Project 4: Time Interval Checker

# Design & Implementation

## Overview

The goal of this project was to develop a JavaFX-based GUI application that allows users to comp[are two time intervals and check whether a specific time falls within either or both of the intervals. The application accepts user input in the format “HH:MM AM/PM”, performs validation, and displays the results in a clear interactive interface.

## Design Approach

The program consists of the following main components:

* Interval<T extends Comparable<T>): A generic class that defines an immutable time interval and provides logic to check containment, overlap, and subinterval relationships.
* Time: A class that represents a 12-hour time and implements Comparable<Time> to support interval logic.
* InvalidTime: A custom checked exception thrown when invalid times are input.
* Project4: The JavaFX launcher that loads the FXML-defined GUI.
* PrimaryController: JavaFX controller responsible for handling user interaction event logic, and output rendering.
* Project4.fxml: Defines the layout of the GUI using a table-like structure and GridPane.

## OOP Principles

* **Encapsulation:** All instance variables are private, and access is controlled via constructors and methods.
* **Modularity:** Each class is responsible for one function. Business logic is separated from the GUI logic.
* **Exception Handling:** The use of InvalidTime ensures that bad input is gracefully handled and reported.
* **Reusability:** The generic Interval class is usable with any Comparable type, not just Time.

## Implementation Summary

The layout was designed first in project4.fxml, using a GridPane to create a table-style format. Users can enter start/end times for two intervals and a single time to check. The controller, PrimaryController.java, reads these inputs, constructs Time and Interval<Time> objects, and handles output generation and error handling.

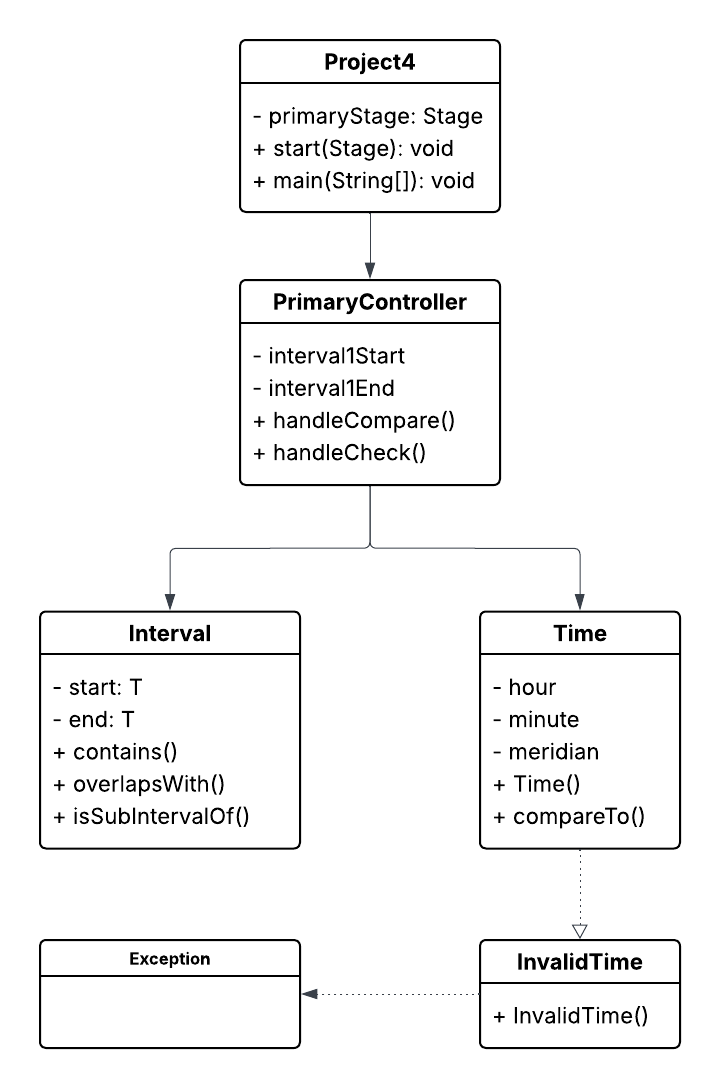
Custom validation ensures hours are between 1 and 12, minutes are between 0 and 59, and the meridian is either AM or PM. If invalid input is detected, an InvalidTime exception is thrown with an appropriate message.

# Lessons Learned

This project reinforced my understanding of the following:

* How to model custom data types like Time using Comparable interface.
* The power of generics in designing reusable interval logic.
* How to structure JavaFX applications using FXML, controllers, and clean separation of GUI/business logic.
* Debugging and validating user input using exception handling
* Refreshing and dynamically testing layouts in Visual Studio Code with launch.json and JavaFX SDK integration.

