Bilkent University

Department of Computer Engineering

CS319 Term Project

Monopoly Bilkent Edition

Group 1B

Iteration 2 Final Report

Project Group Members:

Melike Fatma Aydoğan - 21704043

Ahmet Cemal Alıcıoğlu - 21801700

Zübeyir Bodur - 21702382

Beril Canbulan - 21602648

Mehmet Çalışkan - 21802356

Instructor: Eray Tüzün

Teaching Assistants: Barış Ardıç, Elgun Jabrayilzade, Emre Sülün

Table of Contents

Introduction **3**

Lessons Learnt **4**

Build Instructions **5**

Create a new project with the source code 5

Connect a LogMeIn Hamachi server 15

Compile & Run the Project 18

User's Guide **19**

Work Allocation **25**

Melike Fatma Aydoğan 25

Ahmet Cemal Alıcıoğlu 25

Zübeyir Bodur 25

Beril Canbulan 26

Mehmet Çalışkan 26

1. Introduction

So far, we have successfully implemented the main logic of the system. We were also able to successfully implement some of the features that we have promised in the analysis. These are:

- Multiplayer
- Trading
- Alliance (Teams)
- Speed Die

However, there are also features that weren't implemented. These are:

- Saving the instance of a multiplayer game
- Auctions
- Letting the player move freely when they roll a triple (1-1-1, 2-2-2, 3-3-3)
- Giving player the opportunity to trade or mortgage before bankrupting

Moreover, there are also features that were discarded during the design or implementation stages of the development:

- Different maps: We decided that the game should only include the Bilkent Map and also changed the name of the application to "Monopoly Bilkent Edition"

2. Lessons Learnt

- Dealing with change requires effective communication in a team. If the communication is weak, there might be failure(s) to deal with change, such as redesigning interfaces, merge conflicts in version control etc.

- It is always a challange to understand another developer's code no matter how well commented and readable it is. Of course, it is even harder if it is not so.

- Most additional features require more effort than building the core logic of the game:

    - If the developers want their game to have a multiplayer game mode, they may need to change the whole decomposition of the system.

    - Easy looking feature, the speed die, turned out to be a lot harder than it looked. However, diagrams produced during the analysis stage were helpful, e.g. Sequence Diagram - Player's Turn in the 1st Iteration, Activity Diagrams in the 2nd Iteration etc.

        - Implementing Mr. Monopoly face was the hardest face of the speed die to implement. It required not-so-simple problem solving so that the system works just like in the analysis diagrams.

        - . It required a problem solving for computing the next unowned property/ next property that the player would pay rent.

        - However, a simple looking feature, Bus face and triple rolling, would require more than popping a panel since the system is a multiplayer game.

        - We then realized that Mr. Monopoly face is the one that makes the game faster, which was the whole reason to add a feature called speed die.

3. Build Instructions

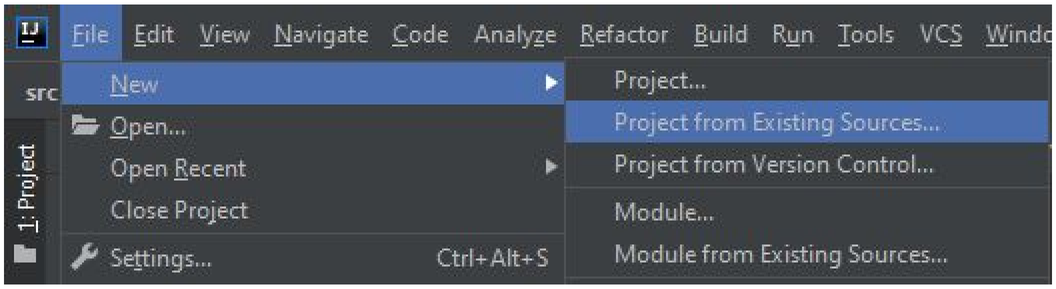
Building the system from the source code consists of three main steps: creating an IntelliJ IDEA project, connecting to LogMeIn Hamachi and compiling the project. After doing all these steps, you should be able to play the game with the people who logged in to the same LogMeIn Hamachi server as you. JDK 1.8 is also required for build.

3.1. Create a new project with the source code

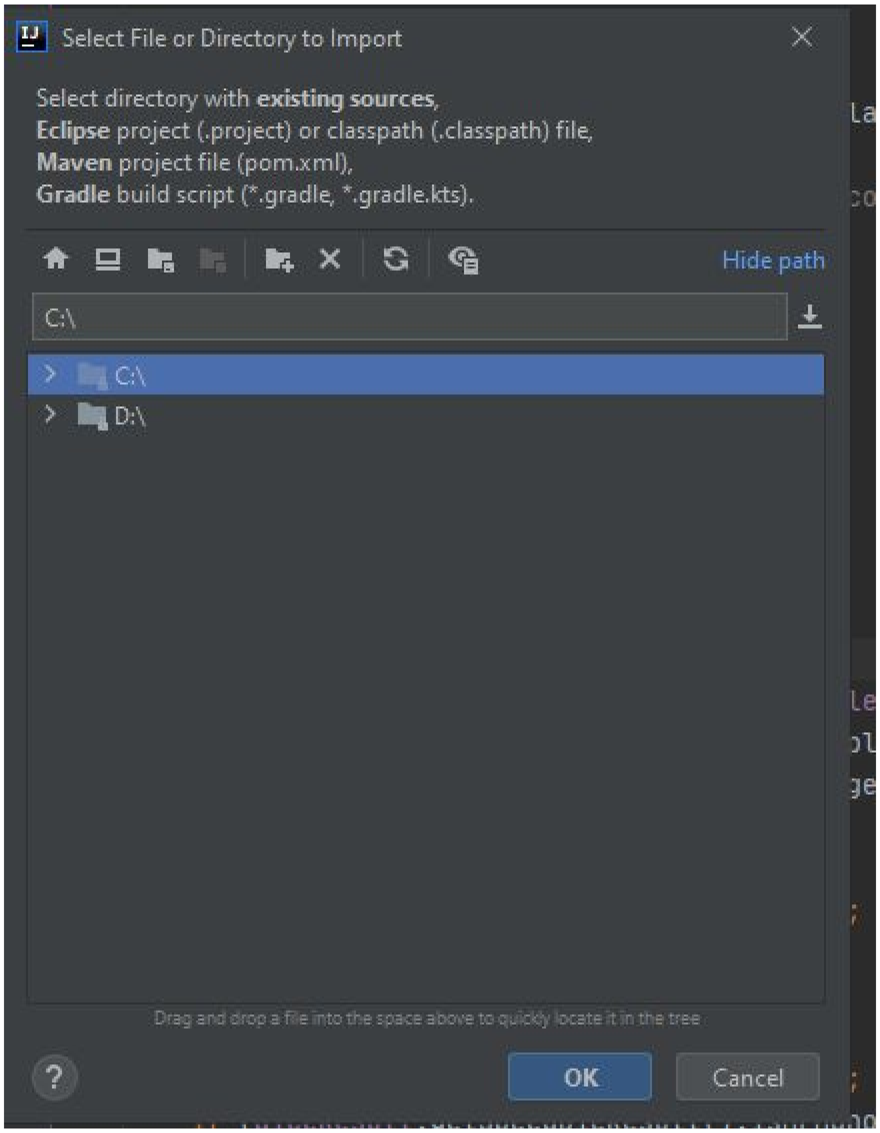
If you don't have the source code, you can download it from https://github.com/melikeaydogan/CS-319-Monopoly-Team-1B. Download the zip file and extract it inside a folder.

Then, you need to install two programs if you don't have: IntelliJ IDEA (2020.2.4 (Ultimate Edition)). If you are not connected to the same network with other players, you need to install a VPN application like LogMeIn Hamachi (2.2.0.633). Small version differences shouldn't prevent you from successfully building thou.

Then, open IntelliJ IDEA. From there, open a new project using the toolbar: File > New > Project from Existing Sources.



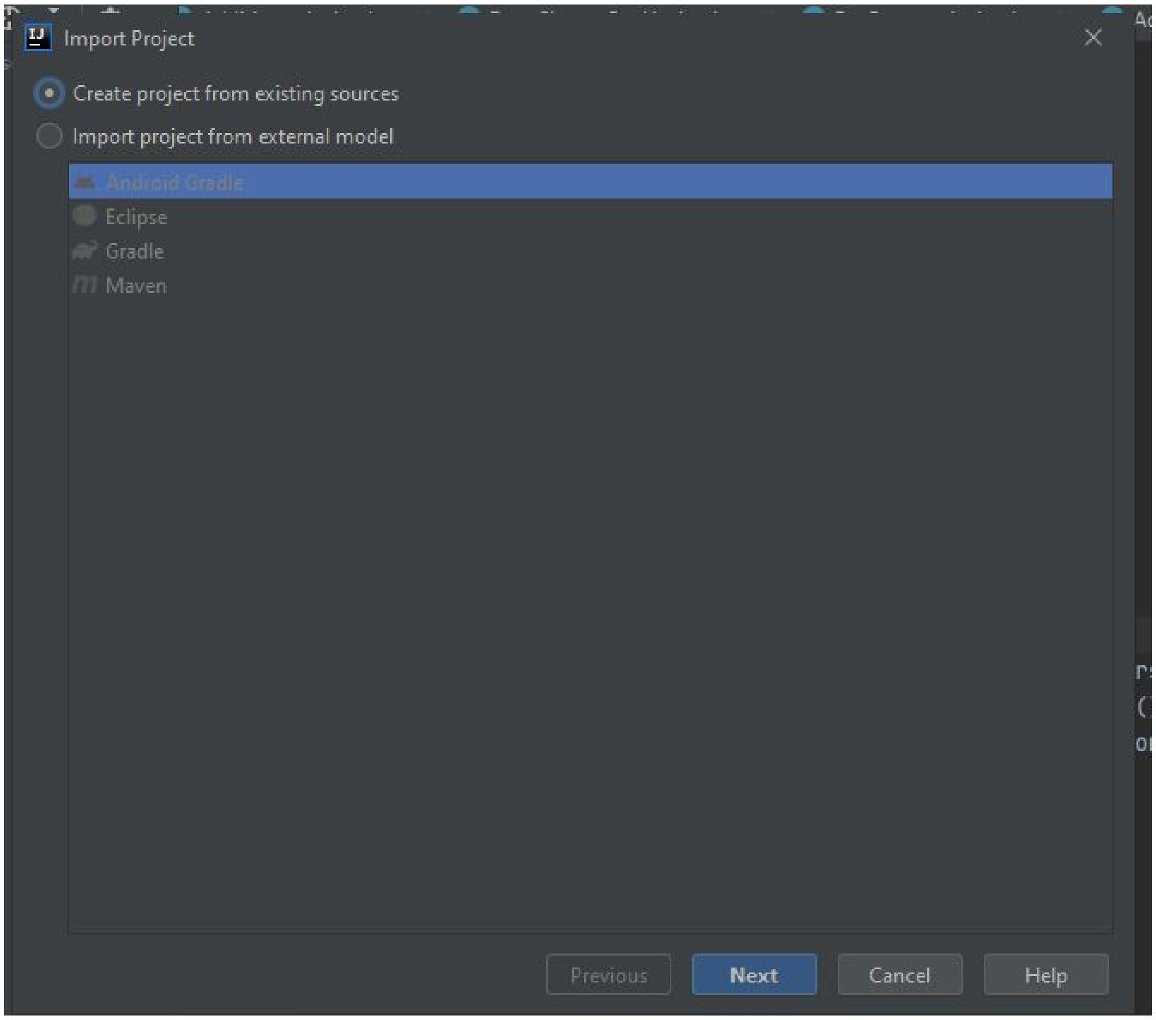
There, choose the file that includes the source code. If you downloaded the source code from the GitHub, this should be the file you extracted the .zip into.


