SCENARIO / ACTION	Test Description	EXPECTED OUTPUT	ACTUAL OUTPUT	P/F	REMARKS
COMPILE THE PROJECT	This is to check that the program has no syntax/compiler errors.	Program compiles using "javac" with JavaFX arguments with no extra compiler messages.	Program compiles using "javac" with JavaFX arguments with no extra compiler messages.	✓	
RUN THE PROGRAM	This is to check that the program opens up its GUI as expected with no issues.	Program opens the main window and is presented with the title with background image, three menu buttons.	Program opens the main window and is presented with the title with background image, three menu buttons.	✓	
OPEN THE SETTINGS WINDOW (CLICK THE SETTINGS BUTTON IN THE MAIN MENU)	This is to check that the program can open the settings window with no issues, which also demonstrates the connection of the view and controller.	Program opens the settings window and is presented with the settings sliders and the back button.	Program opens the settings window and is presented with the settings sliders and the back button.	✓	
PRESS THE BACK BUTTON FROM THE SETTINGS WINDOW WITHOUT CHANGING SLIDERS AND GO BACK TO THE SETTINGS WINDOW	This is to check that data is retained from the window.	Slider position when exiting the settings window must be the same open entering the next time.	Slider position when exiting the settings window must be the same open entering the next time.	✓	
MAKE SOME CHANGES TO BOTH SLIDERS IN THE SETTINGS WINDOW, EXITING THE SETTINGS WINDOW, AND RE-ENTERING	This is to check that data is retained from the window.	Slider position when exiting the settings window must be the same open entering the next time.	Slider position when exiting the settings window must be the same open entering the next time.	✓	
MAKE CHANGES TO ONLY ONE SLIDER, EXIT, THEN RETURN	This is to check that data is retained from the window.	Slider position when exiting the settings window must be the same open entering the next time.	Slider position when exiting the settings window must be the same open entering the next time.	✓	
MAKE CHANGES TO ONLY THE OTHER SLIDER, EXIT, THEN RETURN	This is to check that data is retained from the window.	Slider position when exiting the settings window must be the same open entering the next time.	Slider position when exiting the settings window must be the same open entering the next time.	✓	
MAKE CHANGES TO BOTH SLIDERS, EXIT, THEN RETURN	This is to check that data is retained from the window.	Slider position when exiting the settings window must be the same open entering the next time.	Slider position when exiting the settings window must be the same open entering the next time.	✓	
OPEN THE NEW GAME DIALOG FROM THE MAIN MENU	This is to check that the program can correctly open the new game scene window.	The new game scene window is opened, presenting the user with two buttons, three textboxes, and one drop-down combobox.	The new game scene window is opened, presenting the user with two buttons, three textboxes, and one drop-down combobox.	✓	
GO BACK FROM THE NEW GAME DIALOG TO THE MAIN MENU VIA THE BACK BUTTON	This is to check that the program can correctly go back to the previous scene from the new game window.	Main menu opens	Main Menu opens	✓	Data entered in the new game scene is not saved in state.
CHOOSE 2 PLAYERS FROM THE COMBOBOX IN THE NEW GAME MENU	This is to check that the first two textboxes will be enabled.	The first two textboxes are enabled.	The first two textboxes are enabled.	✓	
CHOOSE 3 PLAYERS FROM THE COMBOBOX IN THE NEW GAME MENU	This is to check that the all three textboxes will be enabled.	All three textboxes are enabled.	All three textboxes are enabled.	✓	
LEAVE ALL THREE TEXTBOXES EMPTY IN THE NEW GAME SCENE	This is to check that the CREATE GAME button remains disabled as long as not all enabled textboxes are populated.	CREATE GAME button remains disabled.	CREATE GAME button remains disabled.	✓	
POPULATE ONE OUT OF THREE TEXTBOXES IN THE NEW GAME SCENE (No. players = 3)	This is to check that the CREATE GAME button remains disabled as long as not all enabled textboxes are populated.	CREATE GAME button remains disabled.	CREATE GAME button remains disabled.	✓	
POPULATE TWO OUT OF THREE TEXTBOXES IN THE NEW GAME SCENE (No. players = 3)	This is to check that the CREATE GAME button remains disabled as long as not all enabled textboxes are populated.	CREATE GAME button remains disabled.	CREATE GAME button remains disabled.	✓	
POPULATE THREE OUT OF THREE TEXTBOXES IN THE NEW GAME SCENE (No. players = 3)	This is to check that the CREATE GAME button remains disabled as long as not all enabled textboxes are populated.	CREATE GAME button is enabled.	CREATE GAME button is enabled.	✓	
POPULATE ONE OUT OF TWO TEXTBOXES IN THE NEW GAME SCENE (No. players = 2)	This is to check that the CREATE GAME button remains disabled as long as not all enabled textboxes are populated.	CREATE GAME button remains disabled.	CREATE GAME button remains disabled.	✓	
POPULATE TWO OUT OF TWO TEXTBOXES IN THE NEW GAME SCENE (No. players = 2)	This is to check that the CREATE GAME button remains disabled as long as not all enabled textboxes are populated.	CREATE GAME button is enabled.	CREATE GAME button is enabled.	✓	
CREATE GAME (2 players)	This is to check that the model is instantiated properly.	Main game window is opened and no exceptions have been printed to console.	Main game window is opened and no exceptions have been printed to console.	✓	

CREATE GAME (3 players)	This is to check that the model is instantiated properly.	Main game window is opened and no exceptions have been printed to console.	Main game window is opened and no exceptions have been printed to console.	✓	
