Test covers

Our unit test coverage for any activity is 0%
For controllers and models we have 80-100%
In total for the entire game center folder we have 27% coverage
These are measured in line coverage through Android Studio

Important classes

Our important classes are our:
LogInActivitity
Controllers for slidingTiles, hangman, tic-tac-toe
Abstract classes gameScreenActivity, complexityActivity

Our log in activity is important as it contains all the code required to connect to firebase and validate an input. This class is run at the beginning and controls all access into the app.
Our controllers are important as they contain the logic for their respective game whereas other activities only contain methods to show objects on screen and interact with user - they can be removed and the game can still function as a command line game
Our abstract classes are important as they remove a lot of excess and repeated code and allow all the games to follow a similar format for the parts they are similar.

Design pattern

For each game we chose to a MCV (model controller viewer) pattern.
This allowed us to separate:

- logic of how moves, how the game should start etc into the controller.
- data that the game needs, the guessed word, the selected grid, where the tiles are, into the model.
- code to interact with the user, showing the relevant information, the UI, getting inputs, into the viewer.
-

We can still run the game in command line by removing all the android specific parts.

We chose to use an interface + abstract classes for things the game had in common, such as choosing difficulty and undos, the save/new/load game screen, and setting and controlling difficulty to work with our score and scoreboard.

Scoreboard

Initially we had our scoreboard only show local score but decided to store the scores on firebase as well to have a global scoreboard. To keep track of whether a game is a high scoring like basketball or low scoring like golf we have a method in our Game interface that returns true or false respectively. They're displayed as their own activity and stored when the game is solved.