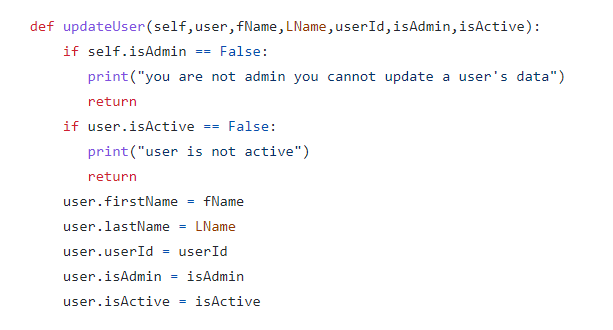
**Design Principles – Contact App**

**Sai Chandana**

1. **DRY (Don’t Repeat Yourself) –** 
   1. No need for an isActive flag and a \_\_isUserActive function, check can be carried out just with flag.
2. **KISS (Keep It Simple Stupid) –** 
   1. Inputting userId/contactId/contactDetailId from the driver code complicates the input, can be auto generated.
   2. Can be made simpler with getattr/setattr
   3. Contacts and Contact details are added with index, something which the client is unaware of. Replace with hooks.
3. **YAGNI (You Aren’t Going To Need It) –**
4. **SOC (Separation of Concerns) -**
5. **BDUF (Build Design up Front) –** File split could have been decided based on the classes in the requirements file. One file for each class and one for the driver code. Reading one big file with a lot of code can be tedious.
6. **TDA (Tell Don’t Ask) -**

Logical Errors –

1. isActive/isAdmin cannot be static since they have different values for each instance of the User class.
2. self.Usercontacts.append(newContact) self.UserContacts is not Declared.
3. Need to write the driver code to figure out more errors

Other errors –

1. Need to follow variable naming conventions

**Praditi**

1. **DRY (Don’t Repeat Yourself) -** Followed
2. **KISS (Keep It Simple Stupid) –** 
   1. In all the user crud operations, an object of the user (performing the action) is passed. Complicates the driver code. Instead, a method can be called on an object to make it simpler.
3. **YAGNI (You Aren’t Going To Need It) -** Followed
4. **SOC (Separation of Concerns)** – Followed
5. **BDUF (Big Design up Front) –** Followed
6. **TDA (Tell Don’t Ask) -** Followed

Logical Errors –

1. Need to write the driver code to figure out more errors.

Other errors –

1. Constructor has operations in it (increment ID)