Reviewer: Eyad Ghanem **Team reviewed: Iron**

- **Does the design model (all diagrams) use the correct syntax? List what you think is wrong and suggest how to improve things.**

Great work, However, in the sequence diagrams, there are areas where syntax might need enhancement. Some classes, especially those representing stages, should have clearer definitions. Instead of leaving types undefined or merely as a “type” attribute, each attribute should include a clear data type. I suggest double-checking that every diagram adheres to the standard UML conventions. Also, adjust sequence diagrams by streamlining lifeline labels and using a consistent numbering or vertical ordering method. Ensure conditions are uniformly worded (e.g., “License is Invalid” and “Race ID is Invalid”) and eliminate any stray artifacts.

In the domain class model, some attributes are listed without clearly defined data types. Also, some relationships could be labeled specifically. For example, “must have” or “can view” could be replaced by “has” or “views”

- **Does the design model use the correct level of abstraction? If so, why? If not, where do you think the group could improve their design?**

In my opinion, as an overall, the design model maintains a decent level of abstraction. The use cases capture high-level system functionality, and the domain model correctly describes key entities and their relationships without diving into unnecessary implementation details. The sequence diagrams outline the main interactions without exposing overly detailed logic.

However, I suggest that within the signUpForRace sequence diagram, the inclusion of a specific "404 search error" may be exposing an implementation detail that could be abstracted into a more general error-handling approach.

So, I suggest to abstract error handling to a higher level (for example, using a generalized “Error in Search” condition) to ensure the diagram remains at the desired abstraction level.

- **Are the diagrams consistent and do they form one cohesive design model?**

I can say that the diagrams generally is consistent. There is consistency between the use case diagram, the domain model, and most of the sequence diagrams. One noted inconsistency is in ordering within the sequence diagrams. for example, I think that the signUpForRace process is better to be reversed its structure by first checking if the license is valid, then going to further conditions (e.g., whether the race is full) and ending with the registration success. Also, some names vary across diagrams.

- **Does the overall design model fulfill the requirements from Deliverable 1/3?**

I think that the design covers the core functionalities starting from race registration, license management, review handling, race organization, and account management. The domain model, however, does not seem to clearly represent variations in race types (e.g., OfficialRace vs. InofficialRace) or the management of different stage types. Instead, race type is handled as a simple attribute.

- **Based on the above, make suggestions on how to improve the overall design.**

➢ Improving overall syntax including clear attribute definitions, standardized multiplicity, and cleaned-up sequence diagram flows will make the design easier to understand and evaluate.

- Maintain a consistent level of abstraction across all diagrams. While details are good, it is better to avoid including too many implementation specifics (such as specific error codes) in high-level interaction flows.

- **Does the Deliverable document contain all the information needed (based on the kickoff document)?**

In my opinion, the deliverable document contains the necessary details as outlined in the kickoff document.

- **Overall Opinion about the Design and Valuable Takeaways**

In general, the design is well done. high-level functionality is well described in use case model. The domain model presents key actions clearly.

Generally, the project deliverable 1 is very good. With improvements to the UML syntax, a more consistent layout in sequence interactions, and enhanced representation for race and stage types, the design model can be perfect.

- **Did you see anything in the design that might be valuable for your own team's design?**

I like the details and the scenarios for sequence diagram which is not the same in our team's deliverable. Also, inclusion of operations in class diagram is better.