Yebelo Spring Developer Assignment

Topic

Make an API based Monopoly game.

Assumptions / Requirements

1. The game should feature 10 places.
2. The game should feature 1 place called “start”
3. Your program need not account for multiple parallel games being played.
4. Your program only supports 2 player monopoly.

Data

Place	**Buy Price**	**Rent**
Old Kent Road	$60	$30
Whitechapel Road	$60	$30
King's Cross station	$200	$100
The Angel Islington	$100	$50
Euston Road	$100	$50
Pentonville Road	$120	$60
Pall Mall	$140	$70
Whitehall	$140	$70
Northumberland Avenue	$160	$80
Marylebone station	$200	$100

Features to Implement

1. Ability for a host (also a player) to create a new game, therefore discarding the old one.
2. Ability for a host (also a player) to play the game with another player.
3. Ability for a player to have their cash system, and start off with $1000.
4. Ability for a player to roll 2 dies and be informed of the place landed.
5. Ability to auto purchase a place when landed and informed.
6. Ability to auto pay to the person whose place was landed on and informed.

7. Ability to gain +$200 when the “start” is crossed.
8. Ability to declare who the winner of a game is based on bankruptcy or the player with the highest cash before turn 50.

Example System

Person A : http://localhost:8080/create-game/ >> Game Created Successfully
Person A : http://localhost:8080/roll-die/p1 >> Die rolled 11 and landed on Place ABC, Unclaimed place and hence bought for $200. Remaining balance is $800.
Person B : http://localhost:8080/roll-die/p2 >> Die rolled 4 and landed on Place DEF, Unclaimed place and hence bought for $150. Remaining balance is $850.
...
Person A : http://localhost:8080/roll-die/p1 >> Die rolled 1 and landed on Place DEF, paid rent $100. Remaining balance is $700.
Person B : http://localhost:8080/roll-die/p2 >> Die rolled 4 and landed on Place UES, Unclaimed place and hence bought for $150. Remaining balance is $700. Also Crossed “Start” gaining +200. Remaining Balance $900.
…
Person A : http://localhost:8080/roll-die/p1 >> Die rolled 1 and landed on Place DEF, paid rent $100. Remaining balance is $-100. Game Over, You lose!

Non-Functional Features to Incorporate

1. Database data persistence so that even if the spring application is restarted the game can be played uninterrupted.
2. Design Patterns for a highly maintainable and readable code base.
3. Documentation for a highly maintainable and readable code base.
4. Unit Testing (Does not have to be end-to-end tested, basic testing suffices) for a robust system.

Git Usage

Project submission shall be done via pushing your codebase to Github.

1. Commit often with descriptive messages (we may look into your commit history).
2. Bonus: use a branch per feature simulating how you would work as a team

Mandatory files of your github repo’s root folder

1. Minimal usage guide and features implemented, as guide.txt file.
2. (Design/Architectural) (Patterns/decisions) used, as decisions.txt file.
3. Picture of Database Design as database_schema.png. We recommend you design on draw.io before starting your project.
4. (Bonus) System Design Drawing as system_design.png.

Extensibility?

Any additional features implemented, should be clearly stated in guide.txt and will be added as bonus points for your project.

Checklist

- Diagrammed the database schema.
- (Bonus) Diagrammed the system design.
- Completed a minimal working model with documentation of the required features.
- Improvise working model to incorporate appropriate design/architectural patterns.
- Perform unit testing.
- Create and Push guide.txt
- Create and Push decisions.txt
- Upload and Push database_schema.png
- (Bonus) Upload and Push system_design.png
- Work on adding more features if time permits.