# System Requirements

## Non-Functional Requirements

This section outlines all of the non-functional requirements of the project which define the general constraints that apply to the development process as well as the final product. This includes programming languages and software tools of choice.

| **nfr-1** | **Non-Functional Requirement 1** |
|---|---|
| Java will be the main programming language of choice throughout this project, it will be used wherever practical and possible. Specifically we will be using Java 8, as this is the latest version supported by NetBeans.[1]<br><br>**Justification:** Our entire team is familiar with Java, having learnt it in our first year of study. Furthermore, the Java Virtual Machine is supported on all major operating systems, including Windows and macOS.[2] | |

| **nfr-2** | **Non-Functional Requirement 2** |
|---|---|
| JavaFX will be the GUI framework of choice for presenting the interface of the game. Specifically we will be using JavaFX 8.<br><br>**Justification:** Since Java is the programming language of choice there are only two first party options with regard to GUI frameworks: Swing/AWT or JavaFX. Our entire team is familiar with Swing/AWT, having used it previously, however since Swing/AWT is an aging API, which JavaFX was designed specifically to replace, we have opted to use the more modern framework. | |

| **nfr-3** | **Non-Functional Requirement 3** |
|---|---|
| NetBeans will be our IDE of choice throughout this project.<br><br>**Justification:** NetBeans is the official IDE for Java. Again, our entire team is familiar with it and it has the added bonus of having built in support for Git version control.[3] | |

[1] See nfr-3
[2] See nfr-7
[3] See nfr-4

| **nfr-4** | **Non-Functional Requirement 4** |
|---|---|
| We will use version control to collaboratively manage all files related to this project. Git and GitHub will be the version control system of choice for developing this project.<br><br>**Justification:** The justification for using version control in general is the ease it provides for collaboratively working on a set of files, whilst also reducing the risk of redundant or missing data becoming a problem. The justification for using Git and GitHub is the fact that it is industry standard and centralised. | |

| **nfr-5** | **Non-Functional Requirement 5** |
|---|---|
| The game will be *'configuration-customisable'* meaning that aspects of the game, including but not limited to board layout and cards, can be customised by editing configuration files which are loaded at startup.<br><br>Any *configuration-customisable* aspects of the game will not be modifiable from within the running game (for example via a settings dialog), any aspects of the game that are modifiable from within the running game shall be referred to as *runtime-customisable* throughout this document.<br><br>**Justification:** Configuration-customisability is a desirable requirement supplied by our client in order to make the game easily modifiable. Whilst such a feature will be a challenge to implement in a flexible manner, it will allow us to react to any potential requirements changes more easily, with less modifications to executable code. | |

| **nfr-6** | **Non-Functional Requirement 6** |
|---|---|
| XML will be the 'configuration' language of choice for this project. 'Configuration' includes the client provided board and card data, as well as any other persistent parameters that need to be customisable between sessions, this includes user settings.<br><br>**Justification:** The justification for using XML as a configuration language is its familiar syntax, as our whole team has worked with HTML previously. Furthermore, Java provides a powerful XML processing and binding API, that should hopefully allow us to streamline the process of loading configuration data. | |

| **nfr-7** | **Non-Functional Requirement 7** |
|---|---|
| The game will, at minimum, run on all modern Windows desktop and laptop machines. We also aim to make the game work on modern machines running macOS if possible.<br><br>**Justification:** Our client has requested that the electronic version of the game work on desktop machines, ideally for both the Mac and PC, but if both are not possible then a PC only version is preferable. | |

| nfr-8 | Non-Functional Requirement 8 |
|---|---|
| The game will be designed in an object-oriented and modular manner. This means our code will take advantage of object-oriented services such as encapsulation, abstraction, inheritance and polymorphism.<br><br>**Justification:** Designing the game in this manner will allow the game to be more easily maintainable and modifiable both pre and post deployment, and as a result should hopefully help us to more easily react to changing client requirements. | |

| nfr-9 | Non-Functional Requirement 9 |
|---|---|
| We aim to have a ‘responsive’ user interface. Specifically our user interface should always quickly respond to input, even during lengthy operations such as I/O. If an action cannot be performed immediately without a ‘visual pause’ then some form of visual feedback (such as a progress bar) shall be provided indicating that the application is working on the requested task.<br><br>Quantitatively ‘responsive’ means the user interface shall take no longer than 100ms to respond to user input, any task that takes longer than this to complete will execute concurrently or in parallel. This will be achieved with the use of techniques such as multithreading and asynchronous programming where necessary. | |

| nfr-10 | Non-Functional Requirement 10 |
|---|---|
| We aim to deliver a ‘reliable’ application, free of bugs, especially ones that would cause the game to ‘crash’ or exit unexpectedly. Quantitatively we aim for a 2-5% ‘error rate’, or in other words the game will work as expected at least 95% of the time.<br><br>**Justification:** Although we would like a perfectly functioning product with a 0% ‘error rate’ given the time constraints and non safety-critical nature of the product 95% reliability is a much more achievable goal. | |

