**20L-0943**

**Saadan Khokhar**

**SQE Assignment 3**

# Architectural and Component level modularity requirements

## **Micro Services**

Micro services - also known as the micro service architecture - is an architectural style that structures an application as a collection of services that are :-

* Highly maintainable and testable
* Loosely coupled
* Independently deployable

The microservice architecture enables the rapid, frequent and reliable delivery of large, complex applications. It also enables an organization to evolve its technology stack.

Microservices architectures make applications easier to scale and faster to develop, enabling innovation and accelerating time-to-market for new features. With a microservices architecture, an application is built as independent components that run each application process as a service. These services communicate via a well-defined interface using lightweight APIs. Services are built for business capabilities and each service performs a single function. Because they are independently run, each service can be updated, deployed, and scaled to meet demand for specific functions of an application.

### Modular Folder Structure

This structure is interesting for projects with a larger scale. The main idea is that each module we define has all the code related to it and only code from the module itself is imported. When we have several modules that need the same piece of code, we can write it into a shared folder and import it into the different modules. The fundamental rule to be followed is not to import code between modules. However, the great drawback of this structure is that it can be complex to define what a module is and this can has a great impact on the success of our organization**.**

### Class/components

Components are independent and reusable bits of code. They serve the same purpose as JavaScript functions, but work in isolation and return HTML via a render() function. Components come in two types, Class components and Function components. Class component is instantiated and a different life cycle method is kept alive and being run and invoked depending on the phase of the class component. Moreover, React lifecycle methods can be used inside class components (for example, componentDidMount). It requires different syntax inside a class component to implement hooks. Constructors are used as it needs to store state.

Example: constructor(props) { super(props); this.state = {name: ‘ ‘}}

### 

### Reusability

Components are usually designed to be reused in different situations in different applications. However, some components may be designed for a specific task. Reusability is the use of existing assets in some form within the software product development process; these assets are products and by-products of the software development life cycle and include code, software components, test suites, designs and documentation.

**Extensibility**

A component can be extended from existing components to provide new behavior. Extensibility is a measure of the ability to extend a system and the level of effort required to implement the extension. Extensions can be through the addition of new functionality or through modification of existing functionality.

### Linters for better code quality

A code linter is a tool that automatically checks your code to see if it meets your project’s or company’s coding convention. If it doesn’t match the convention, linters will show these errors as warnings. During software development, these warnings tend to get overlooked when developers prioritize speed over quality. But these small errors can accumulate fast in a short amount of time, leading to a significant workload for your team. To avoid these hefty workloads, linters identify these errors quickly so that your team can resolve them quickly, too.

## Performance Optimization

Performance optimization, also known as “performance tuning”, is usually an iterative approach to making and then monitoring modifications to an application and its database. It could involve adjusting the configuration of the database and server, or making changes to the applications and the [SQL](https://www.sciencedirect.com/topics/computer-science/structured-query-language) that maintain and query the database. As authors of this book, we can't participate in the specific modify and monitor iterative processes being carried on by any of our readers and their IT organizations. But we can describe factors that are likely to apply to any [Asserted Versioning](https://www.sciencedirect.com/topics/computer-science/asserted-versioning) implementation.These factors include

* Number of users
* Complexity of the application and the SQL
* Volatility of the data
* DBMS and [server platform](https://www.sciencedirect.com/topics/computer-science/server-platform).

High performance systems have complex, diverse and rapidly evolving architectures. The span of applications, workloads, and resource use patterns is rapidly diversifying. Adapting applications for efficient execution on this spectrum of execution environments is effort intensive. There are many performance optimization tools which implement some or several aspects of the full performance optimization task but almost none are comprehensive across architectures, environments, applications, and workloads.

### Usage of Cache

Cache is memory placed in between the processor and main memory. Cache is responsible for holding copies of main memory data for faster retrieval by the processor. Cache memory consists of a collection of blocks. Each block can hold an entry from the main memory.

Cache memory is important because it improves the efficiency of data retrieval. It stores program instructions and data that are used repeatedly in the operation of programs or information that the CPU is likely to need next.

### Usage of Content Delivery Network

To minimize the distance between the visitors and your website's server, a CDN stores a cached version of its content in multiple geographical locations (a.k.a., points of presence, or PoPs). Each PoP contains a number of caching servers responsible for content delivery to visitors within its proximity. In essence, CDN puts your content in many places at once, providing superior coverage to your users.

### 

### Usage of Lazy Loading in web and asset workflows in games

Lazy loading is a technique for waiting to load certain parts of a webpage — especially images — until they are needed. Instead of loading everything all at once, known as "eager" loading, the browser does not request certain resources until the user interacts in such a way that the resources are needed. The advantages of lazy loading are as follows:

**Faster page load:** All else being equal, webpages with smaller file sizes load faster. With lazy loading, a web page starts off smaller than its full size and thus loads faster. [Speedy web performance](https://www.cloudflare.com/learning/performance/why-site-speed-matters/) has numerous benefits, including [better SEO](https://www.cloudflare.com/learning/performance/how-website-speed-boosts-seo/), [higher conversion rates](https://www.cloudflare.com/learning/performance/more/website-performance-conversion-rates/), and an improved user experience.

**No unnecessary content:** Suppose a page loads multiple below-the-fold images but the user exits the page before scrolling down. In such a case, the bandwidth used to deliver the images and the browser's time spent requesting and rendering the images were essentially wasted. In contrast, lazy loading ensures that these images only load when necessary. This saves time and processing power, and it may save money for the website owner because less [bandwidth](https://www.cloudflare.com/learning/cdn/how-cdns-reduce-bandwidth-cost/) is used.

### Usage of PWA

Progressive Web Application (PWA) is a web standard that uses design patterns and advanced technologies to deliver a fast and reliable experience to users on both mobile and desktop devices. It can work on any platform using standard-compliant browsers with the aims to: Increase customer engagement.

With PWAs you can build one website that functions as an iOS and Android application. This saves the cost and complexity of building a separate experience of iOS and Android, while giving users the best functionality regardless of how they are accessing the content.

A progressive web app (PWA) is a website that looks and behaves as if it is a mobile app. PWAs are built to take advantage of native mobile device features, without requiring the end user to visit an app store, make a purchase and download software locally. Because a progressive web app is lightweight and doesn't put a lot of demand on a device's resources, it loads pages fast. Therefore, people will spend more time on your webpage.

### Service workers and local storage

## Service worker may refer to: Social service worker, a person engaged in social work. Pink-collar worker, a person in the service industry whose labor is related to customer interaction, entertainment, sales or other service-oriented work.

## Local Storage data is persistent, unlike cookies which have an expiry date. To clear data from local storage, you’d be forced to manually clear the data from the application tab in your browser developer tools. However, your PC can automatically (meaning without your consent or permission) delete data from local storage when it’s running out of file space to make room for more files. Local Storage is the simplest way to store key-value pairs of data on the browser. Other methods are using *IndexedDB, WebSQL (Deprecated, not advisable to learn or use), File System API and Application cache.*

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## Explore Security Requirements

### Secrets Management

### Application Firewalls

### SAST

### In-built security

### Security standard compliances

## Reliability Requirements

### Auto Scaling

### Load Balancing

### Cloud based and Server less deployments

### Fault tolerance

### recoverability

## Process Requirements

### Versioning and Release Management

### Issue Life Cycle Management

### Open Source contribution

### Requirements for developers

### Review Process

### 

## Documentation Needs

### User Manuals

### Installation/Deployment document

### Development/extension documents

### Tutorials