Select File or Directory to Import

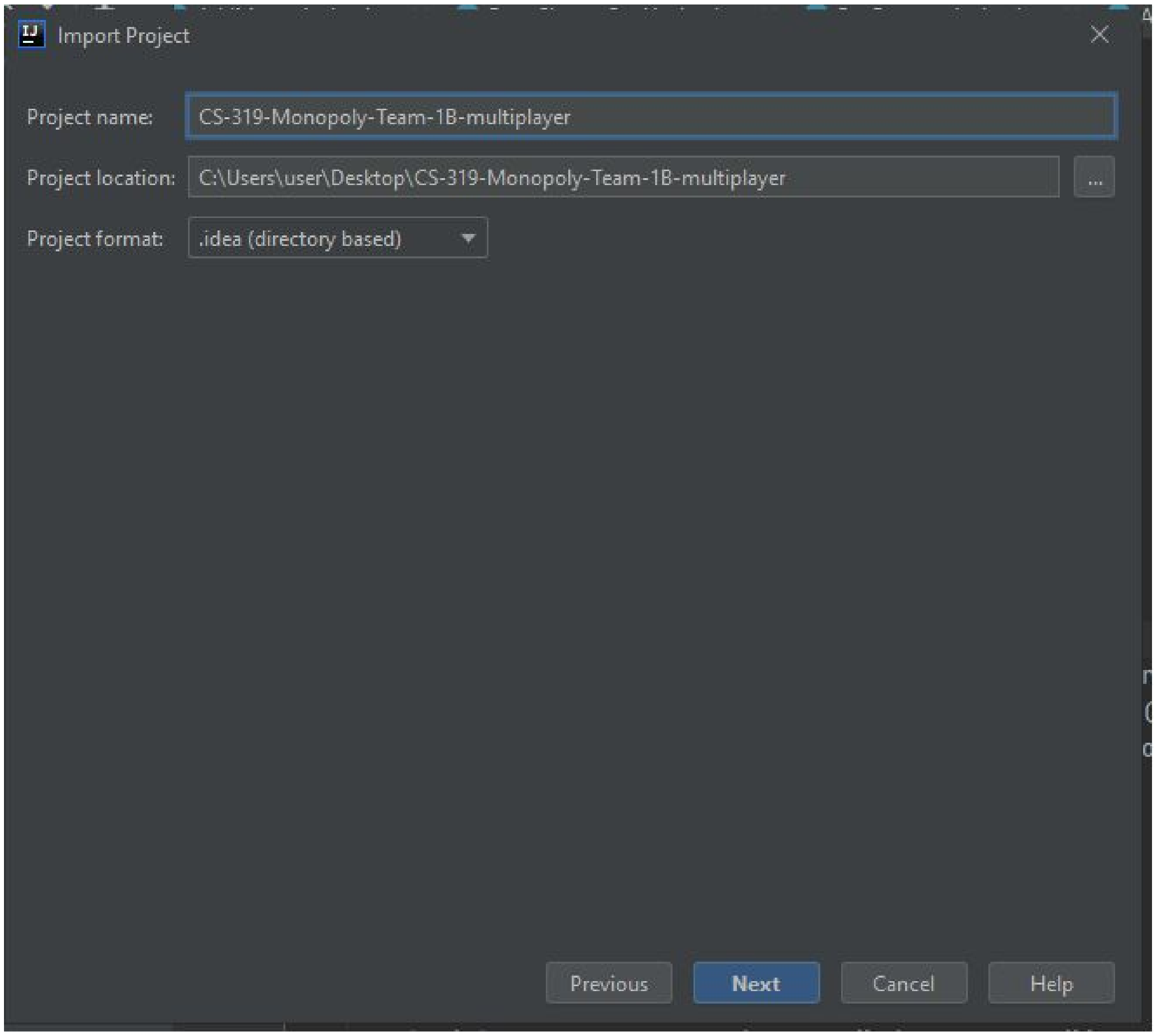
Select directory with **existing sources**,
Eclipse project (.project) or classpath (.classpath) file,
Maven project file (pom.xml),
Gradle build script (*.gradle, *.gradle.kts).

Hide path
C:\
C:\
D:\
Drag and drop a file into the space above to quickly locate it in the tree
OK
Cancel

Then, in the next screen, you will be asked to create the project or import from other sources. Select “Create project from existing sources” and click “Next”.


Import Project
Create project from existing sources
Import project from external model
Android Gradle
Eclipse
Gradle
Maven
Previous
Next
Cancel
Help

In the next screen, you will be asked to enter a project name and location. They should be already there, so click “Next”.

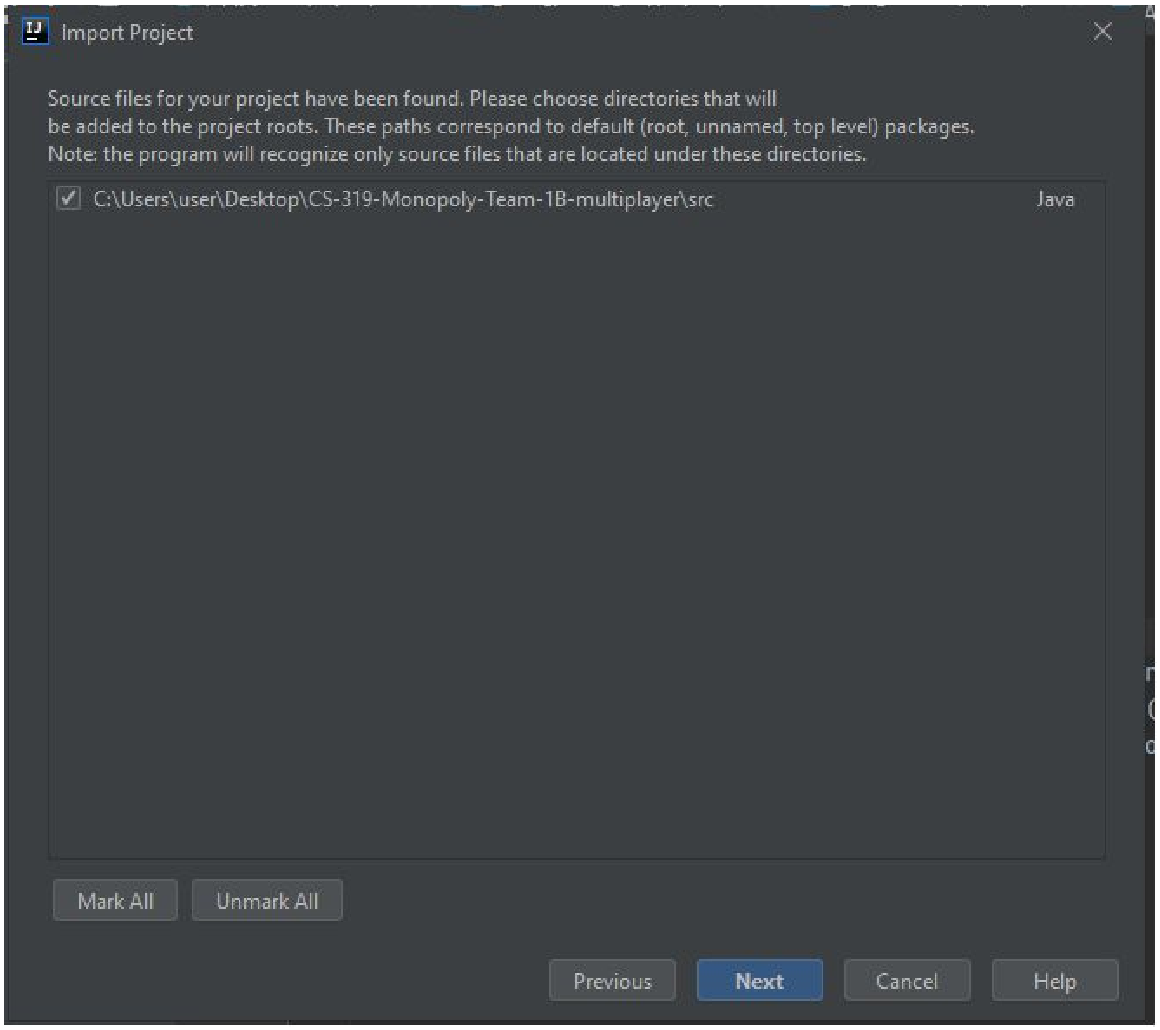

Import Project

Project name: CS-319-Monopoly-Team-1B-multiplayer

Project location: C:\Users\user\Desktop\CS-319-Monopoly-Team-1B-multiplayer

Project format: .idea (directory based)