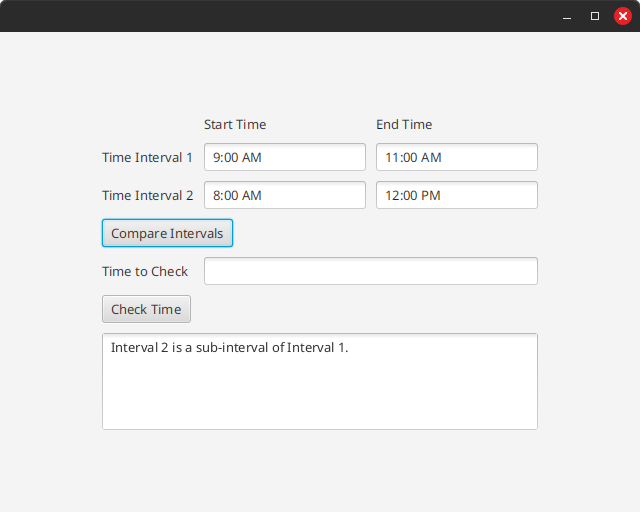
# UML Diagram



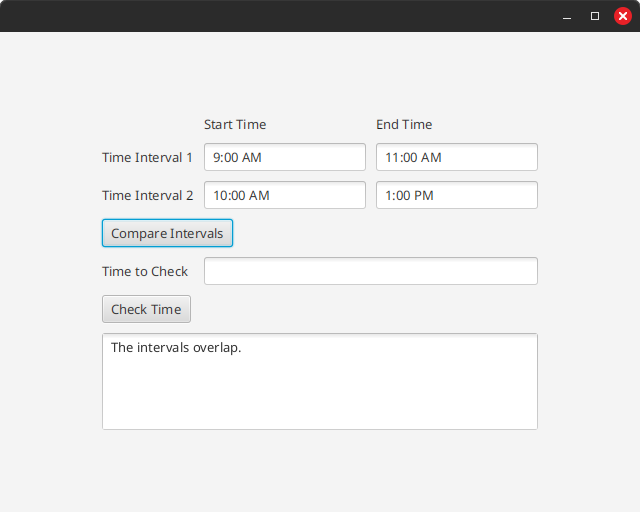
# Test Plan

| **Test Case** | **Description** | **Input** | **Expected Output** | **Status** |
| --- | --- | --- | --- | --- |
| TC01 | Subinterval Check | Interval1: 9:00 AM–11:00 AM, Interval2: 8:00 AM–12:00 PM | Interval 2 is a sub-interval of Interval 1. | Pass |
| TC02 | Overlapping Intervals | Interval1: 9:00 AM–11:00 AM, Interval2: 10:00 AM–1:00 PM | The intervals overlap. | Pass |
| TC03 | Disjoint Intervals | Interval1: 1:00 PM–2:00 PM, Interval2: 3:00 PM–4:00 PM | The intervals are disjoint | Pass |
| TC04 | Check Time: within both | Time: 10:00 AM, Intervals: 9:00 – 11:00 AM & 8:00 AM – 12:00 PM | Both intervals contain the time 10:00 AM. | Pass |
| TC05 | Check Time: within only one | Time: 9:00 AM, Intervals: 9–11 AM & 11–1 PM | Only Interval 1 contains the time 09:00 AM. | Pass |
| TC06 | Invalid format | Interval1: 13:00 PM - 3:00 PM, Interval2: 9:00 AM - 11:00 AM, Time to Check: 10:00 AM | Invalid time: Hour must be between 1 and 12. | Pass |
| TC07 | Empty input | Time to Check: {empty} | Invalid time: Time string cannot be null or empty. | Pass |
| TC08 | Invalid meridian | Time to Check: “10:00 XM” | Invalid time: Meridian must be 'AM' or 'PM'. | Pass |

