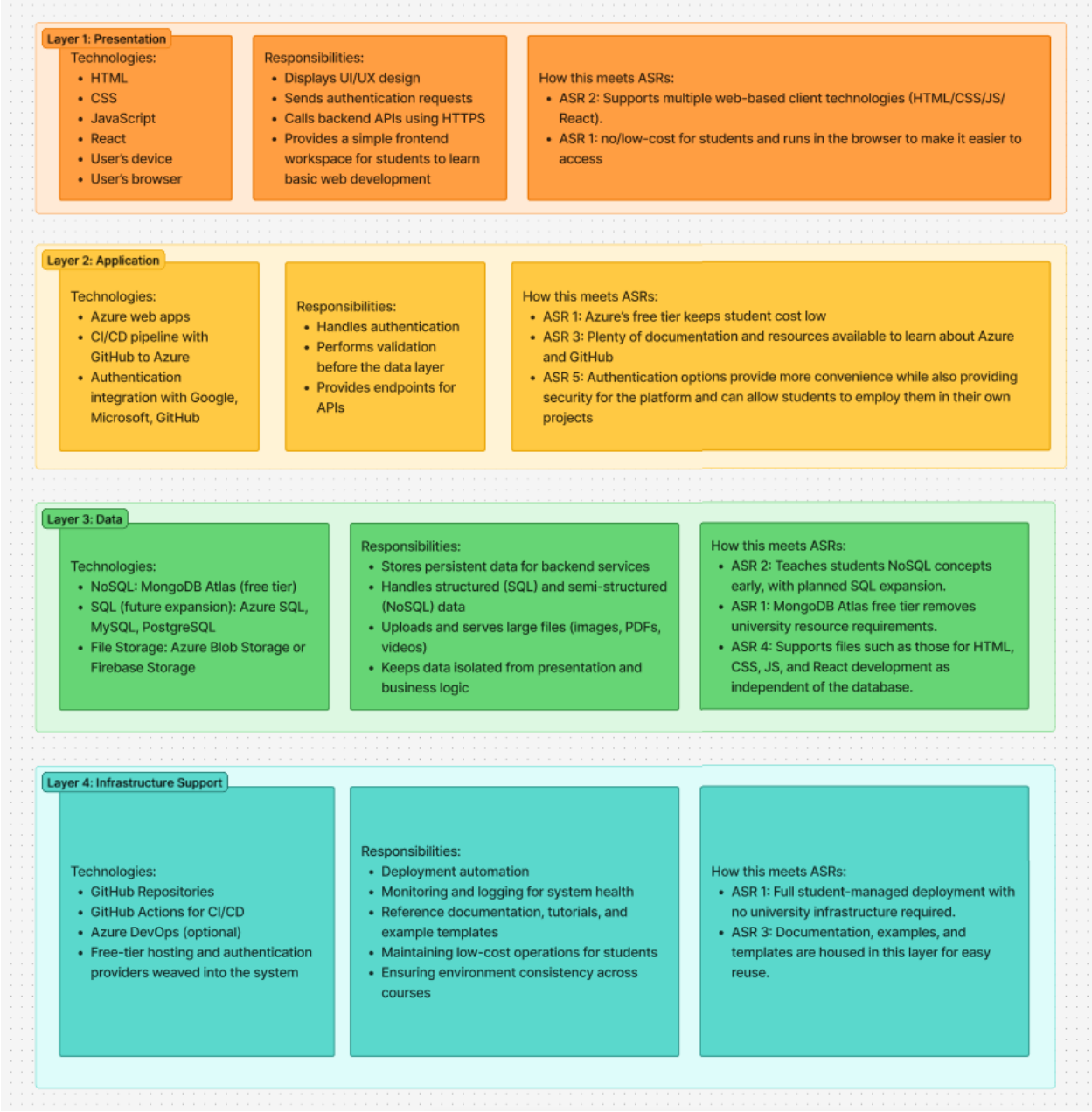
Scenario for ASR 2: A student who is new to learning React finishes their project and wants to push it so the system can build it and they can test it out. In response, the system builds, hosts, and then shows the student the results on HTTPs on their browser to view while also handling any API calls that may be involved as well. In this context, the student has some experiencing with HTML, CSS, and JavaScript, but is not as familiar with some of the other components that go into running and deploying a React app.

Scenario for ASR 3: A new professor is preparing to teach a course on databases and decides to look at this application for resources to brush up their knowledge and to provide their students each week in the class. Based on the professor's goals for the class, the system will provide him/her with links to tutorials online, reference

documentation, and easy-to-use templates with support for different languages. The context for this situation is that the professor should be able to complete this task without needing to leave the system and should be able to rely on it for all the necessary information to include in their course so it is aligned with the goals of the CS department as a whole.

Part 3 – Designing a High-Level Architecture

1) Here is my High-Level Architectural Diagram:



2) Justifications
    a. I think my architecture satisfies the 5 ASRs that I chose to include because it starts with incorporating how the student will connect to the platform using HTTPS and then will authenticate them to the platform through a method of their choice (most likely, I think this could be easily done with their Google accounts since every Lewis student has a Gmail). These methods enhance security when accessing the platform which is critical if students are developing more complex projects and supports ASR 1 and ASR 5. After that, they will be able to access the platform where they will be able to view various resources and tutorials for creating their first web apps with HTML, CSS, JavaScript and/or React depending on the class they are taking. They will also have plenty of ways to learn about hosting their projects through Azure which is a great way to take a simple project they may have made with the previous languages and then host it to be able to show it off on a resume or to their family and friends. I think by enabling them to learn these concepts earlier with the right resources will help them accomplish the goals of ASR 3 and ASR 4. Finally, they will be able to learn about databases and incorporating them into their web applications and meet the requirements of ASR 2 to learn NoSQL before SQL. All of this is at a no/low-cost to students which goes back to supporting ASR 1.
    b. I decided to take a layered approach to this diagram because I felt like it made more sense to explain the stack better. I think this can give students and professors a more structured approach to learning and teaching these concepts because it's about starting off with something fairly simple like using HTML, CSS, JS, and/or React to create a web page and then moving through the structure to host it and use CI/CD and learn about important tools on Azure and GitHub. Then the students can learn more about databases for their sites as well while it still being a no/low-cost to them. It also allows for multiple skill levels of students to interact with the architecture because they will be able go as far into the content as they need to for their courses.
    c. Some of the trade-offs for this stack include using NoSQL first before SQL. I think by learning this skill first, it can help students create something that can easily interact with their hosted website whereas SQL can be a little more tedious to learn and setup for a website. Plus, it gives students more confidence when looking for jobs because there is a higher need for learning specific tools that can work across multiple platforms. Another trade-off that was made was selecting MongoDB instead of Firebase for database learning. As much as I personally like Firebase, I think it would

be better for students to learn something beyond a tool existing in the Google ecosystem.