219114/115 Programming I

OOP assignment 1

You are given a working library management system written in procedural style. Your task is to refactor this code into an object-oriented design. The code is provided at the following link.

https://gist.github.com/parujr/bdba06216e2cffd9a8bdf7 5f41853554

What to do:

- Refactor the code above into an object-oriented design. Your solution should include these required classes:
  Book Class
  - Attributes: id, title, author, total copies, available copies
  - Methods: Methods to manage book state

  Member Class
  - Attributes: id, name, email, borrowed books list
  - Methods: Methods to manage borrowing

  Library Class
  - Attributes: collections of books and members
  - Methods: add books, add members, borrow, return, and display operations

- During the development, you must use Git version control throughout. At least there must be the following commits:
  - Initial commit for the correct procedural code
  - Second commit when one class is developed and tested
  - Third commit when the second class is developed and tested
  - Fourth commit when the third class is developed and tested
  - Final commit for testing the integration of all classes
    - Use code similar to that used to test the procedural version

- You may want to organize your project using the following file structure:

```
library-management-oop/
│
├── README.md                 # This file
│
├── procedural_version/
```

```
│   ├── library_procedural.py       # Original procedural code
│   └── test_procedural.py          # Comprehensive test suite
│
├── oop_solution/
│   ├── library_oop.py              # Student's OOP implementation (to
create)
│   └── test_oop.py                 # Tests for OOP version (to create)
```

You can stage the one whole folder using:

```
# Stage only this folder
git add folder_name/
```

Or stage everything using:

```
# Or stage all changes everywhere
git add .
```

And, finally, committing:

```
# Commit
git commit -m "Messages for the nth commit"
```

Submission

Once you are done with the development locally:

- Create a README.md file. Here's what it includes:
  - Project Overview - Clear overview description of what this project is about
  - Project Structure - How files are organized (see above)
  - Design Overview - Detailed explanation of each class, detailing attributes and key methods
  - Testing - Test Coverage/ Test suite (test_oop.py) includes:
    - Basic Operations
      - Adding books and members
      - Borrowing and returning books
      - Displaying information

    - Edge Cases
      - Borrowing unavailable books
      - Exceeding borrowing limit
      - Returning books not borrowed

        - Non-existent books/members
    - How to run your test code
- Make the final commit that includes this README.md file
- Push your local repository to your Github remote repository
- Send the link to your Github repository to Google Classroom for grading