| **nfr-11** | **Non-Functional Requirement 11** |
|---|---|
| The game should be stored as a collection on files on a user's/player's computer. The collection of files will likely include a Java Archive (JAR) (.jar), that contains the executable portion of the game, as well as various XML (.xml) configuration files.<br><br>The entire game can be packaged into a single JAR file for the purpose of distribution, in order to facilitate easy installation. Any other files can be packaged as resources within the JAR and saved to the users disk when the game is first run.<br><br>**Justification:** JARs are the standardised method for distributing Java applications. JARs not only contain executable code required to run an application, but can also contain any resource files that support the application. Furthermore JARs can be compressed, meaning they are suitable for distribution over the Web. | |

# Functional Requirements

## Game Rules

The functional requirements in this section correspond the game rules of section three in the User Requirements document.

Some of these requirements are also drawn from other sections of the User Requirements document where relevant to gameplay mechanics.

| **ur-gr-1** | **User Requirement Game Rule 1** |
|---|---|
| The game is for two to six players. Each player is assigned one of the game tokens. The tokens are: boot, smartphone, goblet, hatstand, cat and spoon. Each player takes a turn by rolling two dice to determine how they move around the board. At the outset, all players start in the board space labelled Go and move clockwise around the board. | |
| **fr-gr-1-1** | **Functional Requirement Game Rule 1.1** |
| Type | Mandatory |
| Description | The game **shall** support two to six players. |
| **fr-gr-1-2** | **Functional Requirement Game Rule 1.2** |
| Type | Desirable |
| Description | The game **should** potentially be able to support a higher number of players in the future. This should be a low priority requirement[4] |

[4] See ur-cq-1

| | |
|---|---|
| **fr-gr-1-3** | **Functional Requirement Game Rule 1.3** |
| Type | Desirable |
| Description | Minimum and maximum player counts **should** be *configuration customisable*. |
| Detail | `minPlayers = 2`<br>`maxPlayers = 6`<br><br>`minPlayers > 1`<br>`maxPlayers >= minPlayers` |
| **fr-gr-1-4** | **Functional Requirement Game Rule 1.4** |
| Type | Mandatory |
| Description | The game **shall** have a board consisting of 40 positions called tiles. The board **shall** be square in shape, with 10 tiles arranged along each edge and space in the middle for the 'pot luck' and 'opportunity knocks' decks, as well as the 'free parking' money.[5] |
| **fr-gr-1-5** | **Functional Requirement Game Rule 1.5** |
| Type | Mandatory |
| Description | Each player **shall** have a token representing their position on the game board. |
| **fr-gr-1-6** | **Functional Requirement Game Rule 1.6** |
| Type | Mandatory |
| Description | The game **shall** have the following tokens (by default): boot, smartphone, goblet, hatstand, cat and spoon. |
| **fr-gr-1-7** | **Functional Requirement Game Rule 1.7** |
| Type | Desirable |
| Description | The game **should** support *configuration customisable* tokens (appearance and name), including the addition of new tokens.[6] |
| **fr-gr-1-8** | **Functional Requirement Game Rule 1.8** |
| Type | Mandatory |
| Description | The game **shall** ensure there are a minimum of six tokens configured at startup, in order to match the maximum player count. |

[5] See ur-gr-3 and ur-gr-24
[6] See ur-cq-2

| **fr-gr-1-9** | **Functional Requirement Game Rule 1.9** |
|---|---|
| Type | Mandatory |
| Description | Players **shall** start on the position labelled ‘go’. |
| **fr-gr-1-10** | **Functional Requirement Game Rule 1.10** |
| Type | Mandatory |
| Description | Players **shall** move clockwise around the board. |
| **fr-gr-1-11** | **Functional Requirement Game Rule 1.11** |
| Type | Mandatory |
| Description | Players **shall** take turns to move around the board and perform other gameplay actions. |
| **fr-gr-1-12** | **Functional Requirement Game Rule 1.12** |
| Type | Mandatory |
| Description | Players **shall** take turns in a round in descending order of the cumulative face value shown on the dice, after each rolling them. |

| **ur-gr-2** | **User Requirement Game Rule 2** |
|---|---|
| At the outset of the game, each player has £1,500 in cash. One player is designated the banker and is responsible for distributing the correct amount of cash to each player. The bank has a total of £50,000 cash. Players may not borrow additional money from the bank, but they can trade game items with the bank. | |
| **fr-gr-2-1** | **Functional Requirement Game Rule 2.1** |
| Type | Mandatory |
| Description | Players **shall** start with £1,500 in cash. |
| **fr-gr-2-2** | **Functional Requirement Game Rule 2.2** |
| Type | Desirable |
| Description | The game **should** support *configuration customisable* currency and starting cash. |
| Detail | `startingCash = 1500`<br>`currency = ‘£’` |

