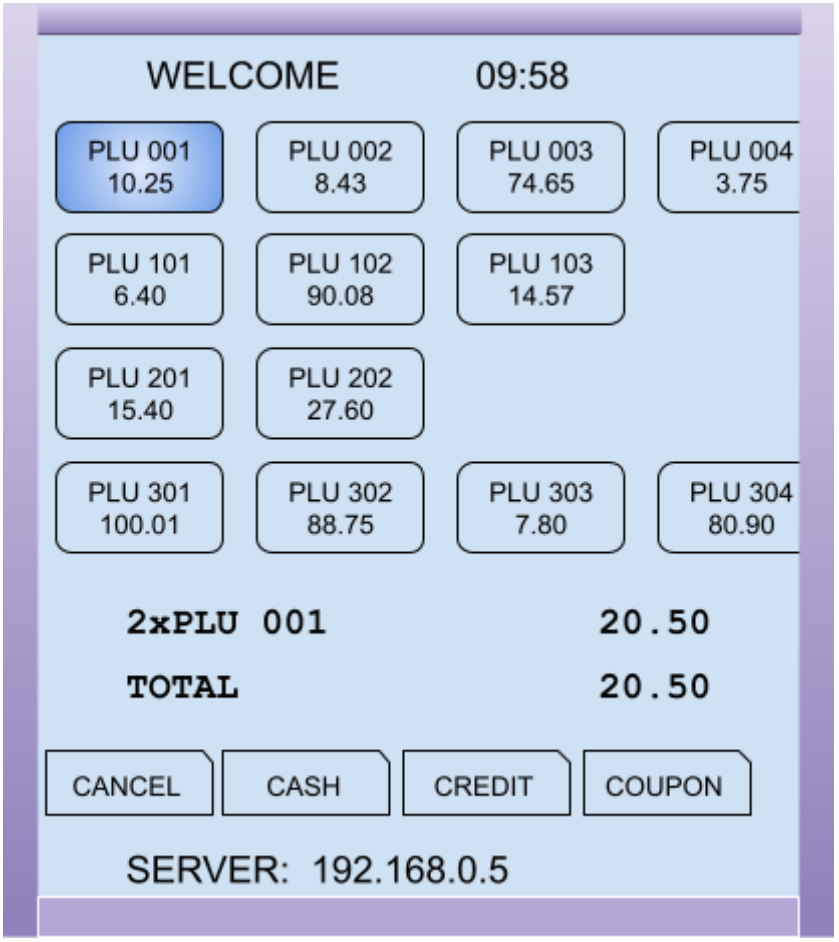
Hourly Sales Report - Merged Phases Requirements

In this set, you are required to develop one client application running on an Android device and one java desktop service.

The client application specifications;

- ☐ Summary
  - ☐ Developed with Kotlin, use compose for ui design.
  - ☐ The client application has an ui to sell products.
  - ☐ The client application sends sales hourly to the Server
- ☐ The client database : “retail.db”
  - ☐ The client application uses “retail.db”. Find a sample database in attachments.
  - ☐ The client application reads properties of the products from the “product” table in the “retail” sqlite database. The fields of the “products” table are “id”, “name”, “vatRate” and “price”
  - ☐ The client application saves sales to the “receipt” table in the “retail” database. The table has fields of “receiptNumber”, “receiptDateTime”, “amountVat0”, “amountVat1”, “amountVat10”, “amountVat20”, and “paymentType”.
    - **receiptNumber** starts from 1 in an hour. And each record increases by 1. For example;
      - The first receipt of hour 10 is completed at 10:05, its number is 1.
      - The next receipt is completed at 10:17, its number is 2.
      - The next receipt is completed at 11:02, its number is again 1. Because the hour of the time is changed
    - **receiptDateTime** is the date and time information of the system when receipt is completed. Format is yyyy-MM-dd HH:mm. For example, If the first receipt is completed at 10.02.2024 10:05, then the receiptDateTime of the record should be “2024-02-10 10:05“
    - **vat0Sales** is the total sale amount of the products whose vat rates are 0
    - **vat1Sales** is the total sale amount of the products whose vat rates are 1
    - **vat10Sales** is the total sale amount of the products whose vat rates are 10
    - **vat20Sales** is the total sale amount of the products whose vat rates are 20
    - **paymentType** is the method of payment performed for the receipt. The payment methods can be 0, 1, 2, 3.
      - 0 : receipt is cancelled
      - 1 : receipt is paid by Cash
      - 2 : receipt is paid by Credit
      - 3 : receipt is paid by Coupon

- [ ] The user interfaces and the flows of the Client application
    - [ ] The example screen below is to show only locations of the Title, product buttons, summary labels and payments. It is an extra point to prepare influential compose ui.



