T1A3 – Terminal Application

Stocktake app

Starting Menu

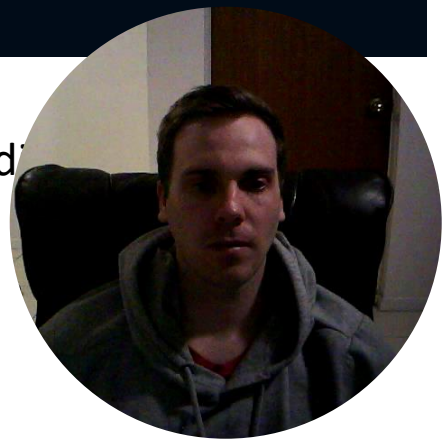
```
Welcome to the Stocktake App, Please choose what you would like to do from the list of options;

    1. Select stocktaking method and define range;
        • Location to Location (L)
        • Cycle Code (C)
    2. Generate Count Sheet (S)
    3. Input Counts (I)
    4. Generate Variance Report (R)
    5. Confirm and Commit Changes (F)
    6. Exit the Program (E)

Please enter the letter corresponding to your menu choice:
```

```python
menu_options = {
    "L": menufunctions.location_to_location,
    "C": menufunctions.cycle_code,
    "S": menufunctions.generate_count_sheet,
    "I": menufunctions.input_counts,
    "R": menufunctions.generate_variance_report,
    "F": menufunctions.confirm_and_commit_changes,
}
```

Full list of functions in a menu like structure.

```python
    while True:
        print(f"""
{open_message}

    1. Select stocktaking method and define range;
        \u2022 Location to Location (L)
        \u2022 Cycle Code (C)
    2. Generate Count Sheet (S)
    3. Input Counts (I)
    4. Generate Variance Report (R)
    5. Confirm and Commit Changes (F)
    6. Exit the Program (E)
    """)

        open_message = "\033[1;95mWhat would you like to do now?\033[0m"
        choice = input("\033[1mPlease enter the letter corresponding to your menu choice: \033[0m").upper()

        if choice == "E":
            break

        elif choice in menu_options:
            if choice == "L" or choice == "C":
                stocktake_selection = menu_options[choice]()
            elif choice == "S" or choice == "I":
                if stocktake_selection:
                    count = menu_options[choice](stocktake_selection)
                else:
```

A loop to allow user input and cycle through di
menu options



1. First menu option

Location to location

```
Please enter the letter corresponding to your menu choice: l

what is the start location: a1

what is the finish location: a2

This is the selection you have chosen:
+-------------+----------------------+------------+---------+---------------+-------------+
| stockcode   | description          | location   |   units |   costperunit | cyclecode   |
+=============+======================+============+=========+===============+=============+
| ULRR1       | ultra light red rope | A1         |       8 |            20 | A           |
+-------------+----------------------+------------+---------+---------------+-------------+
| SLRR1       | super light red rope | A2         |       8 |            15 | A           |
+-------------+----------------------+------------+---------+---------------+-------------+

Ok to continue with the selection? (Y/N):
```

Cycle code

```
Please enter the letter corresponding to your menu choice: c

What is the cycle code: b

This is the selection you have chosen:
+-------------+------------------------+------------+---------+---------------+-------------+
| stockcode   | description            | location   |   units |   costperunit | cyclecode   |
+=============+========================+============+=========+===============+=============+
| RLRR1       | regular light red rope | A3         |      14 |            10 | B           |
+-------------+------------------------+------------+---------+---------------+-------------+
| SLBB1       | super light blue ball  | B2         |       9 |            18 | B           |
+-------------+------------------------+------------+---------+---------------+-------------+
| RLBB1       | regular light blue ball | B3         |      11 |            12 | B           |
+-------------+------------------------+------------+---------+---------------+-------------+
| SLGG1       | super light green glove | C2         |       6 |            17 | B           |
+-------------+------------------------+------------+---------+---------------+-------------+

Ok to continue with the selection? (Y/N):
```

```python
def cycle_code():
    print()
    cycle_selection = input("What is the cycle code: ").upper()
    database = read_data_from_csv(database_file)
    stocktake_selection = [d for d in database if cycle_selection == d["cyclecode"]]
    if stocktake_selection:
        print()
        print(f"This is the selection you have chosen: \n{tabulate(stocktake_selection, headers='keys', tablefmt='grid')}")
        print()
        selection_ok = input("\033[1mOk to continue with the selection? (Y/N): \033[0m").upper()
        print()
        if selection_ok == "Y":
            return stocktake_selection
        elif selection_ok == "N":
            print("Selection removed. Returning to Menu...")
            time.sleep(1)
        else:
            print("Invalid Input. Returning to Menu...")
            time.sleep(1)
    else:
        print()
        print("No data range selected. Returning to Menu...")
        time.sleep(1)
```

