Tìm vi phạm SRP và OCP của SOLID trong codebase

1. Vi phạm SRP:
   - Class PaymentController: phần xử lý ngày thẻ hết hạn nên được tách riêng ra để class chỉ xử lý phần thanh toán

```
private String getExpirationDate(String date) throws InvalidCardException {
    String[] strs = date.split(regex:"/");
    if (strs.length != 2) {
        throw new InvalidCardException();
    }

    String expirationDate = null;
    int month = -1;
    int year = -1;

    try {
        month = Integer.parseInt(strs[0]);
        year = Integer.parseInt(strs[1]);
        if (month < 1 || month > 12 || year < Calendar.getInstance().get(Calendar.YEAR) % 100 || year > 100) {
            throw new InvalidCardException();
        }
        expirationDate = strs[0] + strs[1];

    } catch (Exception ex) {
        throw new InvalidCardException();
    }

    return expirationDate;
}
```

   - Class PlaceOrderController: phần validate các thông tin giao hàng nên được tách riêng ra để class chỉ làm nhiệm vụ placeOrder

```java
public void validateDeliveryInfo(HashMap<String, String> info)
        throws InterruptedException, IOException, InvalidDeliveryInfoException {
    if (validatePhoneNumber(info.get(key:"phone"))
            || validateName(info.get(key:"name"))
            || validateAddress(info.get(key:"address")))
        return;
    else
        throw new InvalidDeliveryInfoException();
}

public boolean validatePhoneNumber(String phoneNumber) {
    if (phoneNumber.length() != 10)
        return false;
    if (!phoneNumber.startsWith(prefix:"0"))
        return false;
    try {
        Integer.parseInt(phoneNumber);
    } catch (NumberFormatException e) {
        return false;
    }
    return true;
}

public boolean validateName(String name) {
    if (Objects.isNull(name))
        return false;
    String patternString = "^[a-zA-Z\\s]*$";
    Pattern pattern = Pattern.compile(patternString);
    Matcher matcher = pattern.matcher(name);
    return matcher.matches();
}
```

- Class ApplicationProgrammingInterface: đang làm quá nhiều nhiệm vụ, nên được tách ra thành các nhiệm vụ nhỏ hơn như: thực hiện GET và POST, thiết lập kết nối HTTP, cho phép kết nối (allow methods)

```java
vi phạm SRP: nên tách thành 3 nhiệm vụ riêng biệt: CRUD, thiết lập connection và cho phép kết nối
public class ApplicationProgrammingInterface {

    public static DateFormat DATE_FORMATTER = new SimpleDateFormat(pattern:"yyyy/MM/dd HH:mm:ss");
    private static Logger LOGGER = Utils.getLogger(Utils.class.getName());

    public static String get(String url, String token) throws Exception {
        LOGGER.info("Request URL: " + url + "\n");
        HttpURLConnection conn = setupConnection(url);

        conn.setRequestMethod(method:"GET");
        conn.setRequestProperty(key:"Authorization", "Bearer " + token);
        BufferedReader in = new BufferedReader(new InputStreamReader(conn.getInputStream()));
        String inputLine;
        StringBuilder respone = new StringBuilder(); // ising StringBuilder for the sake of memory and performance
        while ((inputLine = in.readLine()) != null)
            System.out.println(inputLine);
        respone.append(inputLine + "\n");
        in.close();
        LOGGER.info("Respone Info: " + respone.substring(start:0, respone.length() - 1).toString());
        return respone.substring(start:0, respone.length() - 1).toString();
    }

    public static String post(String url, String data) throws IOException {
        allowMethods(...methods:"PATCH");
        HttpURLConnection conn = setupConnection(url);
        conn.setRequestMethod(method:"PATCH");
        String payload = data;
        LOGGER.info("Request Info:\nRequest URL: " + url + "\n" + "Payload Data: " + payload + "\n");

        Writer writer = new BufferedWriter(new OutputStreamWriter(conn.getOutputStream()));
        writer.write(payload);
        writer.close();
        BufferedReader in;
```

2. Vi phạm OCP:
    - Các class sử dụng field dạng CreditCard hoặc tham số dạng CreditCard đều vi phạm OCP do nhu cầu tương lai của ứng dụng là thêm các loại thẻ khác như DebitCard (phải sửa lại code).
    - Giải pháp: tạo interface PaymentCardType với các phương thức và thuộc tính chung của các loại thẻ. Tạo class DebitCard và sửa CreditCard để cả 2 class này đều implements PaymentCardType. Các class vi phạm OCP trước đó thay vì sử dụng CreditCard thì sẽ sử dụng dạng PaymentCardType.

```java
public interface InterbankInterface {

    /**
     * Pay order, and then return the payment transaction
     *
     * @param card     - the credit card used for payment
     * @param amount   - the amount to pay
     * @param contents - the transaction contents
     * @return {@link PaymentTransaction PaymentTransaction} - if the
     *         payment is successful
     * @throws PaymentException      if responded with a pre-defined error code
     * @throws UnrecognizedException if responded with an unknown error code or
     *                               something goes wrong
     */
    public abstract PaymentTransaction payOrder(CreditCard card, int amount, String contents)
            throws PaymentException, UnrecognizedException;

    /**
     * Refund, and then return the payment transaction
     *
     * @param card     - the credit card which would be refunded to
     * @param amount   - the amount to refund
     * @param contents - the transaction contents
     * @return {@link PaymentTransaction PaymentTransaction} - if the
     *         payment is successful
     * @throws PaymentException      if responded with a pre-defined error code
     * @throws UnrecognizedException if responded with an unknown error code or
     *                               something goes wrong
     */
    public abstract PaymentTransaction refund(CreditCard card, int amount, String contents)
            throws PaymentException, UnrecognizedException;

}
```