CSE470 Assignment 2 (Summer 2024)

Please Read Carefully: You will have individually submit the Assignment. Don't share your code with Anyone else. We will follow strict plagiarism policy. However, you can discuss the questions. Please make sure your submitted code is runnable. You are free to choose programming language between Python and JAVA

udoy.saha@g.bracu.ac.bd Switch account Draft saved

* Indicates required question

Email *

✓ Record **udoy.saha@g.bracu.ac.bd** as the email to be included with my response

Student ID

23341134

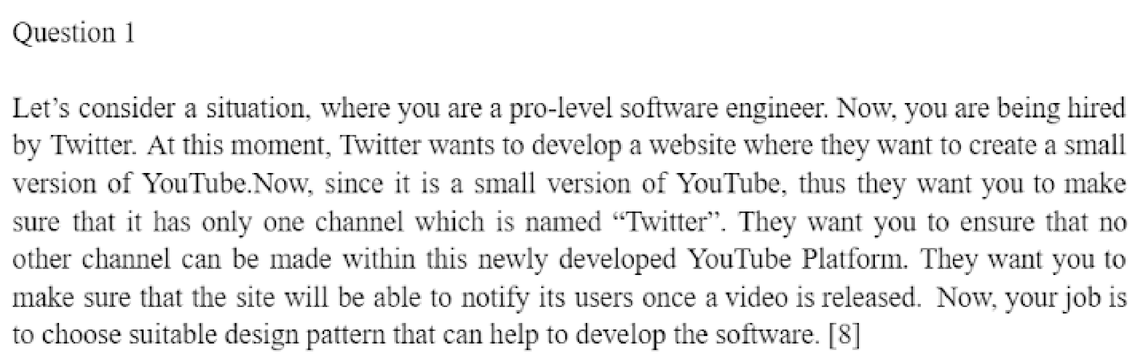
Course section

04

Student Name

Udoy Saha

Please Submit code using either JAVA or Python Programming language and mention your chosen design pattern to solve the scenario below. * 8 points



