**شیوه نامه طراحی قبل از کد زدن برای سبک شی گرایی**

**ابتدا یک توضیح خوب برای اپ مان باید بنویسیم**

**آنچه را که برنامه شما انجام می دهد را توصیف کنید**

**این بدان معناست که برنامه شما چه چیزی را به عنوان ورودی دریافت می کند و چه چیزی را برمی گرداند**

**سپس با نگاه کردن به اسم ها در توضیحاتمان میتونی ابجکت هارو بسازی**

**project title:**

**پرداخت صورتحساب برق برای یک واحد مسکونی ک دو هم خانه دارد و بایت تعداد روزهای حضورشان باید صورت حساب برق را پرداخت نمایند**

**Title: Flatmates Bill**

**Description: an app get that as input amount of a bill for a particular period and**

**and the days that each of the flatmates stayed in the house for that period**

**and returns how much each flatemate has to pay.It also generates a PDF report**

**stating the names of the flatmates, the period, and how each of them had to pay.**

**- با مشخص کردن اسم ها در توضیحات ابجکت های ک باید بسازیم را پیدا میکنم.**

**- سپس از خودمان میپرسیم مثلا app میتونه یک ابجکت برای ما باشه؟**

**- باید داشتنش منطقی باشه.**

**- از میان اسم ها اتربیوتهای امکان پیدا شدن دارند.**

**- مثلا مقدار از صورت حساب یعنی مقدار یک اتربیوت از صورت حساب میتونه باشه.**

**- انگار واقعا به نظر میرسه مقدار یک صفت از صورت حساب هست.**

**- چرا؟ چون وقتی صورت حساب میخواد بوجود بیاد یه مقدارش نگاه میکنیم.**

**- به این شکل باید دقت کنیم ک نیازهای هر کلاس باید دیده شود مثلا فایل PDF نیاز به نام فایل دارد**

**- یا متد برای اجرا به کدام دادهای کلاس های قبلی دارد؟**

**- اکشن های مورد نظر هم متد های کلاسهایمان میشوند، استفاده کننده نیاز به چه اکشنی دارند؟چه محاسباتی احتمالی باید انجام شود؟ ورودی یک متد میتواند یک کلاس باشد برای استخراج دادها(متغیرها).**

**- پیاده سازی انجام نمیدهیم فقط طراحی انجام میدهیم.**

**Object:**

**Bill:**

**amount**

**peroid**

**Flatmates:**

**name**

**Days\_in\_house**

**Pays(Bill)**

**PDFReport:**

**Filename**

**Generate(flatmate1 , flatmate2, bill)**

**همیشه صد در صد اپ را لازم نیست در اینجا طراحی و پیاده سازی کنیم و امکان دارد اپ با ایده های دیگری هم طراحی شود و مشکلاتی که می تواند اشیاء مورد نیاز شما را تغییر دهد.**

**شما می توانید ایده های دیگری داشته باشید**

**بنابراین ممکن است مجبور شوید یک شی را حذف کنید یا ممکن است نیاز داشته باشید نوع شی دیگری را در برنامه خود اضافه کنید.**

**اما احساس نکنید که خیلی با طراحی مرتبط هستید. همه چیز ممکن است تغییر کند.**

**در زمان پیاده سازی در کلاس ها باید در داک استیرنگ چند چمله کار کلاس ک حاوی چه اطلاعاتی هست توضیح بدهیم.**

**برای ارگومان های تابع اینیشیت از صفت های ک در طراحی بدست اوردیم استفاده میکنیم همینطور برای متد های کلاس از طراحی کمک میگیریم و فقط جایگزاری داریم**

