Try to compile/use the source code provided. Can you get it up and running? Is anything problematic?
All ok.

Test the runnable version of the application in a realistic way. Note any problems/bugs.
All ok.

Does the implementation and diagrams conform (do they show the same thing)? Are there any missing relations? Relations in the wrong direction?
No dependency arrows to/from enum GameEvent. If it's added to class diagram then we think it would be interesting to know how it connects to other classes.

Is the dependency between controller and view handled? How? Good? Bad?
Dependency just hidden by putting it in a enum class, since the controller is using int input = m_view.GetInput(); the controller is still dependent on console app as view.

Is the Strategy Pattern used correctly for the rule variant Soft17?
Pattern is used correctly but not correctly implemented. The soft17 requires checking if hand has Ace or not.

Is the Strategy Pattern used correctly for the variations of who wins the game?
The strategy pattern is used however the logic for Winner may not necessarily return who is actually the winner, as the strategies only check if the dealer has higher score than the player, for example; return a_dealer.CalcScore() >= a_player.CalcScore();

- Now there is also code used in Dealer to determine if the dealer is under 21, this makes it impossible to implement a wins rule where dealer would have 22 for instance, creating a limitation in the flexibility of rule design.
- However this depends on your view in terms of to what extent rules should be modified, since busting above 21 tends to be viewed as quite "universal".

Is the duplicate code removed from everywhere and put in a place that does not add any dependencies (What class already knows about cards and the deck)? Are interfaces updated to reflect the change?
Used almost exactly the same solution as we did to remove code duplications. However missed to remove duplication in InternationalNewGameStrategy.
Dependencies in interfaces that are no longer needed after doing refactoring (AddCard) are remaining in the code.

Is the Observer Pattern correctly implemented?
Yes.

Is the class diagram updated to reflect the changes?
Yes.

Other comments:

There’s dead code by commenting it out. Rather have it removed.

Do you think the design/implementation has passed the grade 2 criteria?
No.