Spring 2025 Cloud Native Application Development HW2 Report

Leonard Tsai B10705010

April 13, 2025

1 Project Overview

The Math CLI Application is a command-line interface tool that provides basic mathematical operations. The application is built with Python and follows modern software development practices including unit testing, continuous integration, and code coverage analysis.

2 Technical Framework

2.1 Development Environment

- Python 3.8+
- pytest for unit testing
- GitHub Actions for CI/CD
- Codecov for code coverage reporting

2.2 Project Structure

```
math_cli_app/
  math_cli_app/
      __init__.py
      addition.py
      factorial.py
      odd_even.py
      main.py
  unit_test/
      __init__.py
      test_addition.py
      test_factorial.py
      test_odd_even.py
  setup.py
  requirements.txt
```

3 Core Functions

3.1 Addition Function

```python
def add_numbers(a: float, b: float) -> float:
    """Add two numbers together."""
    return a + b
```

Features:

- Supports floating-point numbers
- Handles positive and negative numbers
- Returns sum as float

3.2 Factorial Function

```python
def calculate_factorial(n: int) -> int:
    """Calculate factorial of a non-negative integer."""
    if n < 0:
        raise ValueError("Factorial not defined for negative numbers")
    if n == 0 or n == 1:
        return 1
    return n * calculate_factorial(n - 1)
```

Features:

- Handles non-negative integers
- Raises ValueError for negative inputs
- Optimized for n = 0 and n = 1

3.3 Odd/Even Function

```python
def check_odd_even(n: int) -> str:
    """Check if a number is odd or even."""
    return "Even" if n % 2 == 0 else "Odd"
```

Features:

- Works with integers
- Returns "Even" or "Odd" as string
- Handles negative numbers

4 Testing Framework

4.1 Unit Tests

The project includes comprehensive unit tests for all functions:

- Addition tests: positive numbers, negative numbers, zero, floats
- Factorial tests: positive integers, zero, error cases
- Odd/Even tests: positive/negative integers, zero

4.2 Continuous Integration

GitHub Actions workflow includes:

- Automated testing on Python 3.8, 3.9, and 3.10
- Code coverage reporting
- Integration with Codecov

5 Issues and Feature Requests

5.1 Current Issues

1. Initial setup of unit tests
2. Integration of code coverage reporting
3. Implementation of CI/CD pipeline

5.2 Implemented Solutions

1. Created comprehensive test suite
2. Configured GitHub Actions workflow
3. Set up Codecov integration

6 Future Enhancements

- Additional mathematical functions
- Enhanced error handling
- Performance optimizations
- User interface improvements

7 Security Considerations

- Input validation for all functions
- Error handling for edge cases
- Secure CI/CD pipeline configuration