- Reads database
- Applies filter to the database from the user selection
- Makes a check to ensure values are in filtered range
- Shows it to the user
- Asks for confirmation
- Time delays so the user has time to read

2. Count sheet

Command line feedback

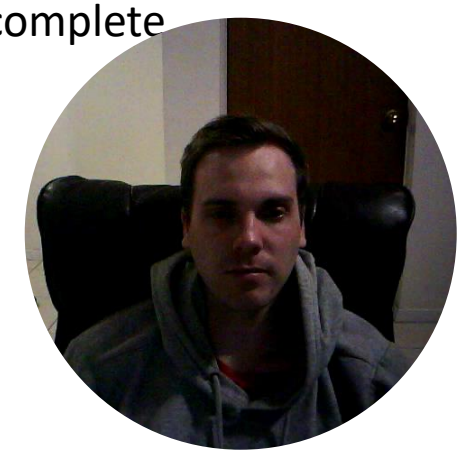
```
Please enter the letter corresponding to your menu choice: s

-----------------------------------------------------
The count sheet has been generated as count_sheet.txt
-----------------------------------------------------
```

Text file output

stockcode	description	location	count
RLRR1	regular light red rope	A3	
SLBB1	super light blue ball	B2	
RLBB1	regular light blue ball	B3	
SLGG1	super light green glove	C2	

```python
def generate_count_sheet(selection_data):
    data = copy.deepcopy(selection_data)
    for i in range(len(data)):
        data[i].pop("units")
        data[i].pop("costperunit")
        data[i].pop("cyclecode")
        data[i].update({"count" : " "})
    print(tabulate(data, headers="keys", tablefmt="grid"), file=open(count_sheet_file, "w"))
    print()
    print("-----------------------------------------------------")
    print(f"The count sheet has been generated as {count_sheet_file}")
    print("-----------------------------------------------------")
    time.sleep(1)
```

- Takes the selection data as input
- Creates a copy to prevent modifying
- Loops over each dictionary removes the elements not required
- Adds empty count field
- Saves the tabulated version of the new table to a text file
- Let’s the user know the process is complete



3. Input Counts

```python
def input_counts(selection_data):
    while True:
        count = copy.deepcopy(selection_data)
        for item in count:
            print()
            while True:
                try:
                    item["units"] = int(input(f"Please enter the count of item {item['stockcode']} : "))
                    break
                except ValueError:
                    print("Invalid input. Please enter an integer value.")
        subset_keys = ["stockcode", "units"]
        print()
        print(tabulate([{k: item[k] for k in subset_keys} for item in count], headers="keys", tablefmt="grid"))
        while True:
            print()
            choice = input("\033[1mAre you happy with the quantities shown on screen? (Y/N): \033[0m")
            if choice.upper() == "Y":
                break
            elif choice.upper() == "N":
                break
            else:
                print("\033[1mInvalid selection. Try pressing either Y for yes or N for no\033[0m")
        if choice.upper() == "Y":
            break
    return count
```

- Takes the selection data as input
- Makes a copy to prevent modifying
- Iterate over items
- Exception error for handling non-integers
- Creates a small table with only the stockcode and units.
- While loop kicks for confirmation kicks in after each input.
- Returns that count that is stored in a variable for access by other functions

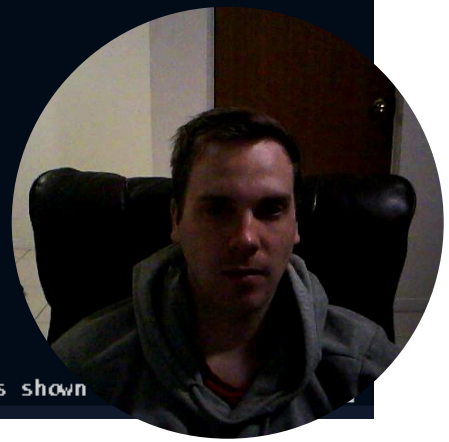
Iterates over items asking for user input for each

