This is the fourth of five assignments that you will complete over the course of the semester:

1: Requirements Draft (10% of homework grade)
2: Final Requirements and Requirement-Based Tests (25%)
3: Design Draft (15%)
4: Final Design and Implementation (25%)
5: Testing (25%)

Each assignment is graded over a series of categories. You will be judged on a scale of 1-4 for each criterion, where a 1 corresponds to a 60%, a 2 corresponds to 75%, a 3 corresponds to 90%, and a 4 corresponds to 100%. If there is no work for a criterion or it is clear that even a minimal amount of effort was not put in, you will receive a 0% for that section of the assignment.

The following is a tentative idea of what we are looking for in Assignment 4. This may change before final grading, but gives criteria to aim for with your submission. A "4" in a category requires all requested elements to be present. Missing elements will result in a lower grade.

Peer Evaluation (5%)

Updated Structural Design (20%):

- Overall design
    - Extensible OO design that is clearly capable of providing the requested functionality.
    - High cohesion and low coupling.
    - All interfacing with BILL is through a defined API. Access is controlled, and proper privacy and scoping is maintained.
    - Customized Exceptions
- Class Diagram
    - Properly formed UML.
    - External files and systems should not be present in class diagram.
- Justification and Explanation
    - VERY IMPORTANT to justify and explain your design. Must show that different options were considered and why/how group arrived at final design. Must demonstrate understanding of OO principles.
    - Automatic maximum of 2 on this section if no justification present.
- Class Descriptions
    - Level of detail is sufficient. Is this implementable by another team?

Dynamic Design (20%):

- Sequence Diagrams
    - Properly formed UML
    - Named instances, not just class names, in boxes.
    - Life lines and activation boxes present

    - Actor present
    - Calls and returns properly labeled
- Diagram description present and understandable.

Code Style (15%):
Based on a random sampling of the source code, we are looking at:
● Consistent bracketing and tab/spacing style
● Descriptive variable names
● JavaDocs present and used correctly
● Sufficient comments to understand code

Missing any one results in -1 to score for that section.

Runtime Behavior (40%):
- Passes a series of test cases, executed through the API.
    - Tests will correspond to each available feature - such as viewing a bill or applying payments.
    - Tests will exercise both standard behavior and error cases (i.e., invalid dates or payment amounts)