| **fr-gr-2-3** | **Functional Requirement Game Rule 2.3** |
|---|---|
| Type | Mandatory |
| Description | Players **shall** be able to trade ‘game items’ or ‘assets’ in their possession to the bank in return for cash. ‘Game items’ include properties, houses and hotels. |
| **fr-gr-2-4** | **Functional Requirement Game Rule 2.4** |
| Type | Mandatory |
| Description | ~~Players **shall not** be able to trade or sell ‘game items’ with/to other players, but~~ |
| **fr-gr-2-5** | **Functional Requirement Game Rule 2.5** |
| Type | Desirable |
| Description | ~~The game **should** be able to potentially support player trading in the future. This should be a high priority requirement[7]~~ |
| **fr-gr-2-6** | **Functional Requirement Game Rule 2.6** |
| Type | Mandatory |
| Description | The game **shall** support player trading of properties. Properties **shall** only be tradeable in exchange for other properties. |

| **ur-gr-3** | **User Requirement Game Rule 3** |
|---|---|
| At the outset of the game, the two packs of cards labelled ‘pot luck’ or ‘opportunity knocks’ are shuffled and placed on the board. When cards are taken, they must be replaced at the bottom of the corresponding pile. | |
| **fr-gr-3-1** | **Functional Requirement Game Rule 3.1** |
| Type | Mandatory |
| Description | The game **shall** have two decks of cards named ‘pot luck’ and ‘opportunity knocks.’ |
| **fr-gr-3-2** | **Functional Requirement Game Rule 3.2** |
| Type | Desirable |
| Description | The game **should** support *configuration customisable* deck names. |

[7] See ur-cr-3

| **fr-gr-3-3** | **Functional Requirement Game Rule 3.3** |
|---|---|
| Type | Mandatory |
| Description | The board **shall** have space for these two decks of cards. The cards **shall** appear face down. |
| **fr-gr-3-4** | **Functional Requirement Game Rule 3.4** |
| Type | Mandatory |
| Description | At the start of the game the decks **shall** be shuffled. |
| Detail | `read deck data from config`<br><br>`...`<br><br>`shuffle(deck)` |
| **fr-gr-3-5** | **Functional Requirement Game Rule 3.5** |
| Type | Mandatory |
| Description | Cards **shall** be drawn from the top of their respective pile when required and replaced at the bottom once ‘spent.’ |
| Detail | `draw_card(deck)`<br>`    return card from top of deck`<br><br>`spend_card(card)`<br>`    perform action shown on card`<br><br>`    ...`<br><br>`    // Replaces card at the bottom of its owner deck`<br>`    replace_card(card, card owner deck)` |

| **ur-gr-4** | **User Requirement Game Rule 4** |
|---|---|
| For each turn, the player rolls two fair, six sided dice. They move the number of spaces shown on the dice. Players move clockwise around the board. | |
| **fr-gr-4-1** | **Functional Requirement Game Rule 4.1** |
| Type | Mandatory |
| Description | On a player’s turn the cumulative face value shown on the dice, after rolling them, **shall** determine the number of tiles that player moves around the board. |

| **fr-gr-4-2** | **Functional Requirement Game Rule 4.2** |
|---|---|
| Type | Mandatory |
| Description | The dice **shall** be six sided and fair. |
| **fr-gr-4-3** | **Functional Requirement Game Rule 4.3** |
| Type | Desirable |
| Description | The quantity of and number of sides on the dice **should** be *configuration-customisable.* |
| Detail | `diceCount = 2`<br>`diceSides = 6` |

| **ur-gr-5** | **User Requirement Game Rule 5** |
|---|---|
| If a player throws a double, then they take another turn. If a player throws another double at the third turn, then they ‘go to jail’ when a player goes to jail, the go directly and do not pass go. | |
| **fr-gr-5-1** | **Functional Requirement Game Rule 5.1** |
| Type | Mandatory |
| Description | A ‘double’ **shall** be defined as the all dice having the same face value after being rolled. |
| **fr-gr-5-2** | **Functional Requirement Game Rule 5.2** |
| Type | Mandatory |
| Description | When a player rolls a double they **shall** immediately take a second turn. |
| **fr-gr-5-3** | **Functional Requirement Game Rule 5.3** |
| Type | Mandatory |
| Description | When a player rolls a second double, on a turn taken immediately after rolling a double on the prior turn, they **shall** immediately take a third turn. |
| **fr-gr-5-4** | **Functional Requirement Game Rule 5.4** |
| Type | Mandatory |
| Description | When a player rolls a third double they **shall** ‘go to jail.’ |

| **fr-gr-5-5** | **Functional Requirement Game Rule 5.5** |
|---|---|
| Type | Mandatory |
| Description | When a player ‘goes to jail’ they **shall** move directly to the ‘jail’ position. Moving directly means they do not ‘step’ on or pass through any intermediate tiles. This is significant with regard to the ‘go’ tile, as players will not collect a bonus. |

