# Design:

Main frame layout:

A close up of a piece of paper

Description generated with high confidence

UML:

|  |
| --- |
| Card //abstract |
| name: String  isTrump: boolean |
| getName(): String  getFileName(): String //abstract  getDescription(): String //abstract  getHardness(): String //abstract  getGravity(): String //abstract  getCleavage(): String //abstract  getCrustalAbundance(): String //abstract  getEcoValue(): String //abstract  getCategoryInPlay(categorySelectionNum: int): String //abstract  getCategoryValueInPlay(categorySelectionNum: int): String //abstract |

|  |  |
| --- | --- |
| MineralCard | SupertrumpCard |
| filename: String  hardness: String  gravity: String  cleavage: String  crustalAbundance: String  ecoValue: String | filename: String  description: String |
| MineralCard(name: String, filename: String, hardness: String, gravity: String, cleavage: String, crustalAbundance: String, ecoValue: String)  getFilename(): String  getHardness(): String  getGravity(): String  getCleavage(): String  getCrustalAbundance(): String  getEcoValue(): String  getCategoryInPlay(categorySelectionNum: int): String  getCategoryValueInPlay(categorySelectionNum: int): String  getDescription(): String | SupertrumpCard(name: String, filename: String, description: String)  getFilename: String  getDescription(): String  getHardness(): String  getGravity(): String  getCleavage(): String  getCrustalAbundance(): String  getEcoValue(): String  getCategoryInPlay(categorySelectionNum: int): String  getCategoryValueInPlay(categorySelectionNum: int): String |

|  |
| --- |
| Game |
| playerChoiceCategory: String  playerChoiceValue: double  winners: ArrayList<Player>  countWinner: int  winnerIndex: int  playingValue: double  playingValueAsString: String  playingCategory: String  startRound: boolean  newRound: boolean  hasWinner: boolean  countOutPlayer: int  playerAmount: int  players: ArrayList<Player> |
| Game()  setDealer(): void  getDealerIndex(): int  setWinnerIndex(winnerIndex: int): void  getPlayerChoiceCategory(): String  getPlayerChoiceValue(): double  setPlayerChoiceValue(c: ArrayList<Card>, cardIndex: int, categorySelectionNum: int): void  getPlayingValue(): double  setPlayingValue(value: double): void  getPlayingValueAsString(): String  setPlayingValueAsString(c: ArrayList<Card>, cardIndex: int, categorySelectionNum: int): void  getPlayingCategory(): String  newRoundChecking(): void  trumpReset(): void  getLeadPlayerIndex(): int  specialWinningRound(playerIndex: int): void  getCleavageIntVal(cleavage: String): int  getCrustalAbundanceIntVal(crustalAbundance: String): int  getEconomicValueIntVal(ecoValue: String): int |

|  |
| --- |
| Hand |
| cards: ArrayList<Card> |
| Hand()  add(c: Card): void  getMagnetite(): Card  getTheGeophysicist(): Card  getCards(): ArrayList<Card> |

|  |
| --- |
| Table |
| allCards: ArrayList<Card> |
| Table()  collectCards(fileName: String, cards: ArrayList<String[]>): void  draw(): Card  isEmpty(): boolean  getAllCards(): ArrayList<Card> |

|  |
| --- |
| Player |
| name: String  table: Table  hand: Hand  inRound: boolean  isWinner: boolean  isDealer: boolean |
| Player(t: Table, name: String)  drawCards(n: int): void  hasMagnetite(): boolean  hasTheGeophysicist(): boolean  getHand(): Hand  getName(): String |

|  |
| --- |
| LaunchControl |
| game: Game  table: Table  jmst: JMineralSuperTrumps |
| launchPlayerSetupFrame(countPlayer: int): void  launchMineralSuperTrumps(): void  launchChooseCategoryFrame(playerIndex: int, cardIndex: int): void  launchSpecialWinningFrame(playerIndex: int, cardIndex: int): void  launchWinnerPromptFrame(): void  setGame(game: Game): void  setTable(table: Table): void |

|  |
| --- |
| JPlayerAmountFrame |
| launchControl: LaunchControl  game: Game  table: Table  mp, buttonPanel, imagePanel, userPanel: JPanel  message, greeting, image: JLabel  button3, button4, button5: JButton  img: Icon  bigFont: Font |
| JPlayerAmountFrame()  main(args: String[]): void  actionPerformed(e: ActionEvent): void |

