UUN: s2002073

Name: Sahar Atif

***Question 1*:** The initial design was composed of minimum number of classes i.e. Location, Environment, Controller and View. Location class had four fields containing view of each side. Most of the fields were static and final so that each view can be located by the class fields directly.

The controller was quite crowded however the design also contained arrays as well. The basic turning was simple iteration of an array in forward and backward direction. It was a working application as I tested it with multiple changes however it was quite harder than the design Paul suggested. Attaching content to a location was also a challenge.

Changes made in the original design:

The original design was changed completely with respect to classes as well as implementation. There was lot more OOP approach as compare to my design. There were multiple classes each having their own functionality. I must pay emphasis on the fact that my previous design was in a working state however it did not have much flexibility as the final implementation. Some of the parts were hard coded but it gave me a heads up to improve the design overlapping with Paul's suggested design.

Firstly, I created a simple Item class and a wall class with its getters and setters. That was pretty straight forward. Second step was to create the Room class which has items and the walls. So, one Hash set and one HashMap was enough to represent it. Afterwards designing the two enum class was pretty straightforward. iRoom was the interface for the Room class however iModel was the main interface of attention. I didn't want the player class to be seen in the Controller so I encapsulated it in World Model class utilizing the iModel as an Interface. This step would help in extending the design to multiple players. The controller is pretty straightforward as it doesn't show the actual implementation components. I created a separate class JasonToJavaObjects to read the json file.

***Question2*:** The code was implemented by myself. I didn't take any sample code from any website. I took ideas from Stack overflow, Oracle docs as well as YouTube videos. There was not much documentation related to this particular assignment however there was a lot of information about the related components. The major difficulty I faced was in pick and drop objects as well as parsing Json file. Moreover, Gradle dependencies and configuration also took some time to fix and understand as working with maven repositories and jars was somewhat new for me. I made a simple enum class but there were two ways to execute the "turn" mechanism. Either by modifying the enum class within it or creating a new class. I tried both of them but keeping the enum class simple I has to create two enum classes to determine the turn.I chose the latter option of making a new enum class for turn which keeps the code quite simple and straightforward. However, the former reduces the number of classes.

There are four rooms connected to each other. Each room has two items but I was able to display one item at a time. However, I was able to move the objects around in different rooms. I also added the drag and drop mechanism because I thought it was pretty cool. I have used Jackson 1.x libraries to access the json file via MyPojo class. Each room had similar components i.e. items and walls. Hence, I tried to create a pojo class by myself and I also tried to take help from some online tools to give me an idea of the pojo class. However, I was not able to map it completely because my classes needed some changes and I didn't want to touch a good working application.

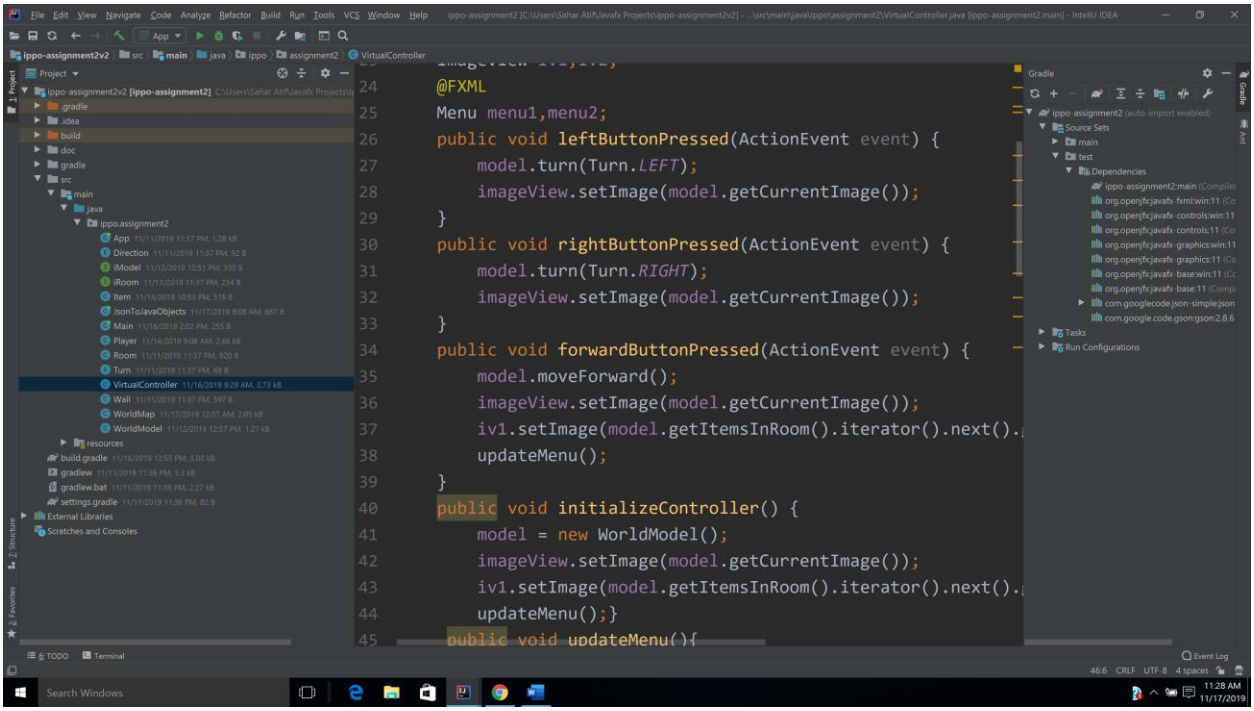
Items Stack: I found imageview creation for each item at run time quite difficult. I tried to create "new ImageView()" at the time of location change so that if there are 5 items in one room it could display all five