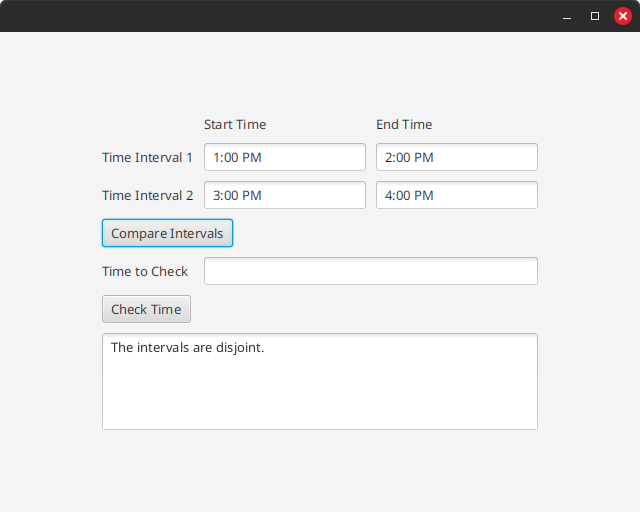
## TC01: Subinterval Check



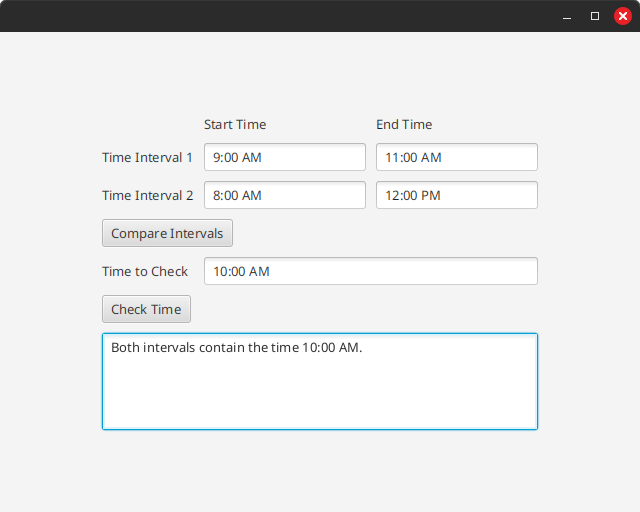
## TC02: Overlapping Intervals



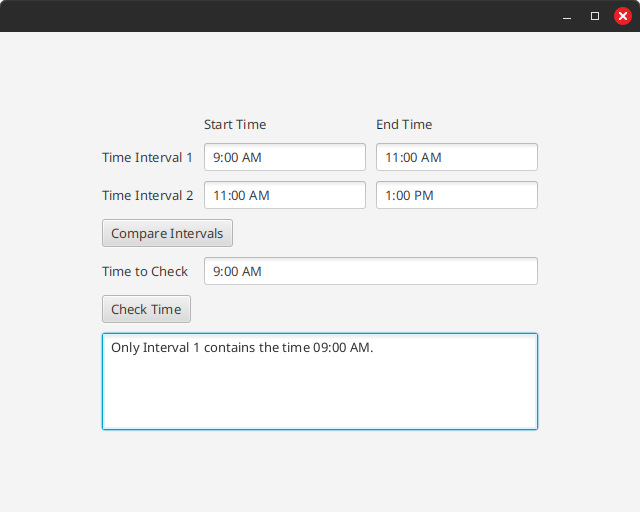
## TC03: Disjoint Intervals



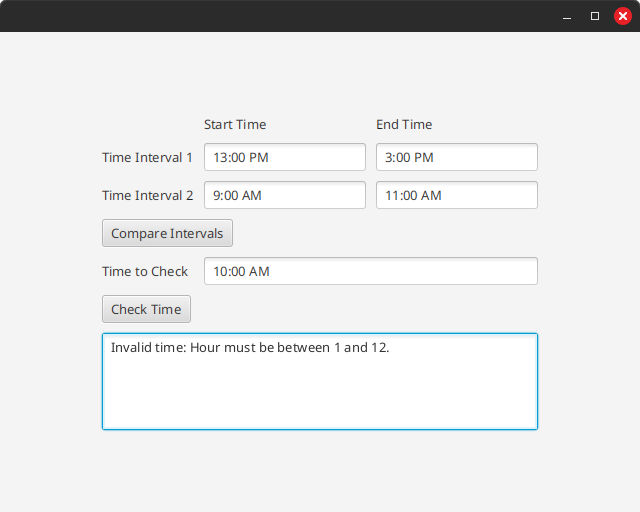
## TC04: Check Time Within Both



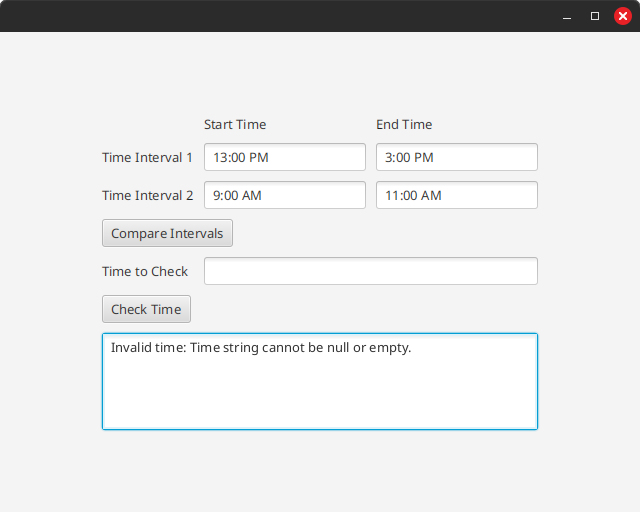
## TC05: Check Time within Only One



## TC06: Invalid Format



## TC07: Empty Input



## TC08: Invalid Meridian