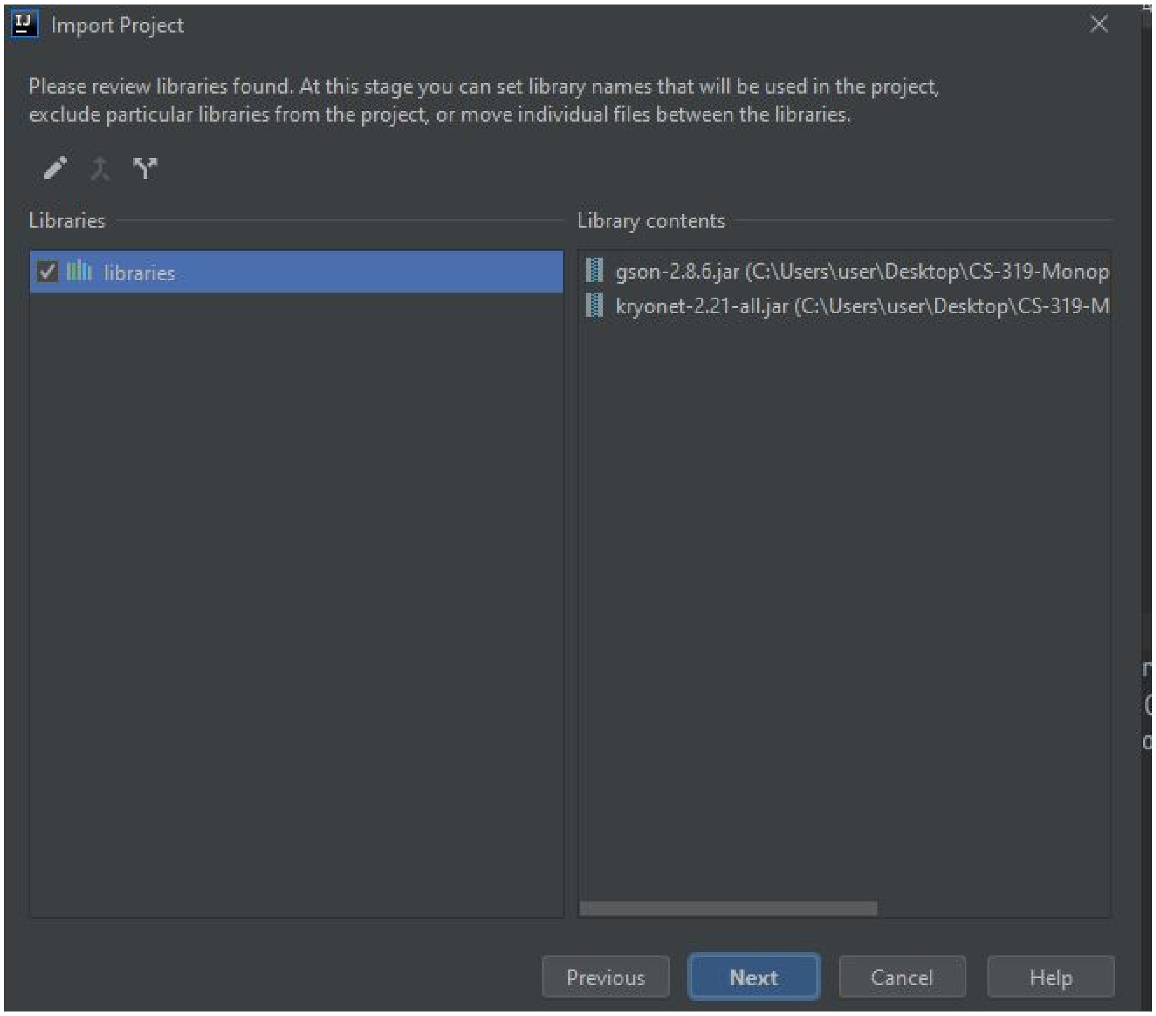
In the next screen, you will be asked to select a src location. In our GitHub, it was already called “src” so it should also be automatically assigned. Click “Next”.


Import Project

Source files for your project have been found. Please choose directories that will be added to the project roots. These paths correspond to default (root, unnamed, top level) packages. Note: the program will recognize only source files that are located under these directories.

✓	C:\Users\user\Desktop\CS-319-Monopoly-Team-1B-multiplayer\src	Java

In the next screen, you should be seeing one library called *libraries*, with library contents *gson-2.8.6.jar* and *kryonet-2.21-all.jar*. *libraries* should be automatically marked. Click “Next”.


Import Project

Please review libraries found. At this stage you can set library names that will be used in the project, exclude particular libraries from the project, or move individual files between the libraries.

Libraries

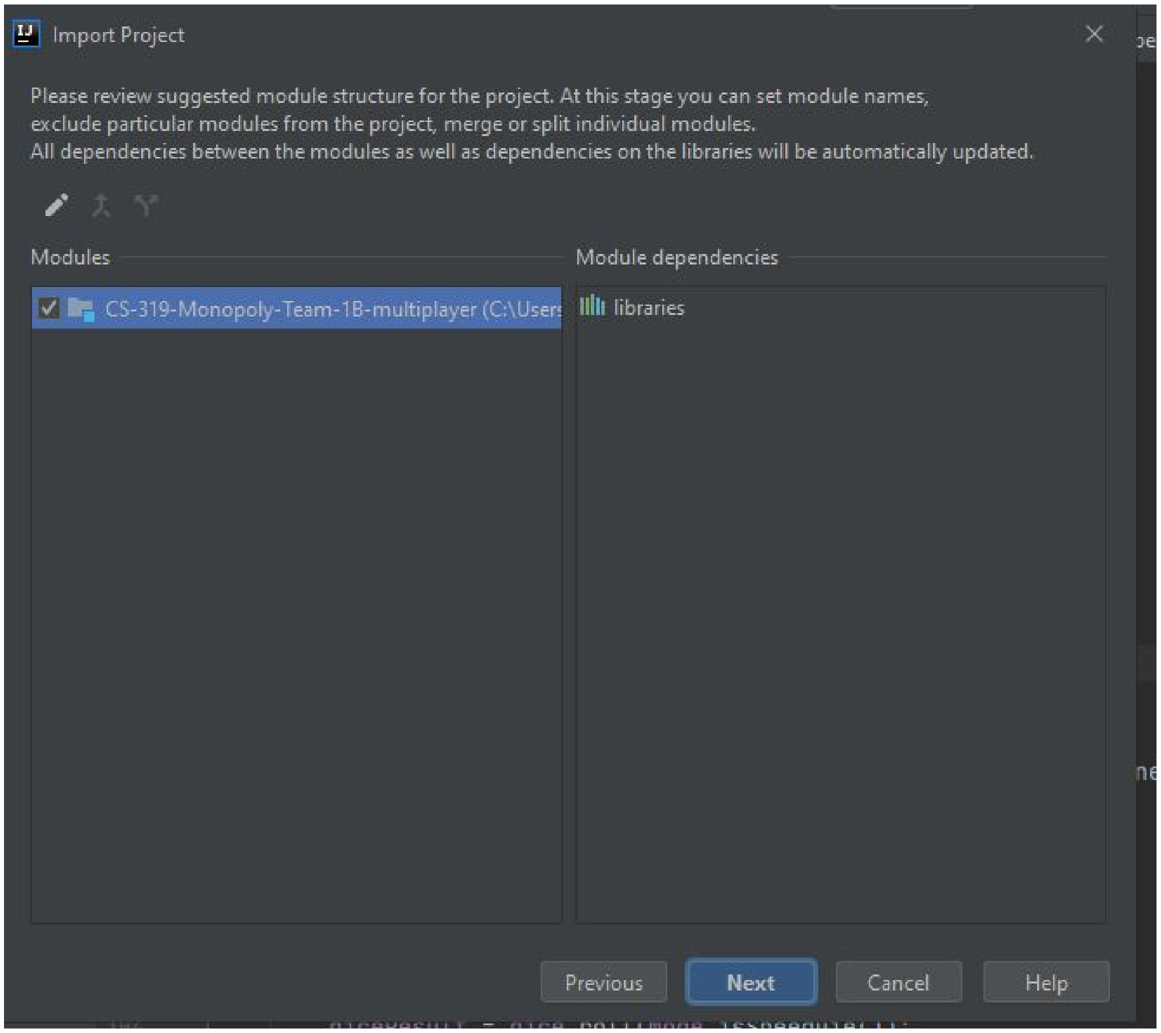
☑ libraries

Library contents

gson-2.8.6.jar (C:\Users\user\Desktop\CS-319-Monop

kryonet-2.21-all.jar (C:\Users\user\Desktop\CS-319-M

In the next screen, you should be seeing 1 module structure, named the same as the project name, with module dependencies *libraries*. Click “Next”.

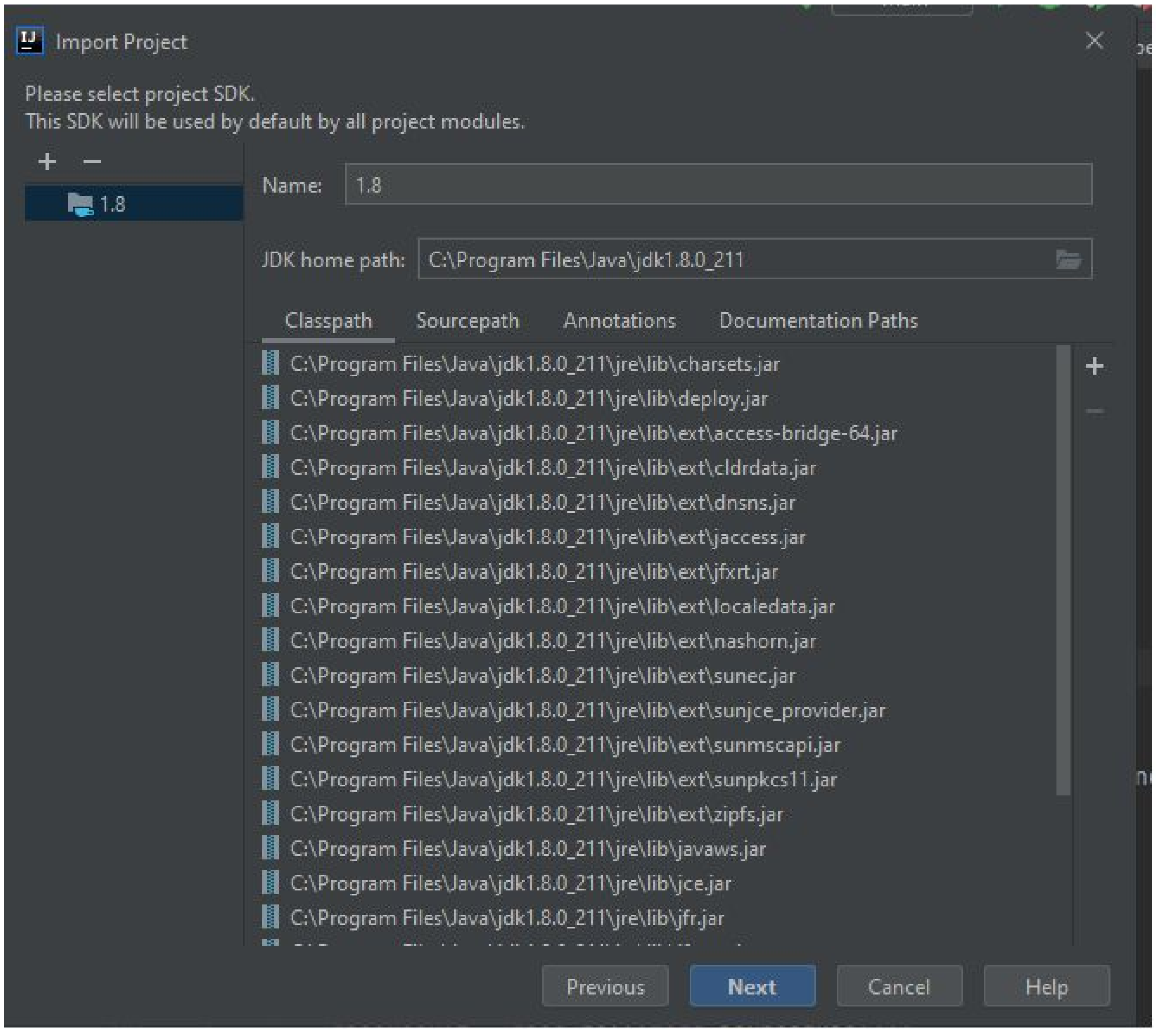

Import Project

Please review suggested module structure for the project. At this stage you can set module names, exclude particular modules from the project, merge or split individual modules.
All dependencies between the modules as well as dependencies on the libraries will be automatically updated.