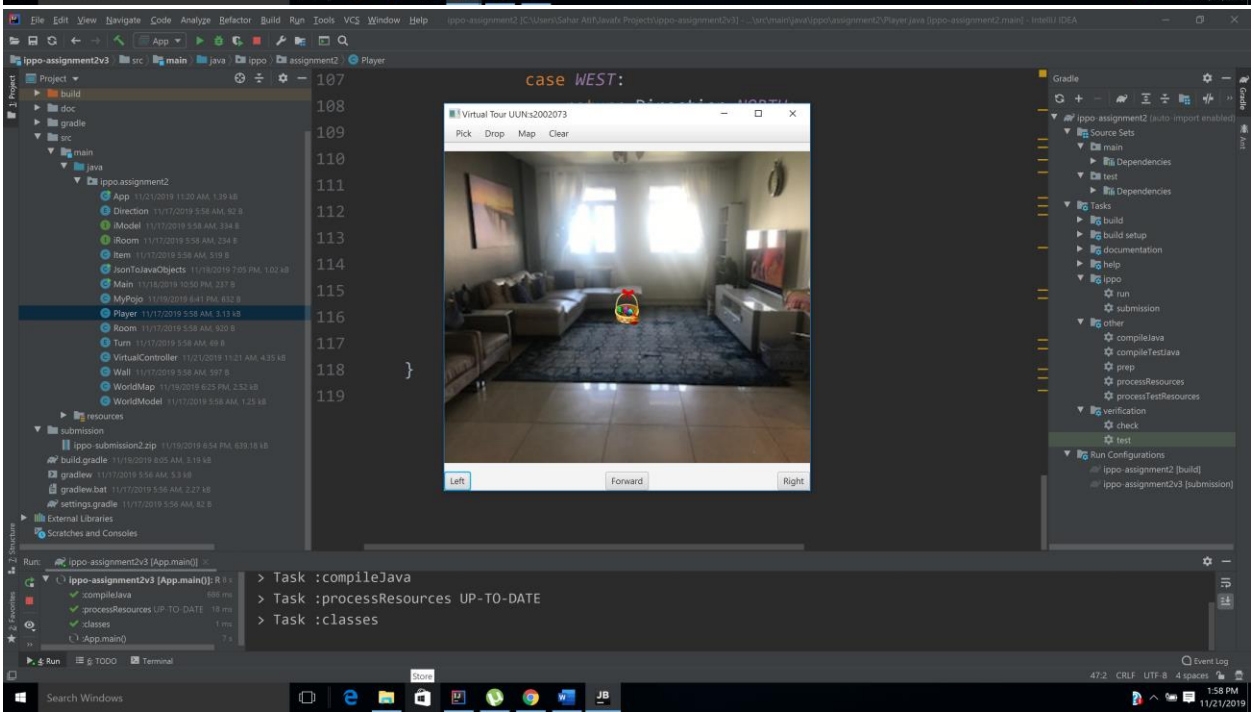
items but somehow I couldn't do implement it however I stacked those items in each room. So practically those items are stacked over each other in one image View and with menu options for their portability.