| **ur-gr-6** | **User Requirement Game Rule 6** |
|---|---|
| Board spaces may consist of properties, a ‘pot luck’ space, an ‘opportunity knocks’ space, ‘free parking, the jail/just visiting space or a space with specific instructions that must be followed by the player. | |
| **fr-gr-6-1** | **Functional Requirement Game Rule 6.1** |
| Type | Mandatory |
| Description | The game **shall** have the following types of board spaces: properties (regular, train stations and utilities), ‘card draw’ (‘pot luck’ or ‘opportunity knocks’), ‘free parking’, ‘jail’/’just visiting’, ‘go’, ‘go to jail’ and ‘taxes’/’fines’. |

| **ur-gr-7** | **User Requirement Game Rule 7** |
|---|---|
| If a player lands on a ‘pot luck’ or ‘opportunity knocks’ space, they take a card from the top of the corresponding pile and carry out the instructions on the card. When this is complete, the card is replaced at the bottom of the corresponding pile.[8] | |

[8] See ur-gr-3

| **ur-gr-8** | **User Requirement Game Rule 8** |
|---|---|
| Players make progress in the game by buying property as they move around the board. Player may not purchase property until they have completed one complete circuit of the board by passing the Go space. When a player passes Go, they receive £200 from the bank. | |
| **fr-gr-8-1** | **Functional Requirement Game Rule 8.1** |
| Type | Mandatory |
| Description | Eligible players[9] **shall** be able to purchase properties that are owned by the bank. |
| **fr-gr-8-2** | **Functional Requirement Game Rule 8.2** |
| Type | Mandatory |
| Description | Players **shall not** be able to purchase any properties until they have made one complete circuit of the game board. |
| **fr-gr-8-3** | **Functional Requirement Game Rule 8.3** |
| Type | Desirable |
| Description | The game **should** support *configuration customisable* toggling of the 'one circuit rule' in 8.2. |
| **fr-gr-8-4** | **Functional Requirement Game Rule 8.4** |
| Type | Desirable |
| Description | The game **should** be able to potentially support *runtime customisable* toggling of the 'one circuit rule' in 8.2. |
| **fr-gr-8-5** | **Functional Requirement Game Rule 8.5** |
| Type | Mandatory |
| Description | A circuit of the game board **shall** be defined as passing 'go' by entering and subsequently exiting the tile. |
| **fr-gr-8-6** | **Functional Requirement Game Rule 8.6** |
| Type | Mandatory |
| Description | When a player passes 'go' they **shall** receive £200 from the bank. |

[9] See fr-gr-8-2

| | |
|---|---|
| **fr-gr-8-7** | **Functional Requirement Game Rule 8.7** |
| Type | Desirable |
| Description | The bonus for passing ‘go’ **should** be *configuration customisable*. |
| Detail | `goCash = 200` |

| | |
|---|---|
| **ur-gr-9** | **User Requirement Game Rule 9** |
| All properties are initially the property of the bank. When a player purchases a property, the card is transferred from the bank to that player and the amount shown on the card is paid to the bank. | |
| **fr-gr-9-1** | **Functional Requirement Game Rule 9.1** |
| Type | Mandatory |
| Description | The bank **shall** initially own all properties. |
| **fr-gr-9-2** | **Functional Requirement Game Rule 9.2** |
| Type | Mandatory |
| Description | When a player purchases a property from the bank[10] ownership of that property (including the property card) **shall** be transferred to the player in exchange for its cash value as shown on the property card. |
| Detail | `purchase_property(player, property)`<br><br>`    ...`<br><br>`    transfer(bank, player, property)`<br>`    transfer(player, bank, property value)`<br><br>`    ...` |

[10] See fr-gr-8-1

| **ur-gr-10** | **User Requirement Game Rule 10** |
|---|---|
| Once a player has made their move, if they land on a property that has not yet been purchased, they have the opportunity to buy that property. If they decide not to buy that property then the property is auctioned by the bank. Each player makes a bid to the bank. The bank sells the property to the highest bidder. If there are no bids, then the property remains unsold. All bidding players must have completed one circuit of the board. | |
| **fr-gr-10-1** | **Functional Requirement Game Rule 10.1** |
| Type | Mandatory |
| Description | When an eligible[11] player lands on a property owned by the bank they **shall** choose whether or not they want to purchase the property.[12] |
| **fr-gr-10-2** | **Functional Requirement Game Rule 10.2** |
| Type | Mandatory |
| Description | When a player chooses not to buy a property it **shall** be auctioned by the bank. |
| Detail | `auction_property(property)` |
| **fr-gr-10-3** | **Functional Requirement Game Rule 10.3** |
| Type | Mandatory |
| Description | ~~All eligible players, including the player who landed on the property, **shall** be able to bid in the auction.~~[13] |
| **fr-gr-10-4** | **Functional Requirement Game Rule 10.4** |
| Type | Mandatory |
| Description | When only the player who landed on the property is eligible to participate in the auction, if they choose not to purchase the property it **shall not** be auctioned. |
| **fr-gr-10-5** | **Functional Requirement Game Rule 10.5** |
| Type | Mandatory |
| Description | ~~Eligible players **shall** take turns to bid on the property, in the same order they take regular turns, starting from the player who would take the next turn.~~[14] |