Modules: CS-319-Monopoly-Team-1B-multiplayer (C:\Users

Module dependencies: libraries

In the next screen, you will be asked to select SDK. If JDK 1.8 is already installed in your computer, it will also be automatically selected. So, click “Next”. Then, IntelliJ IDEA will send you back to the *libraries* screen. Click “Next”. It will also ask you to check your module dependencies again, so Click “Next” again.

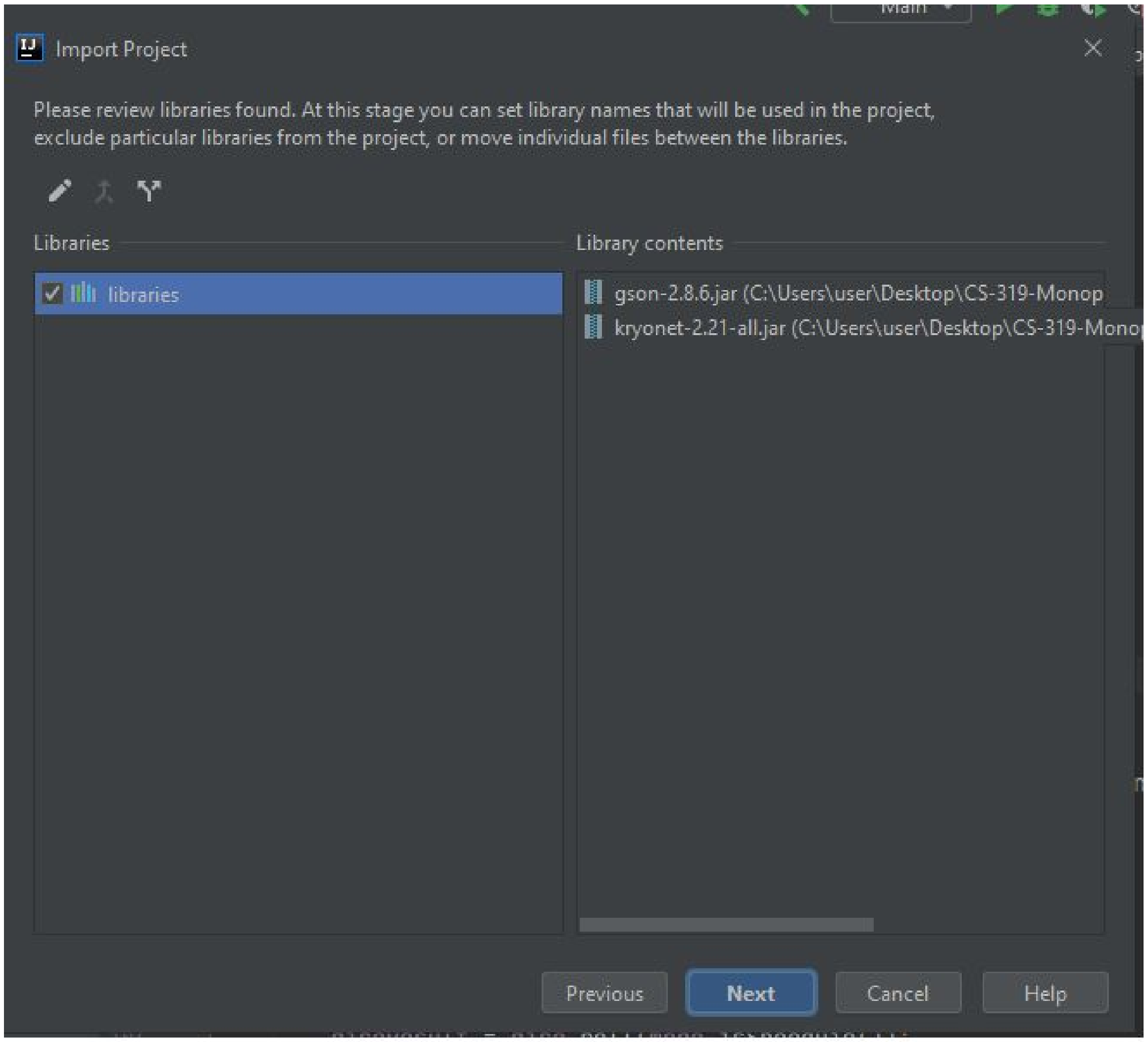


Please select project SDK.
This SDK will be used by default by all project modules.

Name: 1.8

JDK home path: C:\Program Files\Java\jdk1.8.0_211

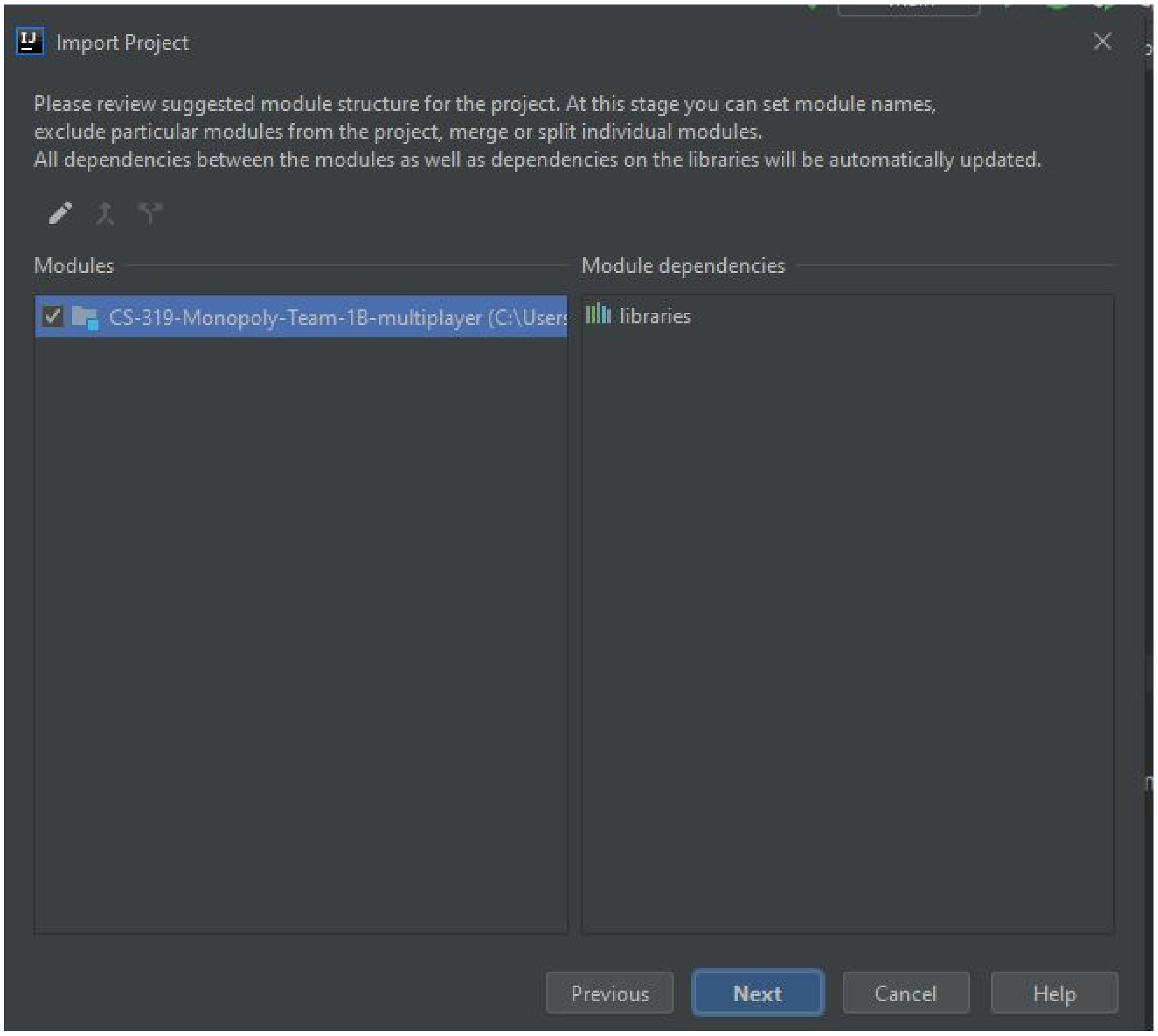
```
C:\Program Files\Java\jdk1.8.0_211\jre\lib\charsets.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\deploy.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\access-bridge-64.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\cldrdata.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\dnsns.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\jaccess.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\jfxrt.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\localedata.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\nashorn.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\sunec.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\sunjce_provider.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\sunmscapi.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\sunpkcs11.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\ext\zipfs.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\javaws.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\jce.jar
C:\Program Files\Java\jdk1.8.0_211\jre\lib\jfr.jar
```