MODEL INSPECTION (3 PLAYERS)	This is to check that everything is initialized properly.	1. All three player pegs are in start position and the sidebar has the correct player names. 2. Correct startup information ($200k balance, no loans, no house, not married, no children) are shown in the player status sidebar. 3. There are no career, salary, and transaction information about all players. 4. It is currently player 1's turn (turn box is filled red)	1. All three player pegs are in start position and the sidebar has the correct player names. 2. Correct startup information ($200k balance, no loans, no house, not married, no children) are shown in the player status sidebar. 3. There are no career, salary, and transaction information about all players. 4. It is currently player 1's turn (turn box is filled red)	✓	
MODEL INSPECTION (2 PLAYERS)	This is to check that everything is initialized properly.	1. Both player pegs are in start position and the sidebar has the correct player names. 2. Correct startup information ($200k balance, no loans, no house, not married, no children) are shown in the player status sidebar. 3. There are no career, salary, and transaction information about all players. 4. It is currently player 1's turn (turn box is filled red) 5. Player 3's sidebar panel is blank and disabled.	1. Both player pegs are in start position and the sidebar has the correct player names. 2. Correct startup information ($200k balance, no loans, no house, not married, no children) are shown in the player status sidebar. 3. There are no career, salary, and transaction information about all players. 4. It is currently player 1's turn (turn box is filled red) 5. Player 3's sidebar panel is blank and disabled.	✓	
MOVEMENT	This is to check that the model returns the correct player space and the controller reflects it to the view.	When a new space is assigned to the player, after his turn, his peg is removed from the old space and transferred to the new space, following AnchorPane formatting rules.	When a new space is assigned to the player, after his turn, his peg is removed from the old space and transferred to the new space, following AnchorPane formatting rules.	✓	
ORANGE SPACE & ACTIONCARD	This is to check that the action card has its effects on the player.	When a player lands on an orange space, an action card is drawn and the event is executed. There is visible change in the player (or other players)'s balance.	When a player lands on an orange space, an action card is drawn and the event is executed. There is visible change in the player (or other players)'s balance.	✓	
MAGENTASPACE	This is to check that player movement is interrupted by MagentaSpace.	When a player passes through a MagentaSpace, the player stops moving, regardless of number of moves left. The MagentaSpace event is executed.	When a player passes through a MagentaSpace, the player stops moving, regardless of number of moves left. The MagentaSpace event is executed.	✓	
JUNCTIONSPACE	This is to check that the player is prompted to choose paths when coming across a junction space.	When a player passes through a junction space, he is halted and prompted to choose path. Once path is chosen, the player starts moving in that path.	When a player passes through a junction space, he is halted and prompted to choose path. Once path is chosen, the player starts moving in that path.	✓	
GREENSPACE (PayDay)	This is to check that the player is credited his salary when he lands on a green space of type pay day.	When a player lands on a payday green space, the amount of his salary is added to his balance.	When a player lands on a payday green space, the amount of his salary is added to his balance.	✓	
GREENSPACE (PayRaise)	This is to check that the player is given a pay raise upon landing on a green space on type pay raise.	When a player lands on a payraise green space, his salary increases by the amount of the salary card's pay raise value.	When a player lands on a payraise green space, his salary increases by the amount of the salary card's pay raise value.	✓	
BLUESPACE & BLUE CARD	This is to check that blue card events are executed when the player lands on a blue space.	When a player lands on a blue space, careers are checked and debited accordingly. The unique blue card event is then executed.	When a player lands on a blue space, careers are checked and debited accordingly. The unique blue card event is then executed.	✓	
ENDSPACE	This is to check that player data and balance are consolidated and finalized when the player lands here.	When the player lands here, his loans are paid off, children value are added, investments are paid out.	When the player lands here, his loans are paid off, children value are added, investments are paid out.	✓	

SAVE GAME	This is to check that the instance of the game is saved onto a serialized binary file.	The instance of the entire game is serialized into a binary file.	The instance of the entire game is serialized into a binary file.	✓	Static parameters such as the static counts are not included in serialization.
LOAD GAME	This is to check that the instance of the game is loaded and deserialized properly when loading from a binary file.	The game window is opened with the exact same state from when the game was saved.	The game window is opened with the exact same state from when the game was saved.	✓	The serialized game must be serialized from the same game version, otherwise, reading exceptions may occur. MessagePrompt text is not replicated.
PLAYER STATUS AND DATA	This is to check that the sidebar is updated every move.	The sidebar reflects changes on player after every move. (for an entire game)	The sidebar reflects changes on player after every move. (for an entire game)	✓	