[11] See fr-gr-8-2
[12] See fr-gr-9-2
[13] See fr-cq-10-1
[14] See ur-cq-9

| **fr-gr-10-6** | **Functional Requirement Game Rule 10.6** |
|---|---|
| Type | Mandatory |
| Description | ~~On their turn, each player **shall** choose to place a higher bid then the current highest bid or to exit the auction.~~[15] |
| **fr-gr-10-7** | **Functional Requirement Game Rule 10.7** |
| Type | Mandatory |
| Description | ~~The auction **shall** end when only one player remains.~~[16] |
| **fr-gr-10-8** | **Functional Requirement Game Rule 10.8** |
| Type | Mandatory |
| Description | Players **shall not** be able to place a bid that exceeds their current amount of cash held. |
| **fr-gr-10-9** | **Functional Requirement Game Rule 10.9** |
| Type | Mandatory |
| Description | ~~The property in question **shall** be transferred to the remaining player by the bank in exchange for the current bid (which is by definition the highest).~~[17] |
| **fr-gr-10-10** | **Functional Requirement Game Rule 10.10** |
| Type | Mandatory |
| Description | Players **shall not** be able to trade assets to raise cash funds during an auction. |

| **ur-gr-11** | **User Requirement Game Rule 11** |
|---|---|
| If a player lands on a property owned by another player, they must pay the player who owns the property the value of the rent shown on the card. | |
| **fr-gr-11-1** | **Functional Requirement Game Rule 11.1** |
| Type | Mandatory |

[15] See ur-cq-9
[16] See ur-cq-9
[17] See ur-cq-9

| | |
|---|---|
| Description | When a player lands on an unmortgaged property owned by another player, they **shall** pay the owner the specified amount shown on the card. |
| **fr-gr-11-2** | **Functional Requirement Game Rule 11.2** |
| Type | Mandatory |
| Description | When a player lands on an owned property that is mortgaged, payments **shall not** be made to the owner of the property. |

| | |
|---|---|
| **ur-gr-12** | **User Requirement Game Rule 12** |
| If a player owns all of the properties in a colour coded group, but the properties are otherwise not developed further with houses and hotels, then the rent due is doubled. | |
| **fr-gr-12-1** | **Functional Requirement Game Rule 12.1** |
| Type | Mandatory |
| Description | Rent due **shall** be doubled when all properties of the same group are owned by the same player and the property landed on has not been improved. |

| | |
|---|---|
| **ur-gr-13** | **User Requirement Game Rule 13** |
| If a property is improved with houses or hotels, then the rent to be paid is as shown on the card. | |
| **fr-gr-13-1** | **Functional Requirement Game Rule 13.1** |
| Type | Mandatory |
| Description | Rent due **shall** be increased to the amount shown on the card in conjunction with the number of houses/hotels on the property. |

| | |
|---|---|
| **ur-gr-14** | **User Requirement Game Rule 14** |
| All rents must be paid for in cash. If a player is unable to pay the rent for a property they have landed on, they must sell game assets to make good on the rent. If they are unable to pay the rent after selling all of their game assets, then they are bankrupt and must leave the game. Their game token is then removed from the board. | |
| **fr-gr-14-1** | **Functional Requirement Game Rule 14.1** |

| | |
|---|---|
| Type | Mandatory |
| Description | A player **shall** pay rent in cash. |
| **fr-gr-14-2** | **Functional Requirement Game Rule 14.2** |
| Type | Mandatory |
| Description | When a player is unable to pay rent in cash, the player **shall** assets to the bank to raise enough cash to pay for rent. |
| **fr-gr-14-3** | **Functional Requirement Game Rule 14.3** |
| Type | Mandatory |
| Description | A player **shall** be declared bankrupt and leave the game when they are unable to pay owed rent after selling all of their game assets. |
| **fr-gr-14-3** | **Functional Requirement Game Rule 14.4** |
| Type | Mandatory |
| Description | If the proceeds of selling all a player's assets to the bank doesn't cover the debt, then those proceeds **shall** be passed to the player owed and the rest of the debt written off. |
| **fr-gr-14-4** | **Functional Requirement Game Rule 14.5** |
| Type | Mandatory |
| Description | A player's token **shall** be removed from the board when they leave the game. |

| **ur-gr-15** | **User Requirement Game Rule 15** |
|---|---|
| Players may not borrow or lend money from each other, and may not borrow money from the bank. | |
| **fr-gr-15-1** | **Functional Requirement Game Rule 15.1** |
| Type | Mandatory |
| Description | A player **shall not** lend or borrow cash. |