Import Project

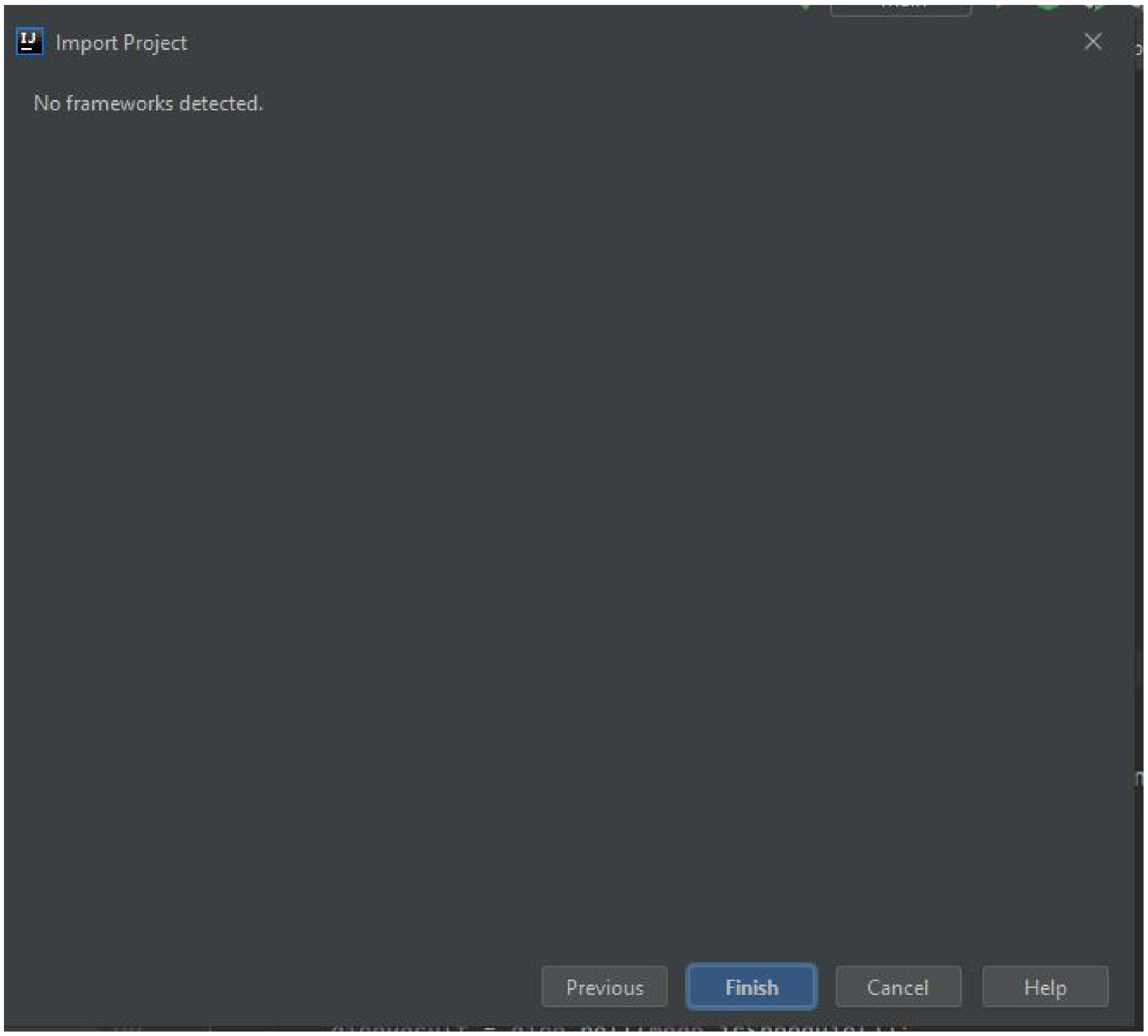
Please review libraries found. At this stage you can set library names that will be used in the project, exclude particular libraries from the project, or move individual files between the libraries.

gson-2.8.6.jar (C:\Users\user\Desktop\CS-319-Monop
kryonet-2.21-all.jar (C:\Users\user\Desktop\CS-319-Mono


Import Project

Please review suggested module structure for the project. At this stage you can set module names, exclude particular modules from the project, merge or split individual modules.
All dependencies between the modules as well as dependencies on the libraries will be automatically updated.

Finally, it should tell you that no frameworks detected and by clicking “Finish”, your IntelliJ IDEA project should be opening.

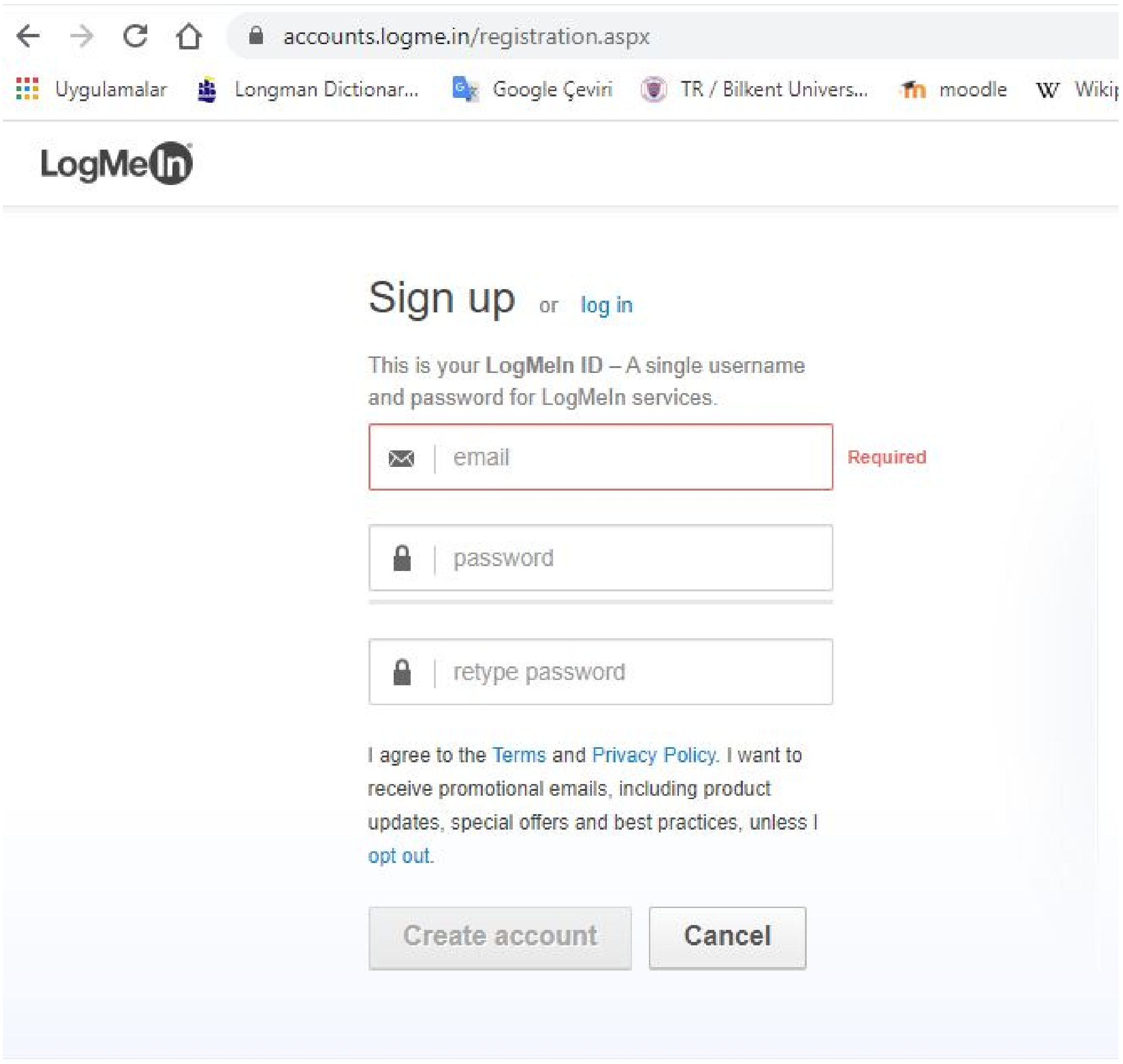


No frameworks detected.

3.2. Connect a LogMeIn Hamachi server

If you and the people you are going to play this game are connected to the same network, you can skip this step; this is required only if the people playing the game are from different networks.

All players should have a LogMeIn account. If you don’t have one, go to the website https://accounts.logme.in/registration.aspx .


accounts.logme.in/registration.aspx

Uygulamalar Longman Dictionar... Google Çeviri TR / Bilkent Univers... moodle Wikip

LogMeIn

Sign up or log in

This is your **LogMeIn ID** – A single username and password for LogMeIn services.

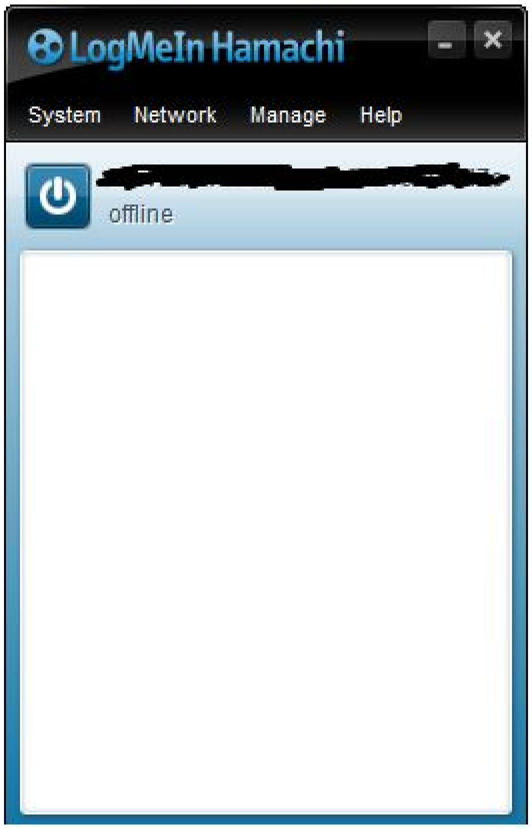
email Required

password

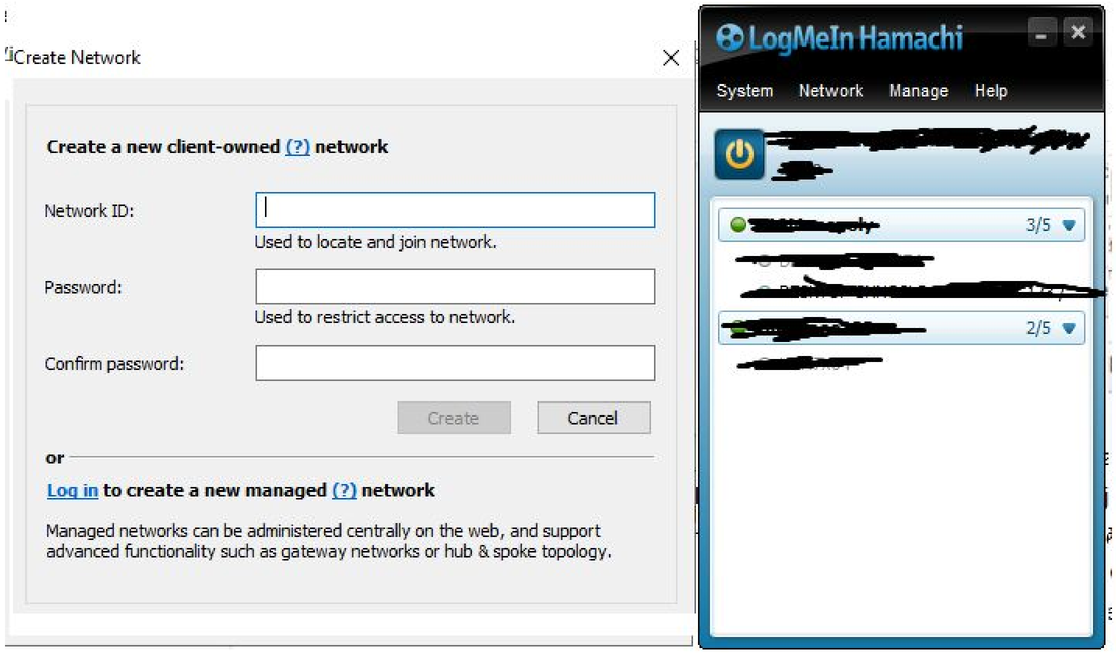
retype password

I agree to the Terms and Privacy Policy. I want to receive promotional emails, including product updates, special offers and best practices, unless I opt out.