|  |
| --- |
| JPlayerSetupFrame |
| launchControl: LaunchControl  countPlayer: int  mp, buttonPanel: JPanel  message: JLabel  nameTextField: JTextField  but: JButton  bigFont: Font  game: Game  table: Table |
| JPlayerSetupFrame(game: Game, table: Table, countPlayer: int)  setLaunchControl(lc: LaunchControl): void  actionPerformed(e: ActionEvent): void |

|  |
| --- |
| JMineralSuperTrumps |
| launchControl: LaunchControl  mp, cardsPanel, infoPanel, playerPanel, statusPanel, buttonPanel: JPanel  playerNameLb, cardAmountLb, cardPileLb, prompt, playingCategoryLb, playingValueLb, remainingCardsLb: JLabel  passButton: JButton  cardPileDefault, cardImg: Icon  scroll: JScrollPane  bigFont: Font  game: Game  table: Table  cardButtons: ArrayList<JButton>  playerIndex: int |
| JMineralSuperTrumps(game: Game, table: Table)  setLaunchControl(lc: LaunchControl): void  play(): void  gameFlow(): void  clearCardsPanel(): void  changeCategoryByTrump(cardName: String): void  determineWinner(): void  actionPerformed(e: ActionEvent): void |

|  |
| --- |
| JChooseCategoryFrame |
| launchControl: LaunchControl  placeHolder, message: JLabel  hardnessButton, specificGravityButton, cleavageButton, crustalAbundanceButton, ecoValueButton: JButton  bigFont: Font  game: Game  jmst: JMineralSuperTrumps  playerIndex: int  cardIndex: int |
| JChooseCategoryFrame(game: Game, jmst: JMineralSuperTrumps, playerIndex: int, cardIndex: int)  setLaunchControl(lc: LaunchControl): void  assignCategory(category: String, categorySelectionNum: int)  actionPerformed(e: ActionEvent): void |

|  |
| --- |
| JSpecialWinningFrame |
| launchControl: LaunchControl  message: JLabel  yesButton, noButton: JButton  bigFont: Font  game: Game  jmst: JMineralSuperTrumps  playerIndex: int  cardIndex: int |
| JSpecialWinningFrame(game: Game, jmst: JMineralSuperTrumps, playerIndex: int, cardIndex: int)  setLaunchControl(lc: LaunchControl): void  actionPerformed(e: ActionEvent): void |

|  |
| --- |
| JWinnerPromptFrame |
| launchControl: LaunchControl  message, winnerPlace: JLabel  button: JButton  bigFont: Font |
| JWinnerPromptFrame(game: Game)  setLaunchControl(lc: LaunchControl): void  actionPerformed(e: ActionEvent): void |

# Testing:

A screenshot of a social media post

Description generated with very high confidence

The game starts with the JPlayerAmountFrame that shows the window contains the icon of the game, the greeting and three buttons which are represented for the number of players will play in the game. Three options for player amount include “3 players”, “4 players” and “5 players”. The player just needs to click on one of three buttons to determine their number of players then the game will move on to the next stage.

A screenshot of a cell phone

Description generated with very high confidence

A screenshot of a cell phone

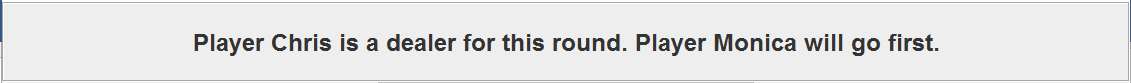
Description generated with very high confidence

After choosing the number of players will play in the game, the game asks for the name of each player by starting the JPlayerSetupFrame. The game asks for the name of the first player and then after the player clicks on the “OK” button, another JPlayerSetupFrame will appear for the second player to type their name. The iteration will keep going on until the last player finishes typing their name and click on the “OK” button.

A screenshot of a cell phone

Description generated with very high confidence

After the player’s name setting up process, the JMineralSuperTrumps, which is a main frame of the game, will appear and the players will start playing.



This label shows the main status of the game such as who is the dealer, who is the player will go first, which card, category and value are just played by the player, prompting for the new round, prompting for the error card (the card’s value less than the current playing value), and prompting for the winner.

