Lee Yu Cong - Project Portfolio

PROJECT: CLInic

Introduction

Hello, I am Yu Cong. I am currently studying Computer Science as my major in the School of Computing, National University of Singapore. I formed a team with 4 other students to create CLInic as part of the project requirement under the CS2103T module (Introduction to Software Engineering) as part of my university education.

This project portfolio serves to document my contributions to this project. You can see my contributions to the code contribution, the user guide and the developer guide below.

CLInic is a desktop application made for clinics who wish to automate their daily operations. The CLI in CLInic is an acronym for Command Line Interface, which is the main method to use this application. Unlike traditional address books, the doctor or clinic receptionist can easily find patient or medicine information by entering one simple line in the command box. You can also manage patient and medical records easily with the easy-to-use CLinic.

Examples of the main features include:

1. Efficient management of patient and medicine information
2. Realistic model of clinic's daily operation and commands to cater to clinics.
3. Customised receipts, medical certificates and referral letters for different clinics.

Summary of Contributions

In this section, you can find the major and minor enhancement that I made to CLInic:

- **Major enhancement**: Implementation of **medical records in CLInic**
  - What it does: It allows the clinic to keep track of medical stocks in the clinic.
  - Justification: The medical stocks are managed and the useful commands can organise and find the information of a medicine.
  - Highlights: With a simple command, you can easily list out all medical stocks that are low in stock.
- **Minor enhancement**:
  - Rewrote command output to make it more user friendly.
- **Code contributed**: [Functional code] [Test code]
- **Other contributions**:
  - Wrote the utilities for test cases for other members to build their tests upon.
  - Proof-read all code written to adhere to standard Java writing style.
  - Advised my team to apply knowledge learnt in lecture wherever necessary.

- Reviewed pull requests : #27, #31, #107
- Created the trailer for our project release.

- **Future contributions**
  - Implement sort command for medicine
  - Add tags to medicine, to allow flexible management of controlled drugs

Contributions to the User Guide

Given below are sections I contributed to the User Guide. They showcase my ability to write documentation targeting end-users.

Medicine Inventory Management

Adding a medicine: `addmedicine`

Adds a medicine to the CLInic medicine database.
Alias: `am`
Format: `addmedicine mn/MEDICINE_NAME msq/MINIMUM_STOCK_QUANTITY ppu/PRICE_PER_UNIT sn/SERIAL_NUMBER s/STOCK`

> **NOTE** The serial number of a medicine **must** be a 5 digit integer!

Example:

- `addmedicine mn/panadol msq/500 ppu/0.50 sn/91853 s/1000`
  Adds a medicine *named* `panadol` with *minimum stock quantity* of `500` units, *price per unit* of `$0.50`, *serial number* of `91853` and *stock* of `500` units to the CLInic inventory.
- `am mn/asprin msq/100 ppu/0.20 sn/53068 s/100`
  Adds a medicine *named* `asprin` with *minimum stock quantity* of `100` units, *price per unit* of `$0.20`, *serial number* of `53068` and *stock* of `100` units to the CLInic inventory.

Editing a medicine: `editmedicine`

Edits the details of an existing medicine.
Alias: `em`
Format: `editmedicine INDEX [mn/MEDICINE_NAME] [msq/MINIMUM_STOCK_QUANTITY] [ppu/PRICE_PER_UNIT] [sn/SERIAL_NUMBER] [s/STOCK]`

> **NOTE** The serial number of a medicine **must** be a 5 digit integer!

> **TIP** Edits the medicine details at the specified `INDEX`. The index refers to the index number shown in the displayed medicine list. The index **must be a positive integer** (i.e. 1, 2, 3, …).

TIP	At least one of the optional parameters must be provided.
TIP	Existing values will be updated with the newly input values of the corresponding field.
TIP	You can remove any of the medicine details by typing the prefixes `msq/ ppu/ sn/ s/` without specifying any contents after the prefix.

Example:

- `editmedicine 1 mn/hydrazine s/1500`
  Renames the medicine at index 1 to `hydrazine` whilst updating its stock to `1500`.
- `em 1 sn/91853`
  Updates the serial number of the medicine at index 1 to `91853`.

Restocking a medicine: `restock`

Restocks an existing medicine with **additional** quantity.
Alias: `rs`
Format: `restock INDEX amt/AMOUNT`

TIP	Restocks the medicine at the specified `INDEX`. The index refers to the index number shown in the displayed medicine list. The index **must be a positive integer** (i.e. 1, 2, 3, …).

Example:

- `restock 2 amt/123`
  Adds `123` additional units of the 2nd medicine to the clinic's current stock.
- `rs 3 amt/500`
  Adds `500` additional units of the 3rd medicine to the clinic's current stock.

Listing all medicines: `liststock`

Lists all medicine information in the CLInic medicine inventory.
Alias: `ls`
Format: `liststock`

Finding details of a medicine: `findmedicine`

Finds the details of a medicine from its medicine name.
Alias: `fm`
Format: `findmedicine MEDICINE_NAME`

Example:

- `findmedicine panadol`
  Finds the details of the medicines tagged with the keyword `panadol`.

- `fm chlorpheniramine`
  Finds the details of the medicines tagged with the keyword `chlorpheniramine`.

Listing medicines that are low in supply: `checkstock`

Lists all medicines that are low in supply.
Alias: `cs`
Format: `checkstock`

Contributions to the Developer Guide

> *Given below are sections I contributed to the Developer Guide. They showcase my ability to write technical documentation and the technical depth of my contributions to the project.*

Medicine Records System

Current Implementation

The medicine records system is used to manage the medicine inventory in the clinic. The `addMedicine` Command creates a new medicine entry in the inventory. `editMedicine` Command updates the stock level of the specified medicine. In addition, whenever the `receipt` Command is called, the records will update the stock automatically by subtracting the number of each medicine issued to the patient.

Given below is are examples of how the medicine class is used:

Scenario 1. A new medicine arrives. The user uses the `addMedicine` command to add the stock into the records.

Scenario 2. A new shipment of medicine has arrived. The user uses `restock` command to update the new stock levels of exiting medicine.

Scenario 3. When a patient is allocated medicine, the 'dispensemedicine' command is entered. During payment a `receipt` command is executed by the user to fetch the prices for the medicine that is allocated, in addition, the stock level of the medicine is updated.

Scenario 4. The user needs to check for the stock level of every medicine. The user executes the `checkStock` command which lists the medicine that are below their Minimum Stock Quantity.

Design Considerations

Aspect: Execution of the command

- **Alternative 1 (current choice):** Stores the medicine in a list.
  - Pros: Consistent format with the list with patients.
  - Cons: Need to search through the list to find a medicine.
- **Alternative 2:** Stores the medicine in a hash map.

- Pros: Quick reference for object given index.
- Cons: Difficult to sort the medicine.