**How lessons learned were applied**

1. Containment Relationship

This is the class relationship we learned in mini 1. In mini 2, I applied this concept into the final project. Just like in the UI layer, every Activity contains an instance of EntityAdapter in order to realize various functions. There are some entity classes which are also contained in EntityAdapter class as some instance variables.

1. Encapsulation

This is the concept we learned in mini 1. In mini 2, I applied this concept into the entity classes and EntityAdapter class. All the entity classes have private instance variables and public getters and setters functions. All the outside operations about the entity classes can only be operated through the EntityAdapter class. This what I do to realize the concept of encapsulation.

1. The way to analyze data to design Objects.

Here, in the final project, I analyzed that there are 4 entity classes in our project and they are independent to each other. So, It becomes more easier to design the objects.

1. The strategy can be used to design core classes for future requirements.

The strategy I learned in mini 1 is that we just need to expose all of the operations of core classes through one class to the outside, which means that all of the outside operations can be realized through the API provided by one class. So, here, in the final project all of the operation are done through EntityAdapter class. In other words, all of the operations will be exposed to outside through EntityAdapter class.

1. The good conventions for making a java class readable.

In final project and several small projects in mini 2, I have tried my best to make the code well-commented. The instance variables in classes were put in the first part of the class body. And the functions were placed in the second part of the body. I have made the name of the classes and packages more readable. The classes with nearly same characteristics were put into a same package.

1. The advantages of reading data from sources such as text files or databases in a single pass and not use intermediary buffering

we put the persistent into a mysql database SMARTLOCK. This database contains the entities and the relationship between those entities.

1. Advantages of Serialization

I applied Serialization into Entity class in client side and the server side in order to use the ObjectStream to transfer serialized object between client and server.

1. User of Interfaces

The role of interface is just providing you with the declarations of methods. And they didn't tell you the actual implementation of these methods. In this final project, I used interfaces between UI layer and business logic layer. In the interfaces, various functions were defined and the EntityAdapter class, implementing all the interfaces, realized all the functions defined in interfaces.

1. Exposing methods using different interfaces

I created 9 interfaces between UI layer and business logic layer. Each of these interfaces represent different functionalities. So, the maintenance of these interfaces will become much more convenient.

1. Used static object of Information

I used a static object of Information class in EntityAdapter class. Using this static object we can store some temporary data without need to communicate with database.

1. I have implemented the CRUD operations in EntityAdapter class. By using these CRUD operations, the outside can communicate with inside entity classes.
2. Database Optimization

I applied the 3 rules of database optimization into the database schema design in the final project. There is no repeated columns in the tables and I didn’t insert any field that doesn’t exist in tables. And when deleting the records, I designed to delete the referenced record first and then delete the current records.

1. Servlets

We design our backend by running serlvets on tomcat, which can used for send response to the request coming from client side.

1. Response in Servlet

We used response to get the outputstream of object in order to send the object back to the client side.

1. Request in Servlet

We used request to get the inputstream of object in order to get the object sent from client side.

1. Post method

In our final project, we always use post to send the object to the server side. Comparing with get method, post method can transfer a large amount of data into server side.

1. Customized Exception Handling

In the final project, we used customized exception handling method to detect the network connection exceptions.

1. ArrayList

We used ArrayList to store multiple objects in the Information class in final project.

Such as List of properties and list of locks.

1. Object-Oriented Programming in our final project. Object-Oriented Programming is an approach that provides a way of modularizing programs by creating partitioned memory area for both data and functions that can be used as template for creating such modules on demand.
2. Static variables are used when we want to have a variable common to all instances of a class. In our final project, we used static object of Information class, which is common to various classes in this project.
3. Listview is generally used by writing an activity (extending android.app. ListActivity).

The data association with List control is set usingsetListAdapter() method.

1. setListAdapter has smarts to take rows from data source and pull out the name of each contact and populate the user interface.
2. Keep in mind that adequate permissions need to be set to read / write data from a data source in AndroidManifest.xml file.
3. setContentViewis called to set the user interface. It sets the layout for each item at the time of Adapter instantiation.
4. GridView enables data display in form of a Grid. It can display data, text, images etc.
5. All application resources should be externalized in order for maintainability .

1. Android automatically applies the appropriate resources by matching the device’s current configuration to your resource directory names.
2. Table, Linear and Relative Layout each has one way to position and orient multiple widgets on screen.
3. ContentProvider manages structured set of data, which also provides mechanism for defining data security and provides mechanism for performing data access to clients.
4. Views are user interface (UI) elements that form the basic building blocks of a user interface.
5. An activity is a user interface concept. An activity usually represents a single screen in your application. It generally contains one or more views, but it doesn’t have to.
6. An intent generically defines an “intention” to do some work. Intents encapsulate several concepts, so the best approach to understanding them is to see examples of their use.
7. Local services are components that are only accessible by the application that is hosting the service
8. Remote services are meant to be accessed remotely by other applications running on the device.
9. Signed digital certificate on app development Prevents application from being updated with a new version that original author may not have published.
10. On 32 bit processors data should be **aligned** on a 4- byte memory boundary – entire data set in .apk file looks like data in memory
11. Applications continue to run if certificate expires as Certificates are checked at install time.  So life of certificate should be long enough to apply an update.
12. Showing a datepicker creating the **DatePickerFragment** class and building an **AlertDialog** and getting the dialog on screen via the **FragmentManager .**
13. LocationManager service enables to obtain device geographical location and enables notification facility when device enters a specified geographical location.
14. Steps for calling the remote service: create a new android project named StockQuoteClient, create a new java package in this project named com.example.stockquoteservice.
15. AIDL allows you to pass other AIDL interfaces, but you need to have an import statement for each AIDL interface you reference (even if the referenced AIDL interface is in the same package).
16. Nonprimitive types, other than String, require a directional indicator. Directional indicators include in, out, and inout. in means the value is set by the client, out means the value is set by the service, and inout means both the client and service set the value.
17. BluetoothSeverSocket – use to open and listen for incoming connections. Instantiates BluetoothSocket object when a connection is made.
18. BluetoothSocket class is used by client to make a connection to a remote device. Used by both client and sever to handle connection and manage input/output streams.
19. BluetoothDevice – represents the hardware on remote device.
20. Use intent named BluetoothAdapter.ACTION\_DISCOVERABLE using startActivityForResult method. User is presented with an option to make device discoverable for a set period.
21. Use intent named BluetoothAdapter.ACTION\_REQUEST\_ENABLE using startActivityForResult method. Return value RESULT\_OK indicates Bluetooth enabled.
22. each device has a valid BlueToothSocket Objects to retrieve InputStream and

OutputStream objects for initiating data communication.

1. Device that acts like Server uses BluetoothServerSocket  class. It makes a blocking call using the accept() method.
2. Device that acts as Client uses BluetoothSocket class and calls connect() method – A blocking synchronous call.