1. Description of the problem scenario

Seekers-R-Us is a non-profit organization engaged in social welfare through spiritual upliftment of the members/citizens. The organization is spread across the globe and holds meetings at its major centers throughout the year. Each meeting is attended by 200,000 to 300,000 visitors. The major centres of Seeker-R-Us have the necessary physical infrastructure to cater for boarding and lodging, food and beverages and other items of daily basic needs. These items are sold at the points of sale (PoS) which are spread across different buildings in each center. Each center roughly has about 4-5 dozen PoS stalls. The PoS stalls accept payment in the form of paper based cash coupons which a visitor buys from designated stalls/kiosks in the center.

There are several problems with the usage of paper based cash coupons for transacting at the PoS stalls:

1. They are easy to fake and thus they cause significant losses for the Seekers-R-Us.
2. Tracking transactions at PoS stalls is not possible with paper coupons.

Seekers-R-Us would like to introduce e-Cash by using simple smart cards based system in order to replace the paper based cash coupons. Fig. 1 shows the proposed system's architecture.

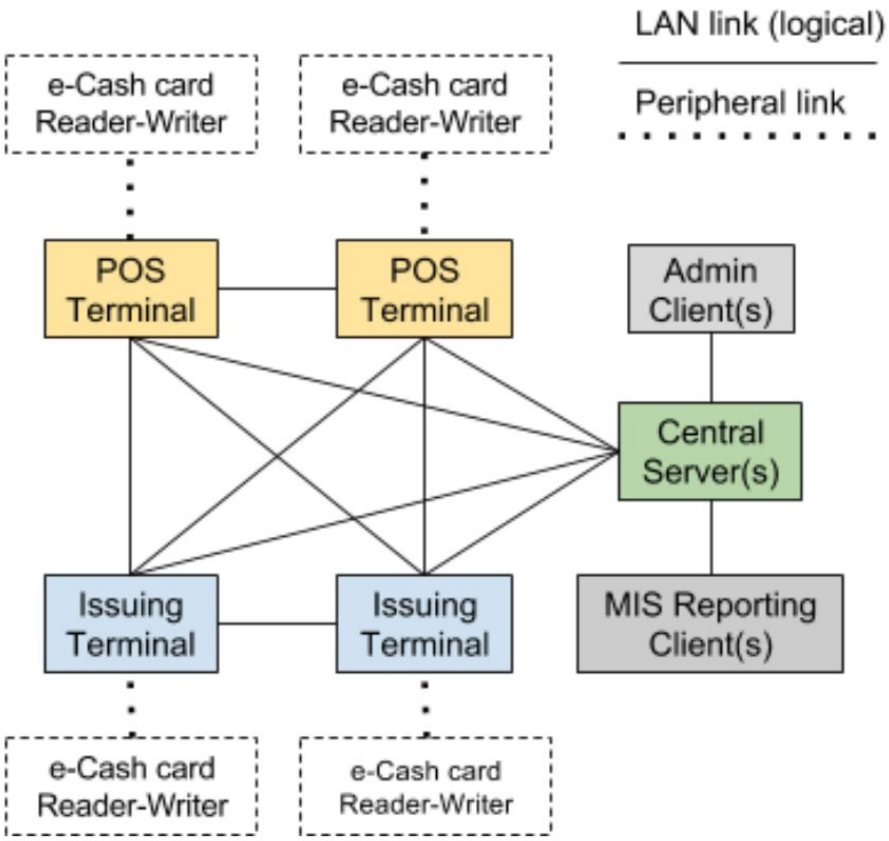


Fig. 1 (Architecture of the smart cards based e-Cash system.)

The following protocol describes the steps that are performed when transacting with the e-Cash cards at the PoS terminals (which are PCs).

1.1 Assumptions (Refer to Fig. 1 also)

1. POS and card issuing terminals are regular PCs/laptops with a card reader-writer attached.
2. POS terminals, card issuing terminals and a back-end server are connected in a logical peer-to-peer topology.
3. Every node on the peer-to-peer network maintains a card balance register (CBR) that has following information for each valid card:
    a. Current cash balance.
    b. Details (e.g. location, amount, time etc.) of last n transactions where value of n can be configurable, e.g. n = 10 may be a reasonable value. Basically, the value of n should be small enough to avoid huge memory/storage needs.
4. Users are not willing to use any PIN/password for the card.

1.2 High-level steps of the card handling protocol

1. A user goes to a node, K, (node could be a card issuing kiosk or a POS terminal), and transacts with (or obtains) a card C.
2. Node K makes an entry reflecting the transaction of C in its CBR.
    a. *Updating on-card balance*: The node also writes the current balance and few more details (e.g. details of last transaction) to the card. The information is encrypted before writing to card.
3. Node K broadcasts the transaction details on peer-to-peer network. Broadcast communication is done asynchronously.
4. Every node on the peer-to-peer network then accordingly updates its own copy of the CBR.
5. The backend server node which is also part of the peer-to-peer network maintains more information about the transaction in addition to the CBR:
    a. Information (location, timestamp etc.) about the transacting node.
    b. Rest of the details of transaction.

