Here are the changes we made:

1. **Model**: We've created sub-packages that have only relevant model classes inside of them. For example, the 'tile' sub-package only contains classes like 'Crate', Portal', 'Jewel', etc. We've reverted from our previous design with 'BasicTile' and 'MovableTile' to a more robust segregation of interface. For this, we added the following types of interfaces - Placeable, Turnable, Pairable, Undoable, and tiles implement the required interface(s). For example, a Jewel (or StoneWall) tile implements Placeable since it can only be placed on the board, but cannot be turned left or right or paired with another tile. Similarly, a Turtle tile implements Placeable, Turnable, and Undoable as it can be placed on the board and turned left and right and its moves can be undone. Only the Portal tile implements the Pairable interface, which allows it to be paired with another Portal tile.

   Previously, playing any of our cards resulted in the turtle moving around in the board, which was made easier by an IMove interface that was implemented by Forward, LeftTurn, RightTurn, and Bug classes. And, our IMove interface had a method that accepted a Turtle tile and manipulated its position or direction based on the move selected. So, with the introduction of a Laser card, we added a new interface IShoot, that is implemented by the Laser class and has a method that accepts an IceWall tile and manipulates its position (e.g. sets it to null).

   We've split our main model (Game) class into Game, GameBoardInfo and GameManipulate classes based on responsibility. Our GameBoardInfo class sends all information about the tiles present in the Game to the ManipulateModel class in the controller. Similarly, the GameManipulate class handles updating information about the game board, or checking card validity, and checking for game completion. And, our Game class now only deals with initialization of the game and has basic getters and setters.

2. **Controller:** We got rid of our somewhat less useful segregation of our controller from our previous milestone and opted for splitting the main controller into two classes - RobotTurtleController, which deals with the main game logic, i.e., starting the game and playing the game until the game is won, and ManipulateModel, which gets information about tiles, and makes it available for the View to use.

3. **View:** Our View remained mostly the same, except we added characters for representing the tiles added. Here are the characters used to represent the game tiles.

Turtle	↑, ↓, →, ←

Jewel	J
StoneWall	S
Portal	⬬
IceWall	*
Crate	□

We decided to use UTF symbols because they were more representative of the Tile than alphabets.