Assuming you have installed LogMeIn Hamachi, there you have two options: You can create a network with your LogMeIn account or you can join to a network created by (one of the) people you are going to play the game with.



First, you press the launch button in LogMeIn Hamachi. Then, one of the people creates the network using the panel below, opened by Network > Create a Network tab. There, the host doesn't enter network ID: they log in by using their LogMeIn account.


Create Network

Create a new client-owned (?) network

Network ID:

Used to locate and join network.

Password:

Used to restrict access to network.

Confirm password:

Create Cancel

or

Log in to create a new managed (?) network

Managed networks can be administered centrally on the web, and support advanced functionality such as gateway networks or hub & spoke topology.

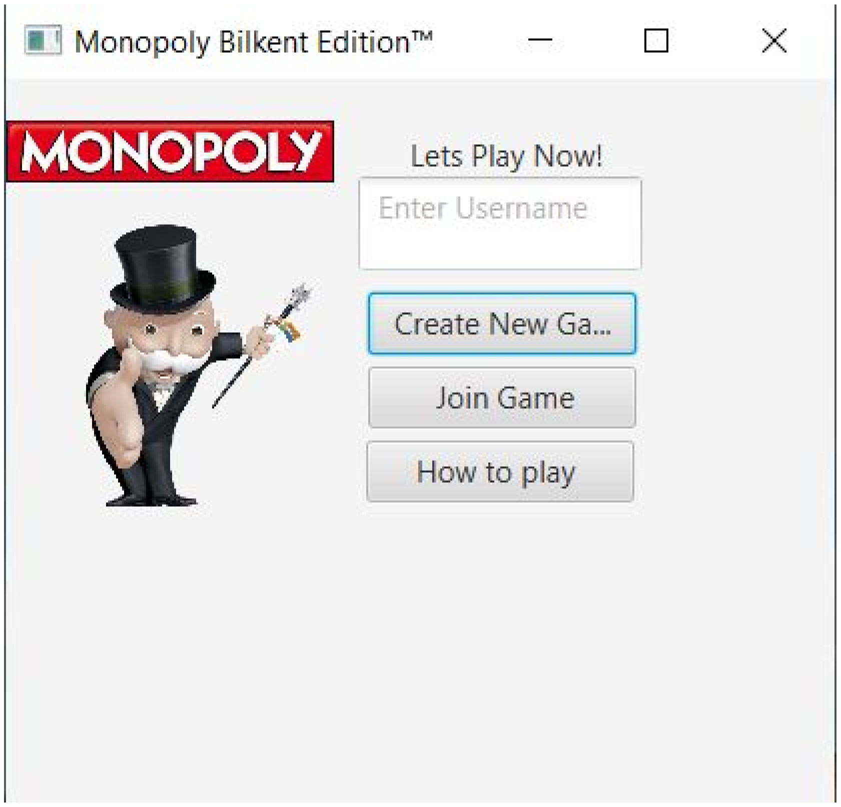
Rest of the players join the game by Network > Join into an existing network. They enter the Network ID and password given by the other player.

After doing all these, the players should see each other’s devices in the app. After that, one of them can be the host and the game PIN host will provide should be working.

3.3. Compile & Run the Project

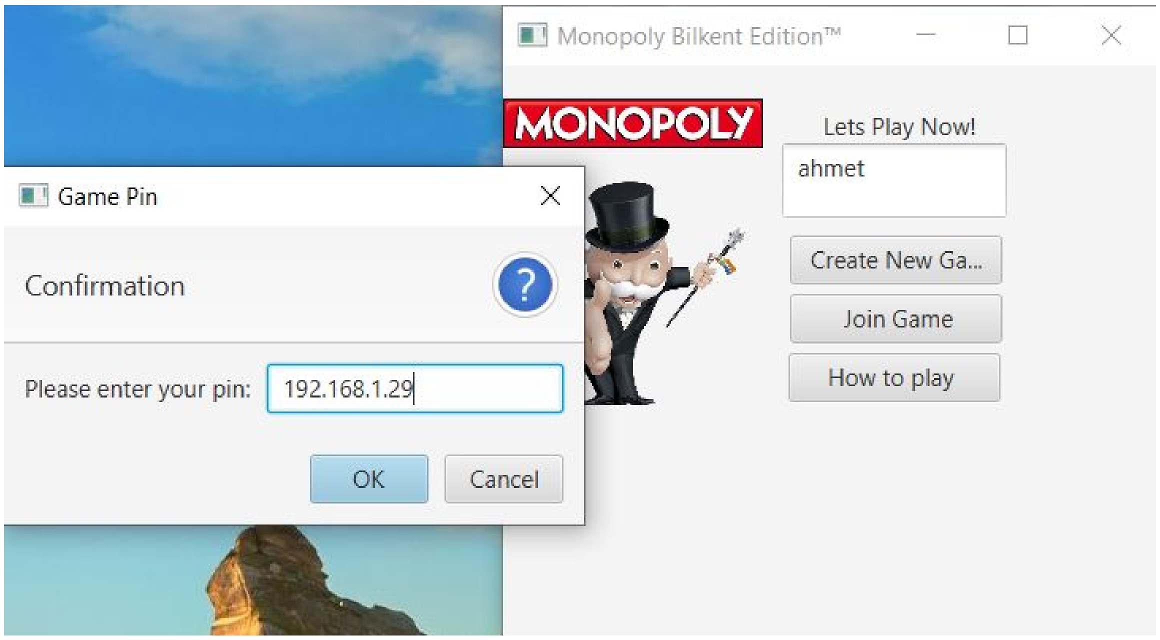
This step is straightforward. Go to Build > Build Project in IntelliJ IDEA and the project should be compiled. Then, from the Project label on the left, select src > gui > Main and from the context menu, select Run ‘Main.main()’.

4. User’s Guide



The main menu of the game is the first screen when the application is run. It currently has three buttons available and a user name field. In order to join or create a game, the user has to enter a valid password (max 8 characters, no new lines).

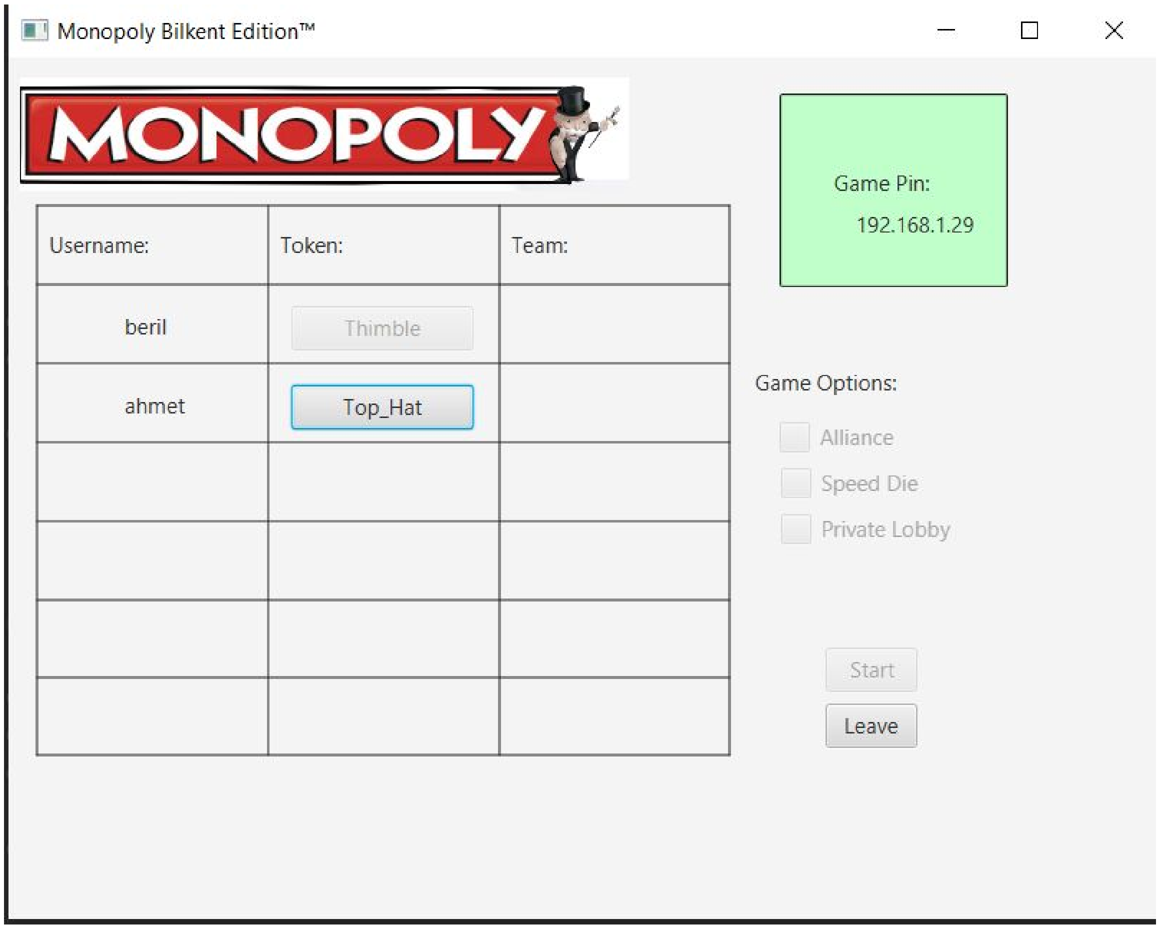
The “Load game” and “Options” buttons are disabled in this version because they are not implemented. The “How to Play” button in the main menu simply redirects the player to a screen with the instructions for the monopoly game.



ahmet

Please enter your pin: 192.168.1.29

After pressing the “Join Game” button, the game asks for a pin. The ip address for the game should be entered here. After pressing “OK” in this dialog, the client connects to the remote server and the player joins the lobby.