Shows a table of the inputs onscreen with a confirmation request

Loops back when answer = n

Forces user input to be of type integer.

```
Please enter the letter corresponding to your menu choice: i

Please enter the count of item RLRR1 : 5

Please enter the count of item SLBB1 : 5

Please enter the count of item RLBB1 : 5

Please enter the count of item SLGG1 : 4

+-------------+---------+
| stockcode   |   units |
+=============+=========+
| RLRR1       |       5 |
+-------------+---------+
| SLBB1       |       5 |
+-------------+---------+
| RLBB1       |       5 |
+-------------+---------+
| SLGG1       |       4 |
+-------------+---------+

Are you happy with the quantities shown on screen? (Y/N): n

Please enter the count of item RLRR1 : g
Invalid input. Please enter an integer value.
Please enter the count of item RLRR1 : 5

Please enter the count of item SLBB1 : 5

Please enter the count of item RLBB1 : 5

Please enter the count of item SLGG1 : 5

+-------------+---------+
| stockcode   |   units |
+=============+=========+
| RLRR1       |       5 |
+-------------+---------+
| SLBB1       |       5 |
+-------------+---------+
| RLBB1       |       5 |
+-------------+---------+
| SLGG1       |       5 |
+-------------+---------+

Are you happy with the quantities shown
```



4. Generate Variance Report

- Takes the selection data and counts as input
- Merges the two lists of dictionaries into one
- Makes calculations for the unit variance and related cost
- Gives the option to view the variance before committing to writing the report to file
- Looking for file directory and creating in the current path.
- Made with today's date in the file name
- Gives the user feedback once the file has been generated.

```python
def generate_variance_report(selection_data, count):
    variance_report = []
    for database, changes in zip(selection_data, count):
        stockcode = database["stockcode"]
        description = database["description"]
        units1 = database["units"]
        units2 = changes["units"]
        costperunit = database["costperunit"]
        variance = int(units2) - int(units1)
        totalcost = int(variance) * int(costperunit)
        variance_report.append({"stockcode": stockcode, "description": description, "units_in_database":
        print()
    show_report = input("\033[1mWould you like to view the report onscreen? (Y/N): \033[0m").upper()
    if show_report == "Y":
        print()
        print(tabulate(variance_report, headers="keys", tablefmt="grid"))
        proceed_next_step = input("Continue? (Y/N): ").upper()
        if proceed_next_step == "N":
            print()
            print("Report generation cancelled, returning to menu...")
            time.sleep(1)
            return
        elif proceed_next_step == "Y":
            pass
        else:
            print()
            print("Invalid input. Report generation cancelled, returning to menu...")
            time.sleep(1)
            return
    elif show_report == "N":
        pass
    else:
        print()
        print("Invalid selection. Report generation cancelled, returing to menu...")
        return
    current_date = datetime.now().strftime("%d-%b-%Y")
    folder_name = variance_report_folder
    file_name = f"variances_{current_date}.txt"
    script_dir = os.path.dirname(os.path.abspath(__file__))
    folder_path = os.path.join(script_dir, folder_name)
    file_path = os.path.join(folder_path, file_name)
    if not os.path.exists(folder_path):
        os.makedirs(folder_path)
    print()
    print(tabulate(variance_report, headers="keys", tablefmt="grid"),
    print("-----------------------------------------------------------
    print(f"The variance report has been generated as {file_name}")
    print("-----------------------------------------------------------
    time.sleep(1)
    return variance_report
```

5. Confirm and commit changes

- Asks for confirmation from the user.
- Makes a copy of the counts
- Opens the database
- Makes the changes to each group of data based on the counts
- Writes the changes over the top of the current database
- Some error handling at the bottom for cases of rejection to the warning message

Terminal

```
Warning! Are you sure you wish to proceed in committing changes to the database? (Y/N): y

-----------------------------------------------------------------
Changes have been commited to the database, returning to menu...
-----------------------------------------------------------------
```

Code

```
def confirm_and_commit_changes(count):
    print()
    confirmation = input("\033[1;31mWarning!\033[0m Are you sure you wish to proceed
    if confirmation.upper() == "Y":
        changes = copy.deepcopy(count)
        confirmation_file = []
        database = read_data_from_csv(database_file)
        for item_in_changes in changes:
            item_in_changes_id = item_in_changes["stockcode"]
            for item_in_database in database:
                if item_in_database["stockcode"] == item_in_changes_id:
                    item_in_database.update(item_in_changes)
                    break
        fieldnames = database[0].keys()
        write_data_to_csv(database_file, database, fieldnames)
        print()
        print("-----------------------------------------------------
        print("Changes have been commited to the database, retu
        print("-----------------------------------------------------
        time.sleep(1)
    elif confirmation.upper() == "N":
        print()
        print("No changes have been made, returning to menu
        time.sleep(1)
    else:
        print()
        print("Invalid Input. No changes have been made, retur
        time.sleep(1)
```

