**Assignment: Expand the Traffic Light Simulator to Model a Four-Way Intersection**

**Objective:** The goal of this assignment is to enhance the existing traffic light simulator to accurately represent a four-way intersection. The current implementation supports only one traffic light. Your task is to modify the data structures and main state machine logic to accommodate the additional three sides of the intersection, ensuring proper synchronization without turn lanes.

**Requirements:**

1. **Intersection Model:** Redesign the current `TrafficLight` struct to represent one side of the intersection. You will then need to create a new struct, `Intersection`, that contains four instances of `TrafficLight`, one for each direction (North, South, East, West).

2. **State Machine Logic:** The state machine should now manage the states of all four traffic lights in a synchronized manner. Only traffic lights facing each other should be green at the same time (e.g., when North and South are green, East and West should be red, and vice versa).

3. **Data Structure Changes:** The `TrafficLightState` enum may need additional states or flags to indicate whether a particular direction is allowed to go or not. Alternatively, you could create a new enum to represent the state of the `Intersection` as a whole.

4. **State Transition Rules:** Define clear rules for state transitions that consider the traffic flow in all four directions. Ensure there is a safe transition period, such as a yellow light, before the direction of traffic flow changes.

5. **UI Update:** Modify the UI to visually represent the four-way intersection. Each traffic light should have its own display indicating the current state. The UI should update in real-time as the state machine transitions between states.

6. **User Interaction:** Implement a UI control that allows the user to manually cycle through the states of the intersection or put the simulator in an automatic mode, where it transitions states on a timed interval.

**Deliverables:**

1. **Code:** Submit the Rust source files containing your implementation of the `Intersection` and updated `TrafficLight` structs, along with the modified state machine logic.

2. **UI Demonstration:** A gif or video of the UI showing the four-way intersection cycling through all its operational states.

3. **Documentation:** A README file documenting your design decisions, how you have synchronized the traffic lights, and instructions on how to run the simulator.

**Evaluation Criteria:**

- **Functionality:** The simulator must accurately model the state of a four-way intersection, with appropriate synchronization between the traffic lights.

- **Code Quality:** The code should be well-organized, with clear separation between the data model and the UI. Data structures should be designed to effectively represent the states and transitions.
- **Usability:** The UI should be intuitive, allowing users to easily understand the current state of the intersection and interact with the simulator controls.

**Hints:**

- Consider using a finite state machine library if managing the states and transitions becomes complex.
- Pay attention to Rust's ownership and borrowing rules when sharing state between the intersection and the UI.

**Bonus Challenge (Optional):**

- Introduce a failure mode where one or more traffic light becomes stuck in a particular state, and implement a way to detect and handle this in the simulator.

**Submission Deadline:** [Insert deadline]

Upon completion, submit your code repository link, along with the required deliverables, to [insert submission email or upload link].