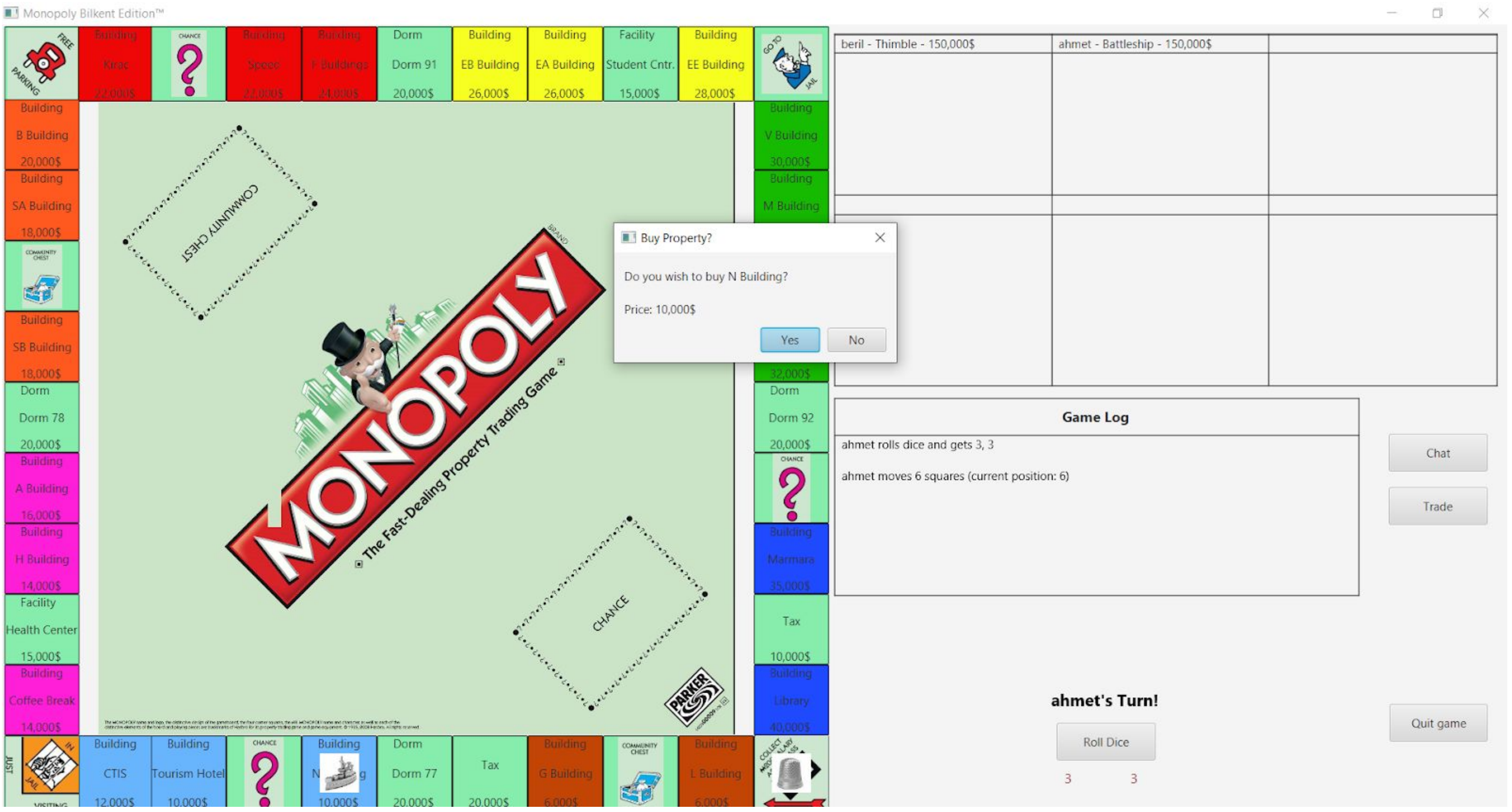
Username:	Token:	Team:
beril	Thimble	
ahmet	Top_Hat	

The lobby screen can be accessed either by simply creating a new game or by joining another game. There can be a maximum of 6 players in a lobby and everyone can only change their own settings. Users can simply press the button that specifies their token to change their token. Team buttons are only visible when the host has enabled the alliance mode in the lobby. A user can also leave the lobby by clicking the "Leave" button on the bottom left side of the screen.

The host has full control over the settings of the lobby. The options checkboxes and the "Start" button on the right are only accessible by the host. "Alliance" and "Speed die" boxes control whether the game has the corresponding game option. The "Private Lobby" box locks the lobby and prevents more players from joining the lobby.

The “Start” button starts the game and changes the scene to game screen for all players. The “Leave” button, when used by the host of the lobby, ends the lobby, causing all of the players to go back to the main menu.

In order to invite other people to the lobby, simply users can share the game pin that is written at the top right part of the screen with others. Other players should use this pin to join the lobby.


Monopoly Bilkent Edition™
FREE PARKING
Building Kirac 22,000$
CHANCE
Building Speed 22,000$
Building F Buildings 24,000$
Dorm Dorm 91 20,000$
Building EB Building 26,000$
Building EA Building 26,000$
Facility Student Cntr. 15,000$
Building EE Building 28,000$
GO TO JAIL
Building B Building 20,000$
Building SA Building 18,000$
COMMUNITY CHEST
Building SB Building 18,000$
Dorm Dorm 78 20,000$
Building A Building 16,000$
Building H Building 14,000$
Facility Health Center 15,000$
Building Coffee Break 14,000$
Building V Building 30,000$
Building M Building
32,000$
Dorm Dorm 92 20,000$
CHANCE
Building Marmara 35,000$
Tax 10,000$
Building Library 40,000$
JUST VISITING IN JAIL
Building CTIS 12,000$
Building Tourism Hotel 10,000$
CHANCE
Building N Building 10,000$
Dorm Dorm 77 20,000$
Tax 20,000$
Building G Building 6,000$
COMMUNITY CHEST
Building L Building 6,000$
COLLECT SALARY AS YOU PASS
COMMUNITY CHEST
MONOPOLY
The Fast-Dealing Property Trading Game
CHANCE
PARKER
Chat
Trade
Quit game
Roll Dice
3
3

beril - Thimble - 150,000$

ahmet - Battleship - 150,000$

Buy Property?

Do you wish to buy N Building?

Price: 10,000$

Yes No

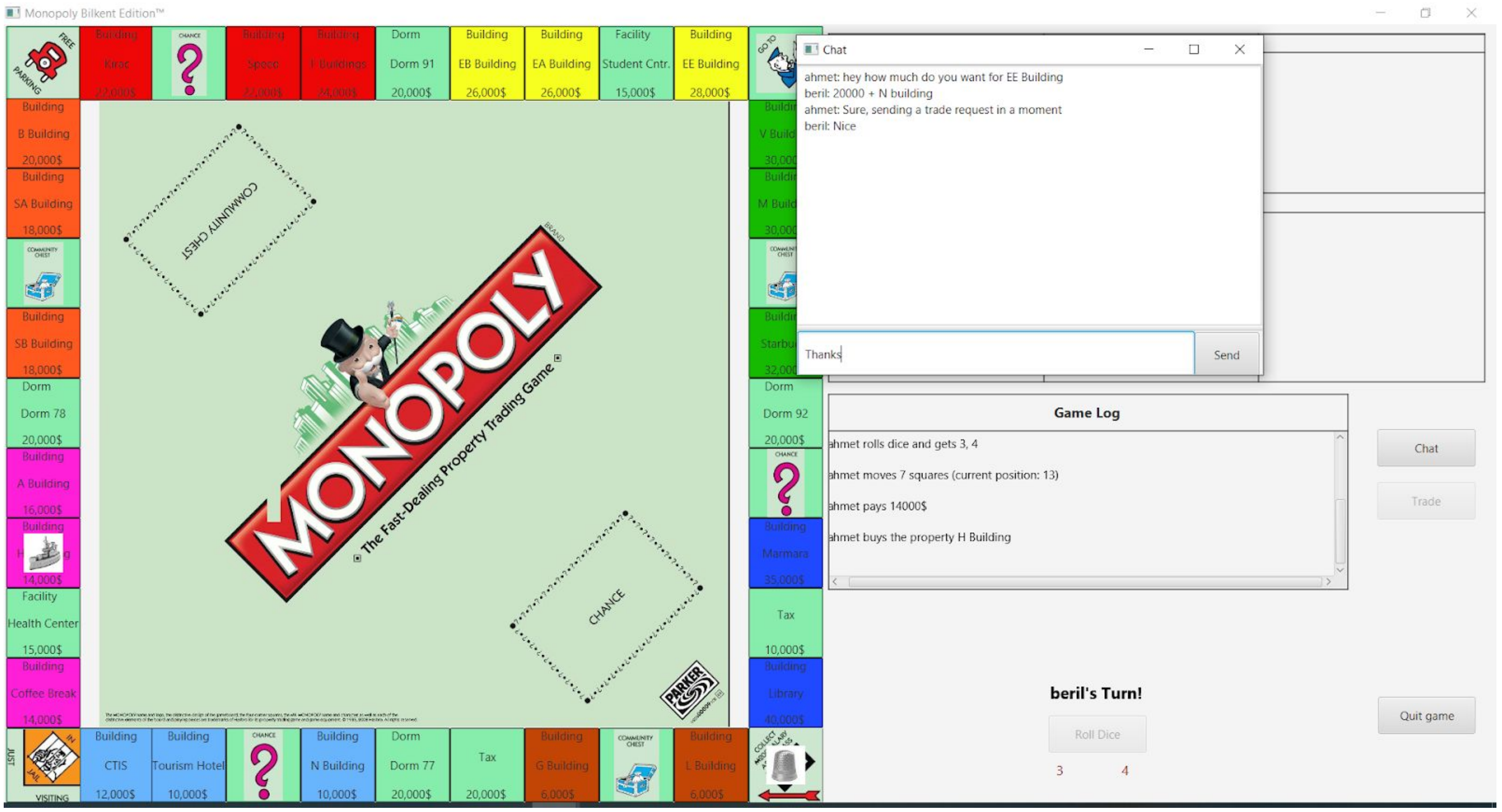
Game Log

ahmet rolls dice and gets 3, 3

ahmet moves 6 squares (current position: 6)

ahmet's Turn!

