**SCHOOLABLE INVOICE SYSTEM**

The information here contains the business logic and designs that will underlie the schoolable invoice system. The first section derives all the formulas that will be used in the computation and development of invoice system. The rest of the sections contains the algorithms, architectural diagrams and Entity relationship diagrams to aid in the thinking process.

**FORMULAS AND EQUATIONS**

*The total invoice amount is simply:*

***1. Percentage based Discounts***

*Given a list of products and discounts (on those products) on an invoice, the total discounts for this invoice is computed as:*

*If we view the list of of products and discounts as vectors, then the above formula is a* ***dot product:***

*The total payable amount on an invoice is computed by (first) removing all discounts applied on each product:*

*Equation (3) can be expanded and simplified as follows*

*Rearranging and grouping like terms, we arrive at the following:*

*If discount K is applied to the entire invoice amount, then the total discount amount M will be computed after all product-based discount is subtracted:*

*Hence, the payable amount N is:*

*Complete substitution will yield:*

1. ***Flat Fee based Discounts***

*For a flat fee based discounts, we begin be modifying equation (4) as follows:*

*However, we can make things easier (and unified) by expressing the flat fee F into percentage x as follows:*

*Equation (8) shows us how to convert a flat fee into percentage of the original amount.*

**ALGORITHMS**

**Algorithm 1: Total Payable Amount**

***Step 1:*** *Load list of products and discounts from db*

*Let P, D, be list of products and discounts (on those products)*

***Step 2:*** *Compute product based discount*

*For each :*

*if (d is a flat fee discount) then*

*Convert to percentage discount*

*n += p \* (1 - d)*

***Step 3:*** *Compute overral Discount (if available):*

*Let k = discount applied to whole invoice amount*

*n \*= (1 - k )*

*return n*

*Note: If k = 0, then there was not discount applied.*

**Algorithm 2: Invoice Generation**

***Step 1:*** *Get invoice from db*

*Let L be list of invoice loaded from db*

*Let P be list of products on each invoice*

*Let D be list of discounts*

*Let A = list of invoice payable amount*

*For each :*

*payableAmount = N(p,d)*

*Add payableAmount to A*

***Step 2:*** *Generate invoice items:*

*For each product in Products*

*Add product item to invoice*

*Add description to item*

***Step 3:*** *Aggregate invoice items:*

*Include invoice items*

*Include totalPayableAmounts*

*return invoice*

**ARCHITECTURAL DIAGRAM**

Repository

Invoice Core

Invoice Generation

Wallet event Listener

Invoice Settlement

INVOICE SYSTEM

**ENTITY RELATIONSHIP DIAGRAM**

Invoice Order

Products

Owner

Customer

Discount

Invoice

Transactions

Invoice Period

M

1

1

M

1

1

M

1

M

M

1

1

1

1

**Invoice**

- id

- ownerId

-customerId

- totalAmountPaid

- totalDueAmount

- outstandingAmount

- serviceCharge

- invoicePeriodId

- isCompleted

- organisationId

**Invoice Owner**

- id

- name

- logo

- organisationId

**Invoice Period**

- id

- issueDate

- dueDate

- term

- session

- organisationId

**Customer**

- id

- name

**Invoice Order**

- id

- productId

- discountId

- invoiceId

- organisationId

**Invoice Transactions**

- id

- invoiceId

- amount

- organisationId

**Discount**

- id

- description

- type (flat or percent)

- organisationId

**Products**

- id

- name

- organisationId

**End Points**

*- Create Invoice*

*- Add Items to invoice*

*- edit invoice items*

*- Generate Invoice*

*- Invoice settlement*

*- List all invoices (belonging to an organisation)*