**زمان تولید توضیحات و ابجکتها در طراحی هست در کد فقط باید جال گزاری کنیم با فکر ازاد.**

**با طی کردن مراحل بالا میشود گفت ما اسکلت بندی برنامه مان را طراحی کردیم.**

**خوبی طراحی برنامه با شی گرایی این هست ک در کلاس دیگر کلاس مورد نظر را فرا میخوانیم و با اضافه کردن متد یا اتربیویت ها اون کلاس یه داده مورد نیاز یا خروجی مورد نیاز دسترسی پیدا میکنم. یعنی داده های یک کلاس ک یک جا دیگر به نمونه ساخته شده از کلاس داده شده در جای دیگری برای ما قابل دسترس هست. چون دیگه متغیر سراسری ک وجود ندارد در برنامه شی گرایی پس باید از این روش جلو برویم برای دسترسی به دادها یا خروجی های مورد نظر.**

* **هر کلاس باید یک مسولیت واحد داشته باشد**
* **مراحل تبدیل مسله به کد و کلاس**
* **Design**
* **write empty class**

**برای دسترسی به کارکرد های متد ها یا خروجی های یک کند در داخل کلاس باید برای فراخوانی ان مند مورد نظر از سلف.اسم متد() ستفاده کنیم.**

**برای برنامه های ک اینترفیس دارند باید ابتدا یک ترسیم ابتدایی از قالب کلی برنامه برای قسمت frontend انجام دهیم و سپس فایل design را ایجاد کنیم برای طراحی اسکلت برنامه backend.**

**برای در دسترس قرار دادن مقدار یک متغیر در یک کلاس برای کلاس های دیگه، قبل اسم متغیر باید self بگزاریم.**

**نسخه هاردکد دو عیب دارد: اول، اگر تصمیم به تغییر یک مقدار کدگذاری شده دارید، باید تمام رخدادهای آن مقدار را در کد خود تغییر دهید. این کار خیلی زیاد می شود. اگر به جای آن از متغیری استفاده کرده اید (یعنی غیر کدگذاری شده)، فقط باید مقدار متغیر را تغییر دهید و فرمول ها از آن مقدار استفاده خواهند کرد.** **مقادیر هاردکد را در عبارات وارد نکنید. آنها را در متغیرها ذخیره کنید و از متغیرهای کد خود استفاده کنید.**

**یکی از دلایل اصل در استفاده از کلاس ها این هست ک شما در زمان ارایه خروجی در یک نقطه یا دوکلمه و یک نقطه و دادن ورودی های مورد نظر به متد کلاس خروجی مورد نظر را دریافت میکنی. بسادگی و راحتی بدون دوباره کاری و تعداد خط کد بالا و تنها راه ممکن.**

**برای طراحی فایل دیزاین برای کلاس ها ما نیاز به ابجگت های داریم که بخوایم دادهای اون ابجکت را ذخیره و دوباره نیاز به بازیابی ان داده ها داشته باشم. در غیر این صورت**

**میتوان از ساخت این ابجکت صرف نظر کرد.**

**برای طراحی کلاس ها نیاز هست دقیقا فکر چه ورودی های برای انجام چ کاری نیاز داریم**

**دقیقا روی ورودی های ک دریافت کردیم چ عملیاتی میخواهیم انجام و چه روش یا متدهای برای بدست اوردن خروجی مورد نظر و مور نظر نیازهست.**

**برای ورودی ها باید از اتربیوت های برای دریافت ورودی های مورد نیاز عملکرد درونی کلاس ها استفاده کرد.**

**متد ها عملکرد داخلی کلاس ها را تعریف و انجام میدهند.**

**کلاس ها را میتوان بصورت فایل جداگانه نوشت. خروچی فایل میشود خروجی کلاسمان ک با اون کلاس و دسته بندی از متغییر ها و متدها ک بصورت یکجا و هماهنگ یک خروجی به ما تحویل میدهند.**

**برای پیدا کردن اینکه چه کلاسی باید درست کنیم ابتدا از خود بپرس قرار داده این مورد را دستکاری کنی ک نیاز به ذخیره سازی باشد؟ مثلا ما به دادهی کاربر نیاز نداریم چون قرار نیست داد های کاربر را دستکاری کنیم هدف ارسال یک ایمل به اوس پس ب کلاس ایمل نیاز داریم برای رسیدن هدف.**

**ارث بری:**

**در ارث بری ما کلاس زیر مجموعه مان هست یک از بالاسری خودش.**