Kata

Kata: Library Management System

Objective

Create a simple library management system that allows users to perform basic operations such as adding books, borrowing books, returning books, and viewing available books.

Requirements

1. **Add Books:**
   - Users should be able to add new books to the library.
   - Each book should have a unique identifier (e.g., ISBN), title, author, and publication year.
2. **Borrow Books:**
   - Users should be able to borrow a book from the library.
   - The system should ensure that the book is available before allowing it to be borrowed.
   - If the book is not available, the system should raise an appropriate error.
3. **Return Books:**
   - Users should be able to return a borrowed book.
   - The system should update the availability of the book accordingly.
4. **View Available Books:**
   - Users should be able to view a list of all available books in the library.

Instructions

1. **Code Only:**
   - This is a code-only kata. Focus on writing clean, maintainable code and implementing the required features. Do not spend time creating any user interface.
2. **Test-Driven Development (TDD):**
   - Write tests before implementing the functionality. Follow the three laws of TDD
   - Ensure that all tests pass before considering the implementation complete.
   - Aim for high test coverage and meaningful test cases.
3. **Clean Coding Practices:**
   - Write clean, readable, and maintainable code.
   - Follow SOLID principles and other best practices in software design.
   - Ensure the code is well-documented with meaningful comments and clear variable/method names.

4. **Git Usage:**
    - Use Git for version control.
    - Create a Git repository for your project.
    - Commit your changes frequently with meaningful commit messages to show your TDD journey.
    - Push your code to a remote repository (e.g., GitHub, GitLab, Bitbucket) and share the repository link.

Deliverables

1. A Git repository link containing the source code for the library management system.
2. A README file explaining how to set up and run the project.
3. A test report showing the results of the test cases.

Sample Steps and Git Workflow

1. **Initialize Git Repository:**

Shell

```shell
git init

```

2. **Create Initial README:**

Shell

```shell
echo "# Library Management System" > README.md
git add README.md
git commit -m "Initial commit with README"

```

3. **Write Tests for Adding Books:**
    - Create a new test file (e.g., `test_library.py`).
    - Write a test for the add book feature.
    - Commit your changes:

Shell

```shell
    git add .
    git commit -m "Add test for adding books"

```

4. **Implement Adding Books:**

- Implement the add book functionality.
- Ensure the test passes.
- Commit your changes:

Shell

```shell
git add .
git commit -m "Implement add book feature"

```

5. **Repeat Steps 3-4 for Borrowing Books, Returning Books, and Viewing Available Books:**
6. **Push to Remote Repository:**

Shell

```shell
git remote add origin <remote-repository-url>
git push -u origin master

```

Implementation Guidelines

Languages: Implement the system in one of the following languages:

- Ruby
- Java
- TypeScript
- Python

Assessment Focus:

- Adherence to TDD principles.
- Frequency and quality of Git commits.
- Clean coding practices.
- Proper use of language-specific features and idioms.

Good luck, and happy coding!