SURGE GLOBAL

Written Questionnaire

KAUSHALYA SAMMANI MANAWARDHANA
ksmanawardhana@gmail.com

SURGE GLOBAL – WRITTEN QUESTIONNAIRE

1. Explaining what is design pattern and how we can use design patterns in projects.

   As I know,simply a software design pattern is a general,reusable solution that taken to solve a common problem when we design any system or an application.It is not like a framework or library,it is more of a template to approach the problem.Mostly these patterns are used to support OOP(Object Oriented Programming).
   Currently there are about 26 patterns discovered and they classified into 3 main groups.

   - Creational Patterns
   - Structural Patterns
   - Behavioral Patterns

   The usefulness of using a design pattern is that the design pattern can speed up the development process. Using theseno need to reinvent patterns every a problem arise.Also Increase Code readability.

   Design patterns provide a standard terminology and are specific to particular scenario.

   In web development most used design patterns are Observer, Singleton, Strategy, and Decorator. Design patterns makes it easier for developers to communicate about problems and to improve code readability and architecture in early stages of the planning.Generally is reduces the error occurring.

   For better understanding I will take singleton pattern and will see the affect on project with and without a pattern.

Without a singleton pattern	With singleton pattern

class FoodLogger {
constructor() {
this.foodLog = []
}
 	
log(order) {
 	
this.foodLog.push(order.foodItem)
}
}
 	
// this is the singleton
class FoodLoggerSingleton {
constructor() {
if
(!FoodLoggerSingleton.instance) {
FoodLoggerSingleton.instance =
new FoodLogger()
	const FoodLogger =
	require('./FoodLogger')
	
	const foodLogger = new
	FoodLogger().getFoodLoggerInstance()
	
	class Customer {
	constructor(order) {
	this.price = order.price
	this.food = order.foodItem
	foodLogger.log(order)
	}
	
	module.exports = Customer

```
        }
      }

      getFoodLoggerInstance() {
        return
    FoodLoggerSingleton.instance
      }
    }

module.exports = FoodLoggerSingleton
```

2. What is DTO and explain the use of it.

DTO stands for Data Transfer Object. As in the name itself it is an object that carries data between processes. In technical terms it is an object that is used to encapsulate data, and send it from one subsystem of an application to another. Mostly DTOs use in OOP language environment like Python, C++ and Java.

I will explain the use of DTO using a simple example.

Think that for company purpose one software programmer explains,you might require an worker's name and photo to enter your company.And you need to provide that data for a some other official event,but you don't need to give other information about the employee that you have within your database.Here we can use a DTO to transfer only the information required. And a DTO should just contain data, not business logic.

3. How are you going to store secrets in an application without exposing it to the internet?

Up to my knowledge I think most of application secrets would be different for different environments and it is better to keep them in cloud.Application secrets like passwords,API keys,auth tokens etc required to connect to services like peer microservices, external services, databases, file stores which are in most cases different for your different application environment.

It is not wise that the sensitive data used by any application must be externalised and not be directly used in the code. So more often programmers put the secrets either directly in the code or push their .env or .properties or config files to source control. It is a very bad practice and must be avoided.

Some ways that I think we can mange application secrets are:.
By storing secrets in your source control (GitHub/Bitbucket/GitLab/..), CI/CD tool (Jenkins etc..) or cloud (AWS Secret Manager/Azure Key Vault etc..).

4. What is JWT and how does it work?

JWT stands for JSON Web Token(here JSON means JavaScript object notation). JWT securely transmit information between a client and a server as a JSON object. The details in a JWT can be trusted because it is digitally signed using a secret or public/private key pair. JWT are mainly used for authentication.

And it works as follows

After a user logs in to an application, the application will create a JWT and send it back to the user. Subsequent requests by the user will include the JWT. The token tells the server what routes, services, and resources the user is allowed to access. JWT can be easily used across multiple domains so they are often used for Single Sign On. This is simple how a JWT works.

5.What is the difference between SQL and NoSQL databases?

The main key difference between them is SQL databases are table based databases whereas NoSQL databases can be document based, key-value pairs, graph databases.

Some other differences are:

- ✓ SQL databases are vertically scalable while NoSQL databases are horizontally scalable.
- ✓ SQL databases have a predefined schema whereas NoSQL databases use dynamic schema for unstructured data.

Examples:

SQL is the standard language for dealing with Relational Databases such as:

MySQL Database, Oracle, Ms SQL Server etc.

Some popular NoSQL databases used are:

MongoDB,Apache CouchDB,DynamoDB etc.

6. Suggest a good state management for frontend application and explain why you recommend it.

During past few years some state management alternatives implemented.
FLUX (a facebook invention)and REDUX are the most popular.
Among these I think REDUX is a good state management library.

First of all we will see What is a State Management?It is a method of managing the state.
A state is a representation of a system in a given time. State refers to the data stored in Application in the form of a string, array, object, etc. As an application grows, complexity of managing the state increases as well. In a big application where we have a good number of views/components, managing their state is a big pain.
I recommend REDUX because it is easy to understand ,use and also it follows 2 state management principles well. immutability and Unidirectional data flow.Immutable mean we cannot change. An immutable object is an object whose state cannot be modified after it is

created. If you want to modify some property of an object you have to do it on a copy of the object. The second principle is Unidirectional data flow. It is also known as one-way data flow, which means the data has one, and only one way to be transferred to other parts of the application.

A state management works as follows:

- state is passed to the view and to child components
- actions are triggered by the view
- actions can update the state
- the state change is passed to the view and to child components

And also REDUX can use for React,Angular or for VUE.