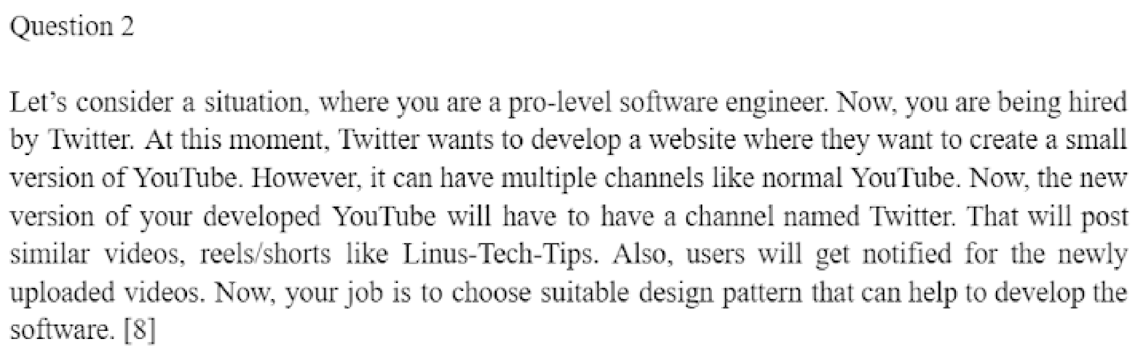
Question 1

Let’s consider a situation, where you are a pro-level software engineer. Now, you are being hired by Twitter. At this moment, Twitter wants to develop a website where they want to create a small version of YouTube.Now, since it is a small version of YouTube, thus they want you to make sure that it has only one channel which is named “Twitter”. They want you to ensure that no other channel can be made within this newly developed YouTube Platform. They want you to make sure that the site will be able to notify its users once a video is released. Now, your job is to choose suitable design pattern that can help to develop the software. [8]

```
// Implemented using SINGLETON + OBSERVER patterns


import java.util.ArrayList;
import java.util.List;


// Youtube interface
interface Youtube {
   void subscribe(User subscriber);
   void unsubscribe(User subscriber);
   void notifySubscribers(String videoName);
}


// Singleton class for Twitter Youtube Channel
class TwitterYoutubeChannel implements Youtube {
   private static TwitterYoutubeChannel channel;
   private static String channelName;
   private List<User> subscribers = new ArrayList<>();

   private TwitterYoutubeChannel() {
   }

   // Get the channel instance
   public static TwitterYoutubeChannel getChannel() {
      if (channel == null) {
         synchronized (TwitterYoutubeChannel.class) {
            if (channel == null) {
               channel = new TwitterYoutubeChannel();
               channel.channelName = "Twitter";
            }
         }
      }
      return channel;
   }

   // Get the channel name
   public String getName() {
      return channelName;
   }

   @Override
   public void subscribe(User subscriber) {
      subscribers.add(subscriber);
      System.out.println(subscriber.getUserName() + " subscribed to " + this.channelName +
" Youtube Channel.");
   }

   @Override
   public void unsubscribe(User subscriber) {
      subscribers.remove(subscriber);
```

```
        System.out.println(subscriber.getUserName() + " unsubscribed from " +
this.channelName + " Youtube Channel.");
    }

    @Override
    public void notifySubscribers(String videoName) {
        for (User subscriber : subscribers) {
            subscriber.update(videoName);
        }
    }
}


// Subscriber interface
interface Subscriber {
    void update(String videoName);
}


// Concrete class for Subscriber
class User implements Subscriber {
    private String name;

    public User(String name) {
        this.name = name;
    }

    @Override
    public void update(String videoName) {
        System.out.println(this.name + " received a notification from " +
TwitterYoutubeChannel.getChannel().getName() + " Youtube Channel: \"" + videoName + "\"
just uploaded!");
    }

    public String getUserName() {
        return name;
    }
}


// Main class
public class Main {
    public static void main(String[] args) {
        TwitterYoutubeChannel channel = TwitterYoutubeChannel.getChannel();

        User sub1 = new User("Alice");
        User sub2 = new User("Bob");

        channel.subscribe(sub1);
        channel.subscribe(sub2);

        channel.notifySubscribers("Video 1");
    channel.unsubscribe(sub2);
    channel.notifySubscribers("Video 2");
  }
}
```

Please Submit code using either JAVA or Python Programming language and mention your chosen design pattern to solve the scenario below. * 8 points