    - [ ] The application shows products on the screen when it is launched. Use 4x4 matrix. If the number of the products of the corresponding vat rates is smaller than 4, show only available products.
    - [ ] Where each row shows the 4 products, and in a row the user can scroll right-left to see other products.
        - [ ] The first row is for the products with vat rate of 0
        - [ ] The second row is for the products with vat rate of 1
        - [ ] The third row is for the products with vat rate of 10
        - [ ] The fourth row is for the products with vat rate of 20
    - [ ] When the user clicks a product button, the application prompts a screen which shows quantity and price.
        - [ ] The default value of the quantity is 1
        - [ ] The default value of the price is the value of the price field in the “product” table in the “retail” database.
        - [ ] The user can edit these values.
            - [ ] Quantity is allowed 1 to 99

- Price is allowed 0.01 to 999.99
- When user submits the prompt screen, a sell item with entered quantity and entered price is inserted to basket

- When an item is added to the basket, update the basket summary section of the screen. The basket summary section is just below the products. It has two rows: The first row is for the last sold item, The second row is for the total amount of the basket.
  - Show the quantity, name and amount in the first row of the basket summary section as shown in the example screen.
  - Show the basket total in the second row of the basket summary section as shown in the example screen.
- Below the basket summary section, there is a payment section.
  - There is a “Cancel” button to cancel the reciept
  - There is a “Cash” button to pay the receipt by cash
  - There is a “Credit” button to pay the receipt by bank or credit card
  - There is a “Coupon” button to pay the receipt by coupon
- When user clicks any buttons in the payment section, the client application saves a “receipt” record to the “receipt” table in the “retail” database.
  - Use the order of the receipt in that hour as receiptNumber
  - Use the current date time of the system as receiptDateTime (format as yyyy-MM-dd HH:mm)
  - Use the sum of the items of vat 0 in the basket as “**vat0Sales**”
  - Use the sum of the items of vat 1 in the basket as “**vat1Sales**”
  - Use the sum of the items of vat 10 in the basket as “**vat10Sales**”
  - Use the sum of the items of vat 20 in the basket as “**vat20Sales**”
  - Use the selected payment method as paymentType

- The client application uploads sales hourly. To achieve this, the client application prepares an hourly report using the “receipt” table in the “retail” database.
  - Hourly report is an xml, which contains followings

```
<hourlyReport>
<reportHour>yyyy-MM-dd HH</reportHour>
<receiptCount>the number of receipts completed in corresponding hour</receiptCount>
<totalAmount>total sale amount of the receipts completed in corresponding hour, do not include cancelled receipts</totalAmount>
<canceledInfo>
<count>number of cancelled receipts</count>
<amount>total amount of cancelled receipts</amount>
</canceledInfo>
<salesVatDistribution>
<sales>
<vatRate>0</vatRate>
<amount>amount of sales of the vat rate of 0 in corresponding hour, do not
```

```
include cancelled receipts</amount>
</sales>
<sales>
<vatRate>1</vatRate>
<amount>amount of sales of the vat rate of 1 in corresponding hour, do not
include cancelled receipts</amount>
</sales>
<sales>
<vatRate>10</vatRate>
<amount>amount of sales of the vat rate of 10 in corresponding hour, do not
include cancelled receipts</amount>
</sales>
<sales>
<vatRate>20</vatRate>
<amount>amount of sales of the vat rate of 20 in corresponding hour, do not
include cancelled receipts</amount>
</sales>
</salesVatDistribution>
<paymentDistribution>
<payment>
<paymentType>Cash</paymentType>
<amount>Total Cash payments in corresponding hour</amount>
</payment>
<payment>
<paymentType>Credit</paymentType>
<amount>Total Credit payments in corresponding hour</amount>
</payment>
<payment>
<paymentType>Coupon</paymentType>
<amount>Total Coupon payments in corresponding hour</amount>
</payment>
</paymentDistribution>
</hourlyReport>
```

- ☐ Hourly report time is checked periodically and after each receipt is completed. If the report for the previous hour is not uploaded, the report is sent to server
  - ☐ If basket is empty, check the report upload requirement in each 60 seconds (this is periodic requirement)
  - ☐ If the basket is not empty, do not check the report upload requirement in each 60 seconds. Instead, check the report upload requirement after receipt is completed (cancellation or any payment method is selected)
- ☐ Hourly report contains only sales of the previous hour in the current hour. It does not contain sales of the current hour.
- ☐ Hourly report for an hour is sent once. Do not send the hourly report of an hour more than one.
- ☐ The server ip is displayed at the bottom of the screen. The user can edit the server ip.
- ☐ The port is 4478.

- ☐ Connect the server via http protocol and send the content of the report xml.
    - ☐ Display message during the connection.
    - ☐ Display a message when a response is received. For example, the message can be “server response code is 200” for successful responses.

The server application specifications;

- ☐ Summary
    - ☐ The server application is implemented in Java language (Java 1.8)
    - ☐ The server application can be either console or form application
- ☐ The communication details
    - ☐ The server starts an http server
    - ☐ The server uses a local ip address and 4478 port for http url. For example, if the local address is 192.168.0.5 than url is http://192.168.0.5:4478
- ☐ Connection
    - ☐ When a client is connected to the server, the server informs the user about connection. If the server has a UI, it can use the UI to inform the user about connection. If the server is an console application, it can prints a message on the console
    - ☐ The server reads the content of the http request.
    - ☐ The content of the report is saved to the file where the value of the “reportHour” tag is used as the file name.
    - ☐ After saving the report, the server returns a response with code of 200, OK.

Note: There is also retail_2.db to test your application not dependent on “retail.db” content. First, test your application with “retail.db”. Then, You can move retail.db somewhere and you can rename “retail_2.db” as “retail.db” to work with retail_2.db content.

Best Regards