| **ur-gr-16** | **User Requirement Game Rule 16** |
|---|---|
| When a player has finished moving their token, and has completed any property purchase activity, they have the option to buy houses and hotels to improve their properties. Players are not permitted to improve their properties at any other time. | |
| **fr-gr-16-1** | **Functional Requirement Game Rule 16.1** |
| Type | Mandatory |
| Description | On their turn a player **shall** first move their token. |
| **fr-gr-16-4** | **Functional Requirement Game Rule 16.4** |
| Type | Mandatory |
| Description | If a player lands on a property that they own no ‘property related action’ **shall** take place, except |
| **fr-gr-16-5** | **Functional Requirement Game Rule 16.5** |
| Type | Mandatory |
| Description | After all other turn activity a player **shall** choose whether or not they would like to purchase houses or hotels for any properties they own.[18] |

| **ur-gr-17** | **User Requirement Game Rule 17** |
|---|---|
| Houses and hotels may only be purchased for properties where a player owns all of the properties in a particular colour coded group. | |
| **fr-gr-17-1** | **Functional Requirement Game Rule 17.1** |
| Type | Mandatory |
| Description | A player **shall** only be permitted to purchase houses/hotels when the player owns all of the properties of a particular colour group. |

[18] See game rules 17 and 20

| **ur-gr-18** | **User Requirement Game Rule 18** |
|---|---|
| Houses and hotels are purchased for the amount shown on the game card. | |
| **fr-gr-18-1** | **Functional Requirement Game Rule 18.1** |
| Type | Mandatory |
| Description | Houses and hotels **shall** be purchased for the amount shown on the game card for the property they are being purchased for. |

| **ur-gr-19** | **User Requirement Game Rule 19** |
|---|---|
| If a player needs to raise funds, they can sell a property back to the bank for its original value as shown on the game card. A property can only be sold when there are no houses or hotels on the property. A player may also sell houses and hotels back to the bank for the original purchase price. | |
| **fr-gr-19-1** | **Functional Requirement Game Rule 19.1** |
| Type | Mandatory |
| Description | A player **shall** be able to sell houses/hotels back to the bank for the original  purchase price (shown on card).[19] |
| **fr-gr-19-2** | **Functional Requirement Game Rule 19.2** |
| Type | Mandatory |
| Description | A player **shall** be able to sell properties back to the bank for the original purchase price, but |
| **fr-gr-19-3** | **Functional Requirement Game Rule 19.3** |
| Type | Mandatory |
| Description | A player **shall not** be able to sell a property with improvements on it unless those improvements are sold as well. |

[19] See game rule 18.

| **ur-gr-20** | **User Requirement Game Rule 20** |
|---|---|
| Where a coloured set of properties is owned and developed by a player, there may never be a difference of more than 1 house between the properties in that set. A hotel is considered to be worth 5 houses. | |
| **fr-gr-20-1** | **Functional Requirement Game Rule 20.1** |
| Type | Mandatory |
| Description | On an owned coloured set, the number of houses on each property **shall not** differ by more than one. A hotel is worth five houses. |

| **ur-gr-21** | **User Requirement Game Rule 21** |
|---|---|
| The maximum development permitted on any one property is one hotel. | |
| **fr-gr-21-1** | **Functional Requirement Game Rule 21.1** |
| Type | Mandatory |
| Description | The maximum development allowed on any one property **shall** be one hotel. |

| **ur-gr-22** | **User Requirement Game Rule 22** |
|---|---|
| If a player needs to raise funds, they may mortgage a property with the bank. The bank will pay the player one half of the value of the property as shown on the game card. No rents may be collected for that property whilst it is under mortgage. | |
| **fr-gr-22-1** | **Functional Requirement Game Rule 22.1** |
| Type | Mandatory |
| Description | A player **shall** be able to mortgage a property to the bank in return for half the value of the property shown in card. |
| **fr-gr-22-2** | **Functional Requirement Game Rule 22.2** |
| Type | Mandatory |
| Description | Rent **shall not** be collected by the owner of the property whilst the property is mortgaged. |

| **fr-gr-22-3** | **Functional Requirement Game Rule 22.3** |
|---|---|
| Type | Mandatory |
| Description | A player **shall not** be able to mortgage an already mortgaged property. |

| **ur-gr-23** | **User Requirement Game Rule 23** |
|---|---|
| If a mortgaged property is then sold back to the bank, it is sold for one half of the property price as shown on the card. | |
| **fr-gr-23-1** | **Functional Requirement Game Rule 23.1** |
| Type | Mandatory |
| Description | A player **shall** be able to sell a mortgaged property back to the bank in return for half the value shown on the property card. |

| **ur-gr-24** | **User Requirement Game Rule 24** |
|---|---|
| Where fines are to be paid, the proceeds accumulate on the free parking space in the centre of the board. When a player lands on free parking, they collect all of the funds currently on the free parking space. | |
| **fr-gr-24-1** | **Functional Requirement Game Rule 24.1** |
| Type | Mandatory |
| Description | When a player pays a fine, the funds **shall** accumulate on the free parking space in the centre of the board. |
| **fr-gr-24-2** | **Functional Requirement Game Rule 24.2** |
| Type | Mandatory |
| Description | When a player lands on free parking they **shall** receive any funds that have accumulated there. |