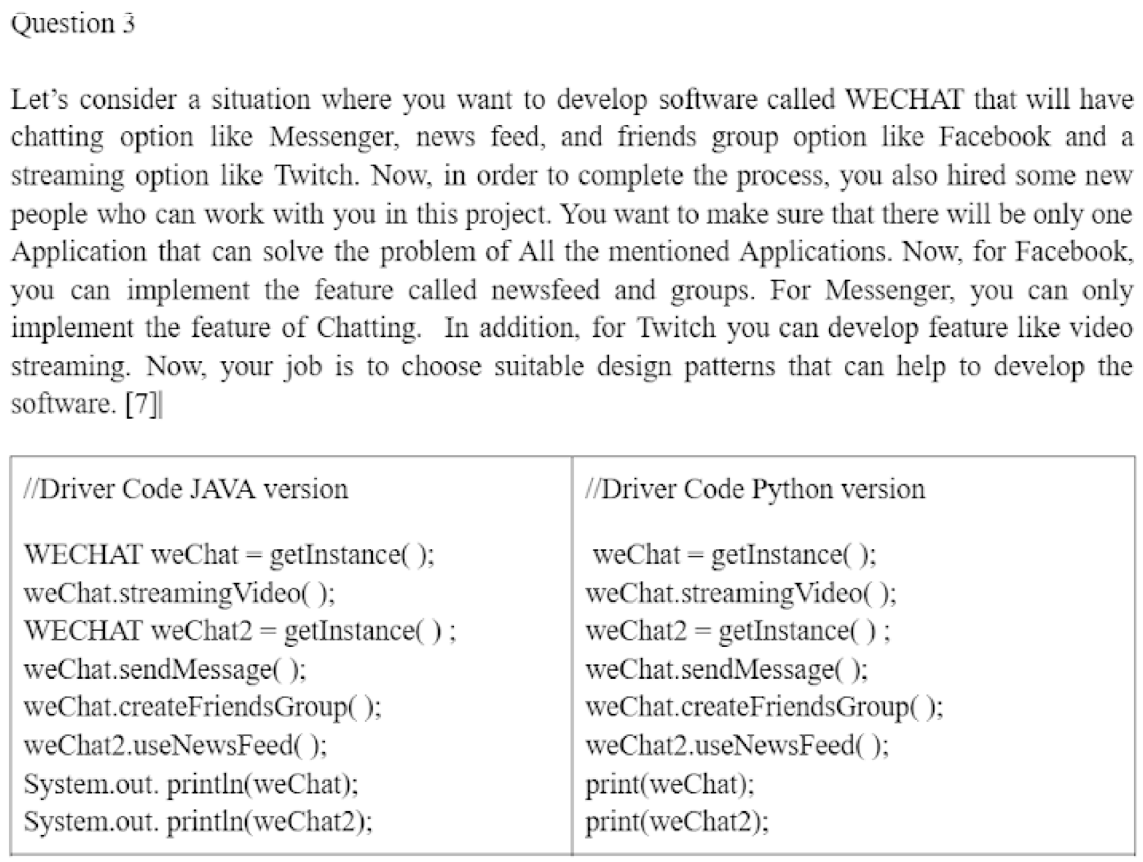
Question 2

Let's consider a situation, where you are a pro-level software engineer. Now, you are being hired by Twitter. At this moment, Twitter wants to develop a website where they want to create a small version of YouTube. However, it can have multiple channels like normal YouTube. Now, the new version of your developed YouTube will have to have a channel named Twitter. That will post similar videos, reels/shorts like Linus-Tech-Tips. Also, users will get notified for the newly uploaded videos. Now, your job is to choose suitable design pattern that can help to develop the software. [8]

```
// Implemented using partial SINGLETON (eager way) + OBSERVER patterns


import java.util.ArrayList;
import java.util.List;


// Youtube interface
interface Youtube {
   void subscribe(User subscriber);
   void unsubscribe(User subscriber);
   void notifySubscribers(String videoName);
}


// Concrete class for Youtube Channels
class YoutubeChannel implements Youtube {
   private static YoutubeChannel channelTwitter = new YoutubeChannel("Twitter");

   private String channelName;
   private List<User> subscribers = new ArrayList<>();

   private YoutubeChannel(String name) {
      this.channelName = name;
   }

   // Get the instance of the channel
   public static YoutubeChannel getChannel(String name) {
      if (name.equals("Twitter")) {
         return channelTwitter;
      }
      return new YoutubeChannel(name);
   }

   // Get the channel name
   public String getName() {
      return channelName;
   }

   @Override
   public void subscribe(User subscriber) {
      subscribers.add(subscriber);
      System.out.println(subscriber.getUserName() + " subscribed to " + this.channelName +
" Youtube Channel.");
   }

   @Override
   public void unsubscribe(User subscriber) {
      subscribers.remove(subscriber);
      System.out.println(subscriber.getUserName() + " unsubscribed from " +
this.channelName + " Youtube Channel.");
   }
```

```
    @Override
    public void notifySubscribers(String videoName) {
        for (User subscriber : subscribers) {
            subscriber.update(this.channelName, videoName);
        }
    }
}


// Subscriber interface
interface Subscriber {
    void update(String channelName, String videoName);
}


// Concrete class for Subscriber
class User implements Subscriber {
    private String name;

    public User(String name) {
        this.name = name;
    }

    @Override
    public void update(String channelName, String videoName) {
        System.out.println(this.name + " received a notification from " + channelName + "
Youtube Channel: \"" + videoName + "\" just uploaded!");
    }

    public String getUserName() {
        return name;
    }
}


// Main class
public class Main {
    public static void main(String[] args) {
        YoutubeChannel channel = YoutubeChannel.getChannel("Linus Tech Tips");
        YoutubeChannel twitterChannel = YoutubeChannel.getChannel("Twitter");

        User sub1 = new User("Alice");
        User sub2 = new User("Bob");
        User sub3 = new User("Charlie");

        channel.subscribe(sub1);
        channel.subscribe(sub2);
        twitterChannel.subscribe(sub2);
        twitterChannel.subscribe(sub3);

        channel.notifySubscribers("Video 1");
        twitterChannel.notifySubscribers("Tweet 1");

        twitterChannel.unsubscribe(sub2);

        twitterChannel.notifySubscribers("Tweet 2");
    }
}
```