The game screen has many different areas. On the left, players can see the board along with the tokens of the players and the map. In the middle, players can see the latest actions that happened in the game on the Game Log panel. On the top-right part of the screen the names, tokens, balances, and the properties of all 6 players are listed. Players can also open the chat screen by clicking the “Chat” button. When landed on an unclaimed land, the game asks the player if they want to buy the property. If the player answers “Yes”, then the player buys the property and it shows on his/her property list, else, the player ends his/her turn.

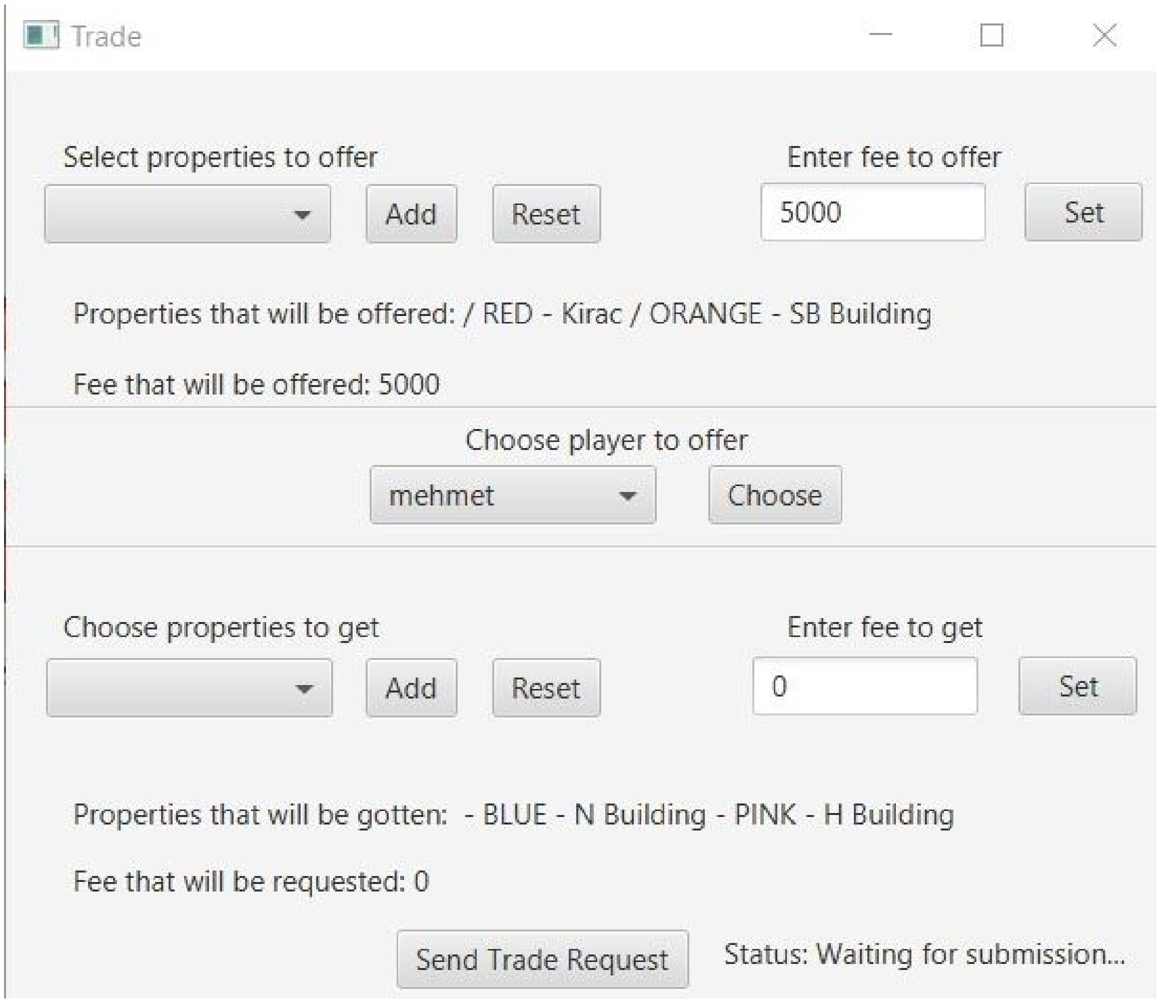


ahmet: hey how much do you want for EE Building
beril: 20000 + N building
ahmet: Sure, sending a trade request in a moment
beril: Nice

Thanks

```
ahmet rolls dice and gets 3, 4
ahmet moves 7 squares (current position: 13)
ahmet pays 14000$
ahmet buys the property H Building
```

If it is your turn, “Roll Dice”and “Trade” buttons are enabled for you. Players have to roll the dice to move and then their turn automatically ends.



5000

Properties that will be offered: / RED - Kirac / ORANGE - SB Building

Fee that will be offered: 5000

mehmet

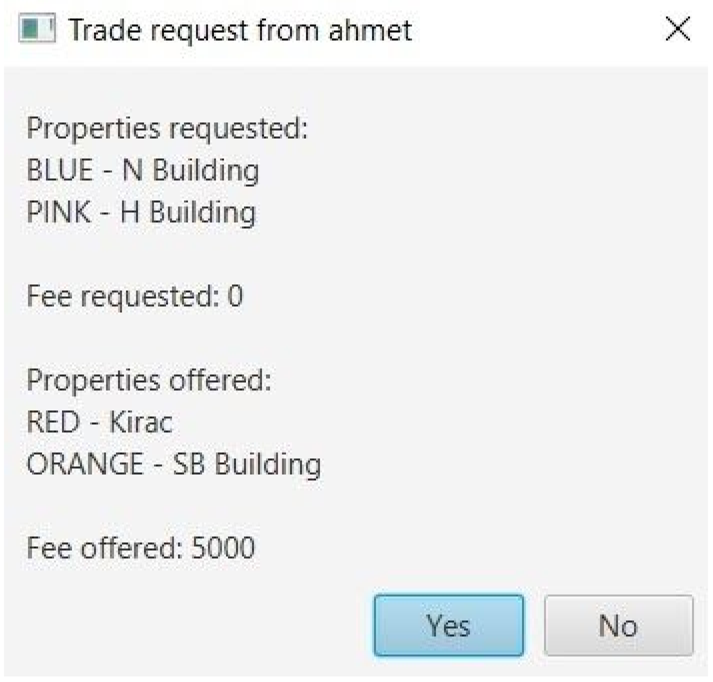
0

Properties that will be gotten: - BLUE - N Building - PINK - H Building

Fee that will be requested: 0

Status: Waiting for submission...

The “Trade” button opens another panel that allows sending a trade request. In this panel, the player designs the trade request as they wish and sends the trade request by clicking the “Send Trade Request” button. After sending a trade request, the other player gets a dialog asking if he/she wants to accept the trade request and the sender has to wait for the other player to answer it.


Trade request from ahmet

Properties requested:
BLUE - N Building
PINK - H Building

Fee requested: 0

Properties offered:
RED - Kirac
ORANGE - SB Building

Fee offered: 5000

5. Work Allocation

5.1. Melike Fatma Aydoğan

- In the Requirements Analysis Iteration I, I worked on functional and non-functional requirements and wrote the overview of the game. Also, in the second iteration, I revised my previous section and created the state chart diagrams and wrote their explanations.
- In the Design Report Iteration I, I wrote some of the Entity Layer Class Interfaces explanations and helped my friends in different sections. In the second Iteration, I wrote the descriptions of some additional interfaces.
- In the code
    - I wrote some parts of the Card, Property, Tile packages and Board class.
    - I wrote the MortgagePropertyAction and UnmortgagePropertyAction classes.

    - I prepared the ChanceCard, CommunityChestCard, Properties and Tiles json files to export these data from the files.
    - I reviewed the codes in the Entity package classes.

5.2. Ahmet Cemal Alıcıoğlu

- In both analysis and the design reports, I was responsible for the object design.
- In the analysis phase, before writing any code, I designed the structure of the code for all of the functionalities specified in the requirements. Everyone then followed my structure to implement the game.
- In the design phase, I created the final object design that covers everything necessary to finish the whole project.
- In the implementation phase, I mostly worked on the controllers of the GUI package. I also helped create some of the visual parts of the GUI and worked on client-gui connection. I implemented the following screens/functionalities:
    - Main menu
    - Lobby
    - Game screen
    - Game log display
    - Property list display
    - Linking the user to property menu using hyperlinks in the property list
    - Server-Client-GUI communication
    - How to play screen

5.3. Zübeyir Bodur

- I dealt with most of the reports:
    - I created the drafts for the reports, kept track of formatting, added figure captions, page numbers etc.
    - I drew all the sequence diagrams (4 in total) in Iteration 1 Analysis Report. I also drew all of the activity diagrams (7 in total), one state diagram and improved the object model in the Iteration 2 Analysis Report.
    - I did the Subsystem Decomposition, Subsystem Services sections and helped with the Class Interfaces in both Design Reports. I also drew the class diagrams in the Final Object Design & Layers, helped with the Class Interfaces in the Iteration 1 Design Report. I also helped with the Final Object Design and Layers in the Iteration 2 Design Report.
    - I did the Introduction, Lessons Learnt and Building Instructions of this report.
- I helped my team with the implementation:
    - dice package
    - Helped with tile package
    - Speed die feature (except Bus face)
    - Helped with trade feature
    - Helped with alliance (teams)

5.4. Beril Canbulan

- In the analysis report, I created the mockup screens. I designed every part of the GUI of the project and visualized them using the Balsamiq Wireframes application
- In the implementation stage, I worked with Ahmet in the GUI package and created the fxml parts for his controller classes, using the SceneBuilder application for JavaFX.
    - Designed the main menu layout
    - Set the boundary conditions in main menu
    - Wrote confirmation and error dialogs in main menu
    - Designed and enhanced the lobby layout
    - Designed the game screen layout
    - Designed the property menu
    - Implemented some property menu functionalities
    - Fixed some bugs on controller classes

5.5. Mehmet Çalışkan

- In the requirement analysis report, drew the Use Case diagram and explained the use cases. Drew all of the sequence diagrams and explained them in the iteration 2.

- In the design report, explained some of the class interfaces. Added the network layer and explained its classes. Added design patterns to the object design.
- Dealt with formats of the reports. Organized the group on drawing diagrams by using the Visual Paradigm Team functionality.
- Dealt with most of the implementation:
    - Reviewed the written code of my teammates.
    - Wrote the Controller and Network layer classes.
    - Wrote the JSON parsers by using the GSON library.
    - Fixed bugs and made improvements on the code.
    - Added Multiplayer, Chat, Trade and some other game logic elements.
    - Built the artifact(JAR) of the game.