بسم الله الرحمن الرحیم

Solid Principle

اصولی هستند که برای توسعه ی شی گرا در زبان های برنامه نویسی استفاده می شوند و هیچ تفاوتی ندارد که از چه زبانی در حال استفاده است. اصولی هستند که باعث می شود برنامه هایی قابل توسعه پیاده کنیم

:Single-responsibility principle اصل تک وظیقه ای

هر کلاسی باید یک تنها یک وظیفه داشته باشد و نه بیشتر.

این مورد می تواند درباره ی توابعی که در داخل javascript استفاده می کنیم صدق کند به این صورت که هر تابع javascript باید تنها یک کار انجام دهد یا به اصطلاحی در js هر بلاکی که داریم اعم از useEffect , functions , … باید تنها یک کار کنند.

Open-closed principle :

این اصل بیان کننده ی این است که کلاس ها باید طوری باشند که برای توسعه دادن باز ولی برای تغییر کردن بسته باشد. یعنی برای افزودن ایتمی یا موردی به یک کلاس نیازی نباشد که کلاس را دستکاری کنیم ولی بتوانیم آن فیچر را با راحتی اضافه کنیم. یعنی مثلا چند نوع متغیر داریم و برای تایپ کردن آن ها نیازی نباشد که مدام if or else بزنیم که اگر از این مدل بود فلان کارکن و اگر از اون مدل بود بیسار کن. برای رعایت این اصل باید مثلا به ازای هر نوع یک interface داشته باشیم که هر کدام ان متد تایپ را داشته باشد و تنها کافی باشدکه متد هر نوع مخصوص به خود را فراخوانی کنیم.

Liskov substitution principle :

اصلی است برای ارث بری.

این اصل بیان می کند که در ارث بری کلاس فرزند باید بتواند تمام رفتار های پدرش را به همان صورت انجام دهد. یعنی اگر در کلاس پدر یک متدی داشته باشیم که return کند یک int ای را. اگر همان متد را در فرزند overwrite کنیم باید int را برگرداند و نه string,object ,…. فقط و فقط همان چیزی که پدر در حال return کردن است.

Interface segregation principle :

اصل تفکیک interface ها : کلاینت ها نباید محبور شوند به واسطه هایی وابسته شوند که از انها استفاده نمی کنند. و همچنین باید حواسمان باشد که interface ها را با هم قاطی نکنیم و اگر چند interface داشته باشیم که کار های مشترکی را انجام بده باید حواسمان باشد که objectای که بهش پاس می دهیم از نوع درستی باشد.

به بیان ساده تر : اینترفیسی نسازید که بعضی از متد های اون را هر جایی استفاده نکنیم

یعنی یک متدی داشته باشیم که در یک کلاس استفاده بشه و در یک کلاس استفاده نشه. همین باعث خطا در نرم افزار میشود.

Dependency inversion principle :

وارونگی وابستگی ها. یعنی یک module سطح بالا نداشته باشیم که به module سطح پایین وابستگی داشته باشه.