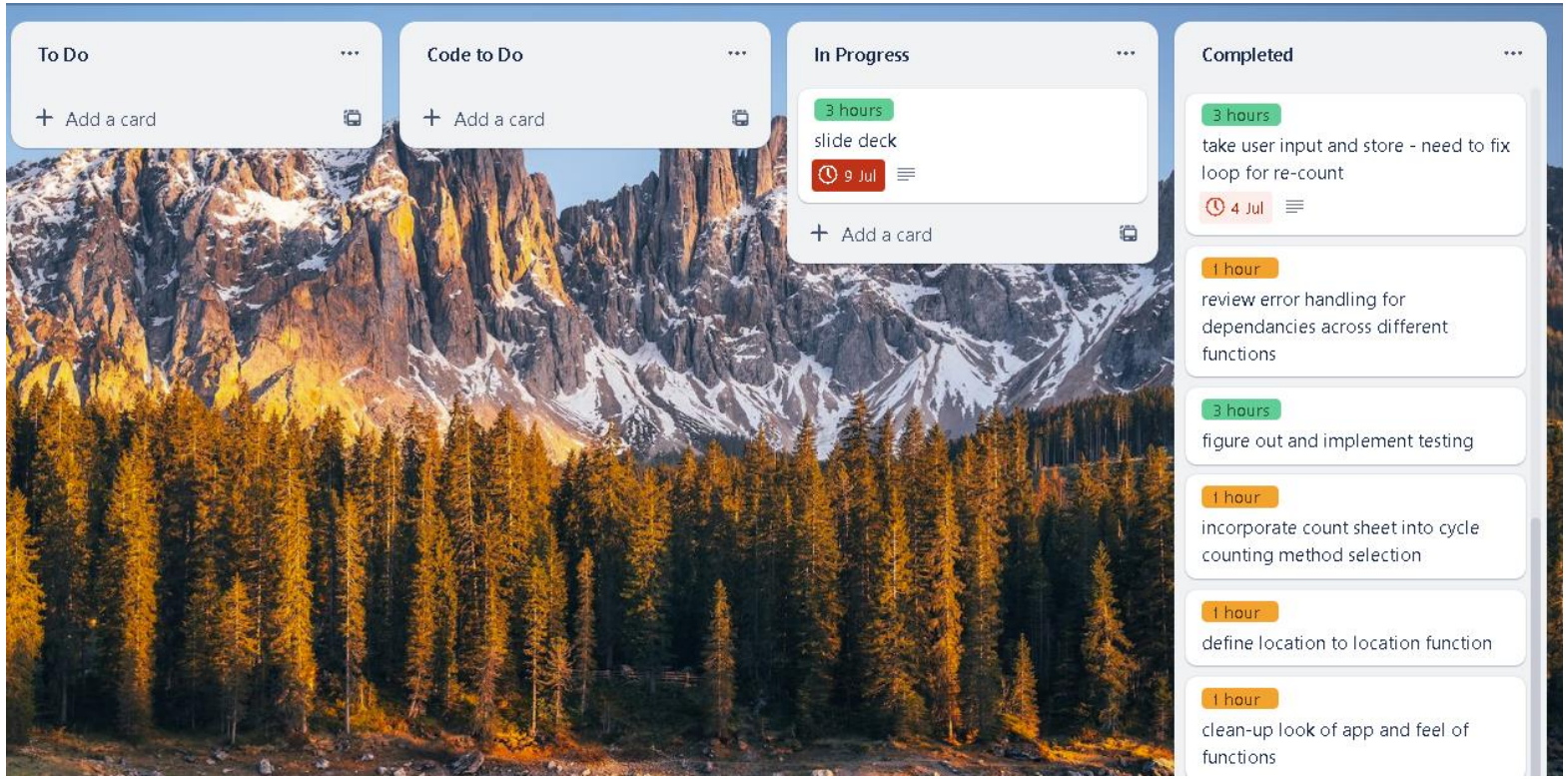
CSV subfunctions

```python
import csv

def read_data_from_csv(file_path):
    with open(file_path) as file:
        reader = csv.DictReader(file)
        data = list(reader)
    return data

def write_data_to_csv(file_path, data, fieldnames):
    with open(file_path, "w", newline="") as file:
        writer = csv.DictWriter(file, fieldnames=fieldnames)
        writer.writeheader()
        writer.writerows(data)
```

* In own file to be called upon when needed

Build Process

- Kanban on Trello.com



- Broke down each stage into steps and how long I expected to take on them

challenges, favourite parts, ethical issues

Challenges

- Time management
- Testing modules

Favourite parts

- If statements
- While loops
- Fixing the errors in my own code
- File manipulation

Ethical question – ChatGPT seems overpowered

End