A screenshot of a cell phone

Description generated with very high confidence

In this portion, the top label shows the current turn belongs to which player. The bottom label shows the number of cards are owned by the player in their turn. The button in the center allows player to pass their turn if they click on it.

A screenshot of a cell phone

Description generated with very high confidence

In this portion, the “Playing category” and “Playing value” labels show the current category and value are played by the player. As the game is just started, those labels don’t show anything but they will change throughout the game. The bottom label shows the number of cards remaining on the table.



This label shows the card pile. If the game is just started or the game turns into a new round, this label will show this default card image. When the player plays a card successfully, this default card image will be replaced by the image of that card.

A screenshot of a cell phone

Description generated with very high confidence

In the last portion, there is a category of card image buttons. Each of them represents for each card the current player has. The player can click on one of those card image buttons to play the card. And there is the scrollbar that can be used by player to see the whole category of card image buttons.

A screenshot of a cell phone

Description generated with very high confidence

As the game is just started, when the first player clicks on one of the card image buttons, the game will ask for the category that player wants to play by starting the JChooseCategoryFrame. The player can click on the button that has their preferred category to start the round. The player can also change another card for choosing category by clicking on “X” in the JChooseCategoryFrame to close the choosing category window and then choose another card image button.

A screenshot of a cell phone

Description generated with very high confidence

After the first player played their card, the top label prompts the card name, chosen category and value of that card played by that player. And then the next player will take turn. The player turn label shows the name of the next player. The card buttons panel in the bottom will change to the next player’s card image buttons. The “Playing category” and “Playing value” labels show the category and the value of the card the previous player played.

A screenshot of a cell phone

Description generated with very high confidence

A screenshot of a cell phone

Description generated with very high confidence

As the playing category has been chosen by the first player, the other players don’t need to choose the playing category anymore until the new round begins. From now, when the player clicks on their preferred card image button, if the card is valid, that card will be automatically played, if not, the top label will prompt a message that remind the player to play the valid card (bigger value in the playing category than the previous card).

A screenshot of a cell phone

Description generated with very high confidence

When the player clicks on the pass button, the top label will prompt that player has passed their turn and the next player will take the turn to continue the game. The player, who clicks on the pass button, cannot play again until next round or the trump card is played by other players. They also add one card to their cards.

A screenshot of a cell phone

Description generated with very high confidence

A screenshot of a cell phone

Description generated with very high confidence

When the player plays the trump card, the game prompts that the current playing category has been changed to the category of the trump card by that player in the top label and the playing value label will be reset. After that, the player, who just played the trump card, will keep continuing by clicking on whatever card image button to resume play. The player, who has passed their turn, is now able to play again.

A screenshot of a cell phone

Description generated with very high confidence

A screenshot of a cell phone

Description generated with very high confidence

For the “The Geologist” trump card, when they player clicks on that card, the game starts the JChooseCategoryFrame for the player to choose their category that they want to change. After clicking on their preferred category, the “The Geologist” trump card will be played.

A screenshot of a cell phone

Description generated with very high confidence

When all but one player has passed, the top label prompts “The current round has ended. Start new round!” and the playing category and value labels are reset. The last player in the previous round will lead this round and decide the playing category for this round.

A screenshot of a social media post

Description generated with very high confidence

A screenshot of a cell phone

Description generated with very high confidence

If the player holds both “The Geophysicist” and “Magnetite” cards, when that player clicks on the “The Geophysicist” card image button, the game will start the JSpecialWinningFrame and ask whether that player wants to place both cards to win the round or not. If player clicks on “No” button, the “The Geophysicist” card will be played as the normal trump card. If player clicks on “Yes” button, the round will end, the player will remove both cards and that player will become the lead player for the next round.

A screenshot of a cell phone

Description generated with very high confidence

The game will keep playing to determine a winner. If the player is the first player to lose all their card, then that player will be the winner and the game will congratulate that player and prompt “You are the winner!” in the top label. And then the next player will continue the round to play with the value of the last card of the winner.

A screenshot of a cell phone

Description generated with very high confidence

If the player loses all their card after the winner, the game will only prompt “Congratulations! Your hand is empty!” in the top label.

A screenshot of a cell phone

Description generated with very high confidence

When all but one player has lost their cards, the game shuts off the main frame and starts the JWinnerPromptFrame that shows the winner places from highest to lowest and then when the player clicks on the “OK” button or “X”, the game will exit.