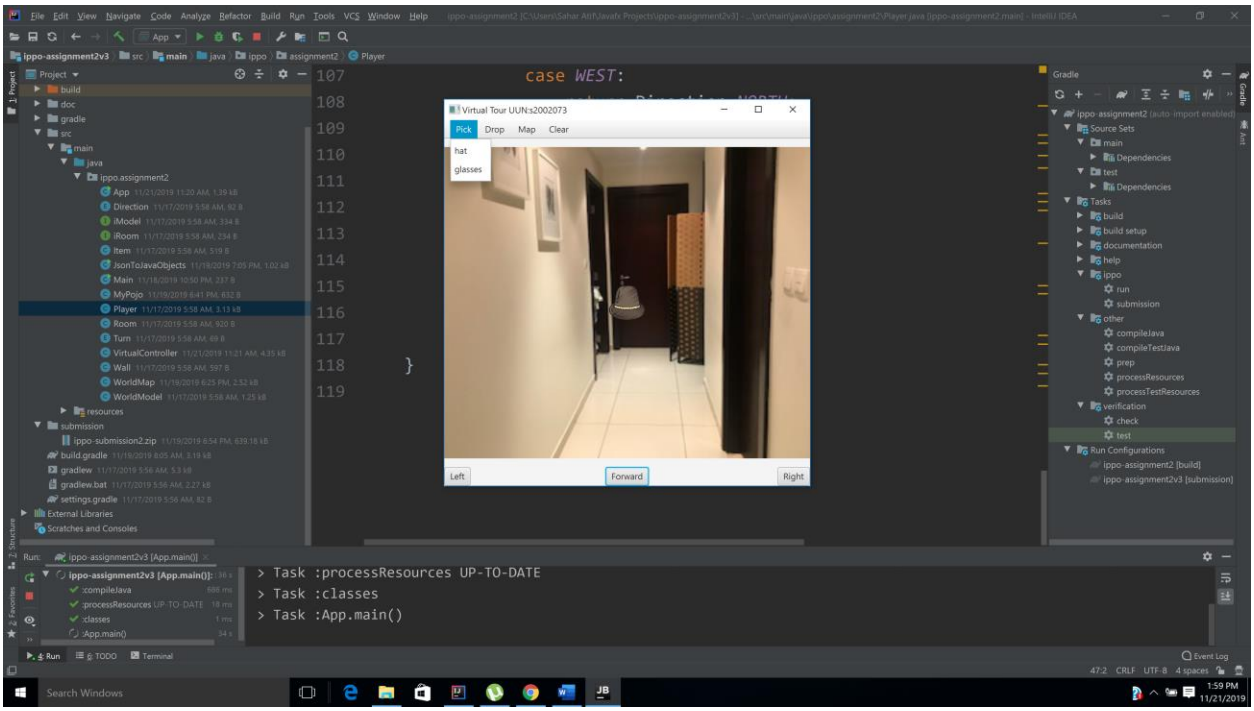
Note: Pick one item and you can see another item stacked behind it and as many items in one room

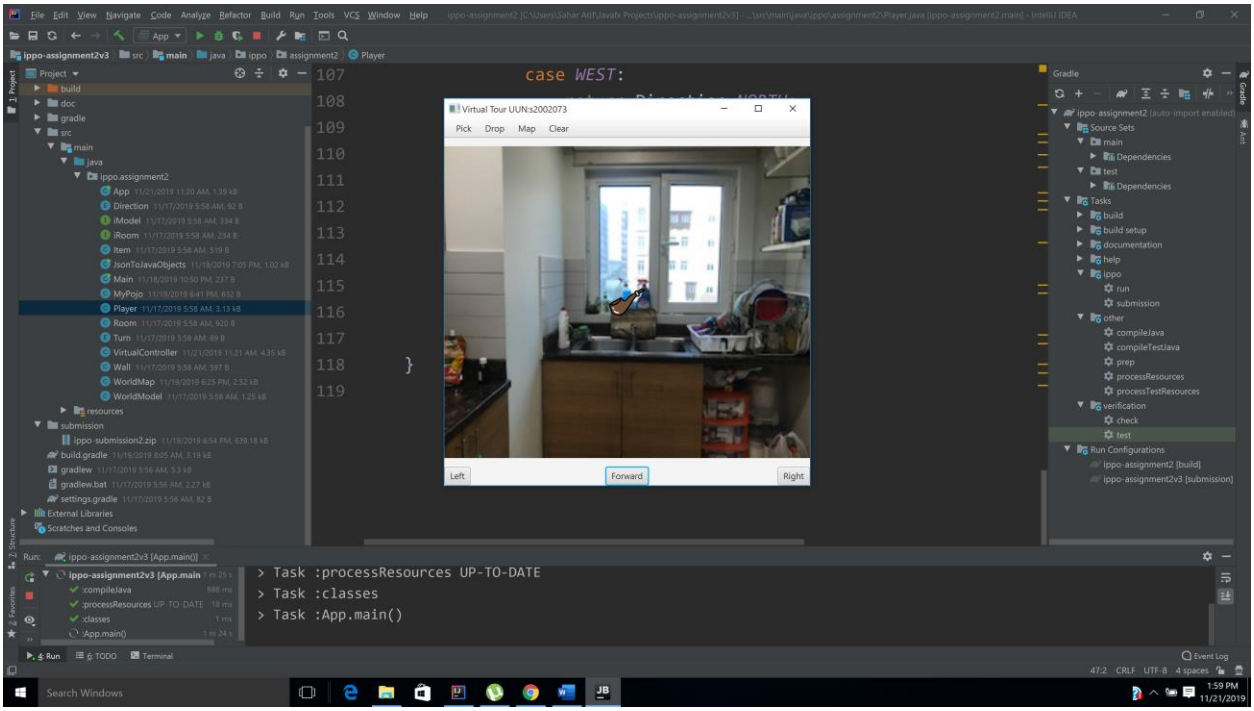
Tasks incomplete:

Proper Json mapping: This part took a lot of time and brain power but though I understood the mapping concept I was not able to properly implement it.






case WEST:
Virtual Tour UUN:s2002073
Pick
Drop
Map
Clear
hat
glasses
Left
Forward
Right
}
> Task :processResources UP-TO-DATE
> Task :classes
> Task :App.main()


case WEST:
Virtual Tour UUN:s2002073
Pick
Drop
Map
Clear
Left
Forward
Right
}
> Task :processResources UP-TO-DATE
> Task :classes
> Task :App.main()