| **ur-gr-25** | **User Requirement Game Rule 25** |
|---|---|
| If a player is sent to jail, they may pay £50 to be released from jail. The £50 is added to the free parking fines. The player token is then moved to “just visiting” and the players turn ends. The player takes a normal turn in the next round. | |
| **fr-gr-25-1** | **Functional Requirement Game Rule 25.1** |
| Type | Mandatory |
| Description | If a player is sent to jail they **shall** choose whether or not to pay £50 to be released. |
| **fr-gr-25-2** | **Functional Requirement Game Rule 25.2** |
| Type | Mandatory |
| Description | The release fee **shall** be added to the free parking proceeds. |
| **fr-gr-25-3** | **Functional Requirement Game Rule 25.3** |
| Type | Mandatory |
| Description | If they pay the fine they **shall** move into the ‘just visiting’ space and their turn will immediately end. |
| **fr-gr-25-4** | **Functional Requirement Game Rule 25.4** |
| Type | Mandatory |
| Description | After paying the fine a players next turn **shall** be a normal turn. |

| **ur-gr-26** | **User Requirement Game Rule 26** |
|---|---|
| If a player opts to stay in jail, they give up their turn for the next two rounds. Whilst in jail, a player may not collect any rents from other players. At the end of the next two rounds, the player token is moved to ‘just visiting’ and the player’s turn ends. The player takes a normal turn in the next round. | |
| **fr-gr-26-1** | **Functional Requirement Game Rule 26.1** |
| Type | Mandatory |
| Description | Players in jail who choose not to pay the release fine and opt to stay in jail **shall not** take their turn for the next two rounds. |
| Detail | |

```
take_turn(player)

    ...

    if (player is in jail, has been for less than 2 rounds
        and release fine is not paid) then
        do not let player take turn

    ...
```

| **fr-gr-26-2** | **Functional Requirement Game Rule 26.2** |
|---|---|
| Type | Mandatory |
| Description | Players in jail **shall not** collect rents from other players. |
| Detail | |

```
collect_rent(player, property)

    ...

    if (property owner is in jail) then
        do not collect rent from player

    ...
```

| **fr-gr-26-3** | **Functional Requirement Game Rule 26.3** |
|---|---|
| Type | Mandatory |
| Description | Players who have been in jail for the past two rounds **shall** move directly to ‘just visiting’ and then their turn will immediately end. |
| Detail | |

```
take_turn(player)

    ...

    if (player is in jail and has been for the past 2
        rounds) then
        move_direct(player, ‘just visiting’)

    ...
```

| **ur-gr-27** | **User Requirement Game Rule 27** |
|---|---|
| If a player has a ‘get out of jail free’ card, then they place the card at the bottom of the ‘pot luck’ or ‘opportunity knocks’ pile as appropriate, the player token is moved to ‘just visiting’ and the player turn ends. The player takes a normal turn in the next round. | |
| **fr-gr-27-1** | **Functional Requirement Game Rule 27.1** |
| Type | Mandatory |
| Description | Players in jail who have a ‘get out of jail free’ card **shall** spend[20] the card, move directly to ‘just visiting’ and then end their turn immediately. |
| Detail | |

```
take_turn(player)

    ...

    // Player goes to jail for some reason
    move_direct(player, ‘jail’)

    ...

    // Check on the same turn if they have the card
    if (player is in jail and has a get out of jail free
        card) then
        move_direct(player, ‘just visiting’)
        replace get out of jail class in pile

    ...
```

[20] See fr-gr-3-5

| **fr-gr-28-1** | **Functional Requirement Game Rule 28.1** |
|---|---|
| Type | Mandatory |
| Description | The game **shall** support *configuration-customisable* cards (deck, description and action). |
| **fr-gr-28-2** | **Functional Requirement Game Rule 28.2** |
| Type | Mandatory |
| Description | The game **shall** support ‘transaction’ card actions including Bank to Player, Player to Bank, Player to Free Parking, Free Parking to Player transactions. |
| **fr-gr-28-3** | **Functional Requirement Game Rule 28.3** |
| Type | Mandatory |
| Description | The game **shall** support Player to Player transactions to facilitate card actions requiring players to pay other players cash. |
| **fr-gr-28-4** | **Functional Requirement Game Rule 28.4** |
| Type | Mandatory |
| Description | The game **shall** support both forward and direct player movement, to provide both turn and card movement capability. |
| **fr-gr-28-5** | **Functional Requirement Game Rule 28.5** |
| Type | Mandatory |
| Description | Forward movement **shall** involve ‘stepping’ on intermediate positions between the origin and destination positions. This is significant both visually, for the movement animation and functionally for the action of ‘passing go’ and collecting the bonus. |
| **fr-gr-28-6** | **Functional Requirement Game Rule 28.6** |
| Type | Mandatory |
| Description | Direct movement **shall** only involve the source and destination positions, players do not ‘step on’ or otherwise interact with intermediate positions. |
| **fr-gr-28-7** | **Functional Requirement Game Rule 28.7** |
| Type | Mandatory |
| Description | Backward movement as required by some cards **shall** be modelled as direct movement. |

## Client Queries

The functional requirements in this section correspond to clarifications/changes/additions of/to the requirements in the User Requirements document, requested and received by our team via email.