Please Submit code using either JAVA or Python Programming language and mention your chosen design pattern to solve the scenario below. * 7 points



Question 3

Let's consider a situation where you want to develop software called WECHAT that will have chatting option like Messenger, news feed, and friends group option like Facebook and a streaming option like Twitch. Now, in order to complete the process, you also hired some new people who can work with you in this project. You want to make sure that there will be only one Application that can solve the problem of All the mentioned Applications. Now, for Facebook, you can implement the feature called newsfeed and groups. For Messenger, you can only implement the feature of Chatting. In addition, for Twitch you can develop feature like video streaming. Now, your job is to choose suitable design patterns that can help to develop the software. [7]

//Driver Code JAVA version	//Driver Code Python version

WECHAT weChat = getInstance( );
weChat.streamingVideo( );
WECHAT weChat2 = getInstance( ) ;
weChat.sendMessage( );
weChat.createFriendsGroup( );
weChat2.useNewsFeed( );
System.out. println(weChat);
System.out. println(weChat2);
	weChat = getInstance( );
	weChat.streamingVideo( );
	weChat2 = getInstance( ) ;
	weChat.sendMessage( );
	weChat.createFriendsGroup( );
	weChat2.useNewsFeed( );
	print(weChat);
	print(weChat2);

```java
// Only SINGLETON is used

// Singleton Wechat interface
interface Wechat {
   void useNewsFeed();
   void createFriendsGroup();
   void sendMessage();
   void streamingVideo();
}


// Concrete Wechat class
class WECHAT implements Wechat {
   private static WECHAT service;

   private WECHAT() {
   }

   public static WECHAT getInstance() {
      if (service == null) {
         synchronized (WECHAT.class) {
            if (service == null) {
               service = new WECHAT();
            }
         }
      }
      return service;
   }

   @Override
   public void useNewsFeed() {
      System.out.println("Using news feed");
   }

   @Override
   public void createFriendsGroup() {
      System.out.println("Creating friends group");
   }

   @Override
   public void sendMessage() {
      System.out.println("Sending message");
   }

   @Override
   public void streamingVideo() {
      System.out.println("Streaming video");
   }
}
```

```
// main class
public class Main {
    public static void main(String[] args) {
        WECHAT weChat = WECHAT.getInstance();
        weChat.streamingVideo();
        WECHAT weChat2 = WECHAT.getInstance();
        weChat.sendMessage();
        weChat.createFriendsGroup();
        weChat2.useNewsFeed();
        System.out.println(weChat);
        System.out.println(weChat2);
    }
}
```

Please Submit code using either JAVA or Python Programming language and mention your chosen design pattern to solve the scenario below. * 7 points

Question 4

In Tech-Park, "Magic Sweets" was famous for its magical cakes, crafted solely by Mr. Shaheed, the master baker. His secret was consistency—each cake tasted exactly like the first, and only he was allowed to make them to maintain this perfect standard. When Jamie, a new baker, suggested allowing others to help increase production, Mr. Shaheed explained, “Our success comes from having one source—the master baker—ensuring every cake is consistently perfect, just like the first.” Inspired, Jamie applied this idea to their software system by implementing a design pattern for configuration settings. Just as Mr. Shaheed’s approach kept the cakes consistent, the design pattern ensured the class maintained stable and controlled settings.[7]

```
// Only SINGLETON is used


// Singleton class for MagicSweets Manager
class MagicSweetsManager {
    private static MagicSweetsManager instance;

    private MagicSweetsManager() {
    }

    public static MagicSweetsManager getInstance() {
        if (instance == null) {
            synchronized (MagicSweetsManager.class) {
                if (instance == null) {
                    instance = new MagicSweetsManager();
                }
            }
        }
        return instance;
    }

    public void cookMagicSweets() {
        System.out.println("MagicSweets are being cooked.");
    }

    public void sellMagicSweets() {
        System.out.println("MagicSweets are being sold.");
    }
}



public class Main {
    public static void main(String[] args) {
        MagicSweetsManager manager1 = MagicSweetsManager.getInstance();
        manager1.cookMagicSweets();

        MagicSweetsManager manager2 = MagicSweetsManager.getInstance();
        manager2.cookMagicSweets();

        System.out.println(manager1);
        System.out.println(manager2);
    }
}
```

Submit Clear form