2. You are required to do the following

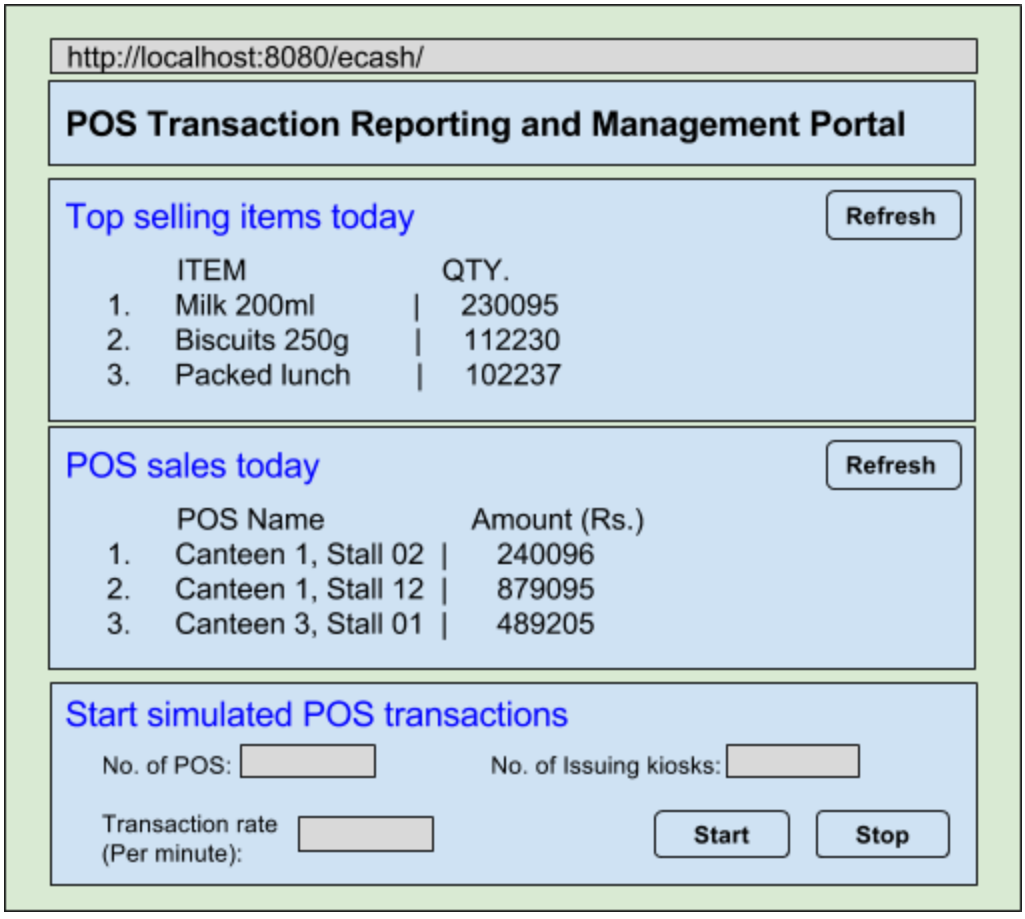
Implement the system depicted in Fig. 1 such that:

1. The program that will run the protocol described in Section 1.2 in each of the nodes of peer-to-peer network shown in Fig. 1. You will also need to have a suitable database running on each of the nodes. You may use SQLite (https://www.sqlite.org/) for the database.

2. You provide a simple web page (you can omit the login etc.) which allows performing the following tasks:
    a. Showing highest selling items across all POS stalls.
    b. Showing daily sales amount per POS stall.
    c. Triggering the generation of simulated user transactions.

    The simulation of the full system is run by starting the protocol program in isolated processes on a single machine. Some of the processes will act as e-Cash issuing nodes while others will act as POS terminal nodes etc. as per Fig. 1. The number of each type of nodes can be user specified (see mock screen up below). Each process should have its own instance of SQLite DB. The simulated purchase transactions are fed into each node by another process which will be generating transactions to purchase randomly selected items.

    The page may look as shown below.



When running the simulation you can assume the following:

Available items for selling:

Item	**Rate (Rs.)**	**Daily available inventory per POS**
Cup of milk 200ML	10	50000
Biscuits (200g)	10	25000
Cup of tea	5	60000
Packed lunch	25	30000
Mineral water bottle	12	50000
Banana 1pc	4	20000
Lunch meal	30	50000

Number of POS terminals:

Building No.	**No of POS terminals**
Canteen #1	15
Canteen #2	10
Canteen #3	20

You should make use of Vert.x API to implement the solution. A relevant starting point could be: http://vertx.io/docs/vertx-core/java/#_writing_http_servers_and_clients

Important instructions about your submissions

- Important instructions for coding submission are here: https://goo.gl/IMWvdF
- Grading scheme to be followed is available here: https://goo.gl/52D82g
- Problem description may be underspecified to allow some room for exploration and creativity.
- Your submission should be packaged as a zip file named **exactly** in this format: CSL456-[your entry no.]-[QUIZ3].zip.