Note that only relevant queries that have an impact on the functional requirements have been included.

Many of these queries have no direct functional requirements associated with them, however requirements from other parts of this document may reference them for clarity.

| **ur-cq-1** | **User Requirement Client Query 1** |
|---|---|
| Question | Amount of players. The game was limited to 6 players as the bank had a finite amount of money, are we following the 6 player maximum rule? If so, would expansion of player quantity limit be something you may look at in the future? |
| Answer | We did some thinking on this. The current game has a limit of 6 players imposed by the player tokens. There is no particular reason why it can't be more than 6, but in practice we found that with more than 6, the chances of any player actually winning was too low and made the gameplay boring. So 6 will remain the maximum number of players. |

| **ur-cq-2** | **User Requirement Client Query 2** |
|---|---|
| Question | If there is the ability to add more players, could you suggest more player piece objects; further to the hatstand, spoon, etc. Alternatively, we can incorporate a way of adding pieces at any time. |
| Answer | Whilst there are no more players, it might be nice if the pieces could be customizable. |

| **ur-cq-3** | **User Requirement Client Query 3** |
|---|---|
| Question | Can players trade with each other? Having played the game, we found that without players being able to trade properties for other property of money, the game does not conclude, as it is rare for players to get a whole street. |
| Answer | We are thinking about this one, and will send out further guidance at a later date. |

| **ur-cq-4** | **User Requirement Client Query 4** |
|---|---|
| Question | Is there a starting price for an auction, or does the price begin at 0? |
| Answer | The starting price should be £1. |

| **ur-cq-5** | **User Requirement Client Query 5** |
|---|---|
| Question | Is a property that goes back to the bank auctioned or goes back to being for sale when someone lands on the property? |
| Answer | The property becomes available for sale when someone next lands on that property. |

| **ur-cq-6** | **User Requirement Client Query 6** |
|---|---|
| Question | Can the user customise the amount of tiles, i.e removing tiles wholesale from the spreadsheet or adding their own tiles? |
| Answer | The amount of tiles should remain fixed as in the original game |

| **ur-cq-7** | **User Requirement Client Query 7** |
|---|---|
| Question | Are tokens allocated to players randomly or can players choose? |
| Answer | It would be nice to be able to choose (you can in real life). But equally a random option is useful when players don’t care. |

| **ur-cq-8** | **User Requirement Client Query 8** |
|---|---|
| Question | Should we implement a timer/turn limit for the players on the abridged version? - A timer specifically so people cannot get into the lead then waste time |
| Answer | A timer would be in the spirit of the original game. |

| | |
|---|---|
| **ur-cq-9** | **User Requirement Client Query 9** |
| Question | When a property goes to auction, the rules say that "Each player makes a bid to the bank." Does this mean that each player may only make a single bid at any one auction, or should the players be able to make multiple bids, much like how an auction would work in real life? |
| Answer | It is intended to work as a sealed bid system. Each player makes a single bid to the bank. A player may may the bid public or may choose to keep it private between them and the bank. Whether a bid is public or not is entirely down to the player and does not influence the bank's decision. The bank sells to the highest bidder |
| **fr-cq-9-1** | **Functional Requirement Client Query 9.1** |
| Type | Mandatory |
| Description | Eligible players **shall** take turns to place a single bid on the property, in the same order they take regular turns, starting from the player who landed on the property. |
| **fr-cq-9-2** | **Functional Requirement Client Query 9.2** |
| Type | Mandatory |
| Description | On their turn, each player **shall** choose to place a bid of no less than £1 or to exit the auction. Bids can be placed privately, where only the player placing the bid and the bank know the value of the bid, or publicly where all players know the value of the bid. |
| **fr-cq-9-3** | **Functional Requirement Client Query 9.3** |
| Type | Mandatory |
| Description | The property in question **shall** be transferred to the player who placed the highest bid (either public or private) in exchange for the value of their bid. |

| | |
|---|---|
| **ur-cq-10** | **User Requirement Client Query 10** |
| Question | If a player decides not to buy a property, are they still allowed to be part of the auction for that property? |
| Answer | No, the player has already decided that they did not wish to purchase the property, so that would not be in the spirit of the original game. |
| **fr-cq-10-1** | **Functional Requirement Client Query 10.1** |
| Type | Mandatory |
| Description | All eligible players, except the player who landed on the property, **shall** be able to bid in an auction. |