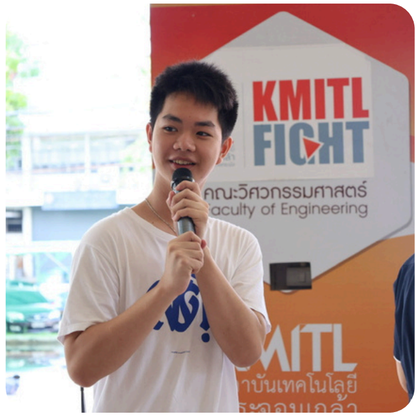
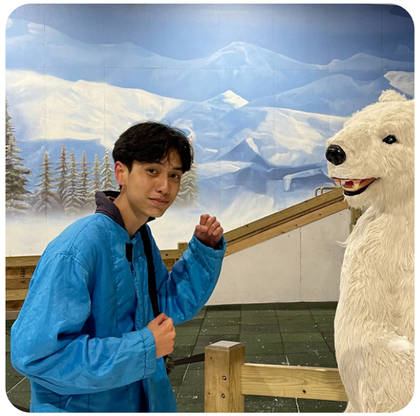
MINI PROJECT

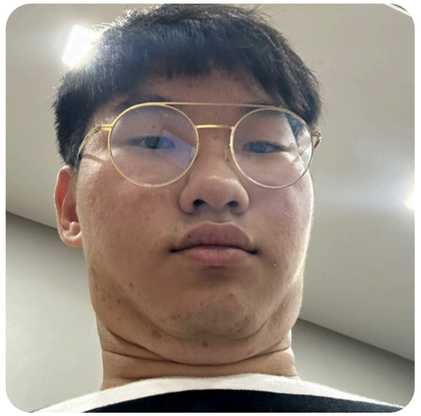
Our Team



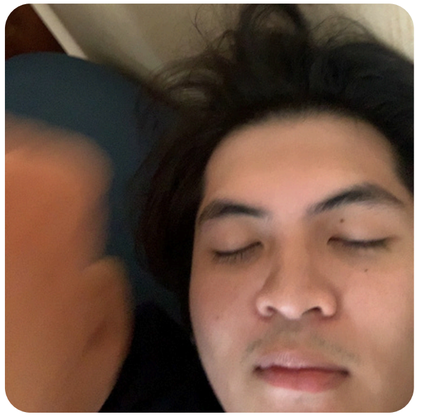
Akesit Akkharasaksiri
65110131



Rattapol Kitirak
65110149



Pann Punyajaray
65110143



Theedhat Chankrut
65110155

INTRODCTION

The reason why you chose this process

We aim to leverage our understanding of algorithms to streamline our company's inventory management system, making it easier to monitor tool borrowings and item withdrawals. We believe this approach can have practical applications in various industries, offering significant real-world benefits.

CODE

```
import json
import os
from datetime import datetime

class StockManager:
    def __init__(self):
        self.stock = {}
        self.history = {}
        self.users = {}

    def group_stock(self):
        grouped_stock = {}
        for item, quantity in self.stock.items():
            item_type = item.split('-')[0]
            if item_type not in grouped_stock:
                grouped_stock[item_type] = {}
            grouped_stock[item_type][item] = quantity
        return grouped_stock

    def search_stock(self, item_name):
        if item_name in self.stock:
            return {item_name: self.stock[item_name]}
        else:
            return None
```

```
    def insert_stock(self, item_name, quantity, user):
        if item_name in self.stock:
            self.stock[item_name] += quantity
        else:
            self.stock[item_name] = quantity
        timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        action = f"{user} inserted {quantity} {item_name}"
        if item_name not in self.history:
            self.history[item_name] = []
        self.history[item_name].append((timestamp, action))
    def delete_stock(self, item_name, quantity=1, user=None):
        if item_name in self.stock:
            if self.stock[item_name] >= quantity:
                self.stock[item_name] -= quantity
                if self.stock[item_name] == 0:
                    del self.stock[item_name]
                timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
                if user:
                    action = f"{user} deleted {quantity} {item_name}"
                else:
                    action = f"Deleted {quantity} {item_name}"
                if item_name not in self.history:
                    self.history[item_name] = []
                self.history[item_name].append((timestamp, action))
```

CODE

```
i else:
            print(f"Not enough {item_name} in stock to remove {quantity}.")
        else:
            print(f"{item_name} not found in stock.")

    def save_to_file(self, filename):
        with open(filename, 'w') as file:
            data_to_save = {
                "stock": self.stock,
                "history": self.history,
                "users": self.users
            }
            json.dump(data_to_save, file)
        print(f"Stock data, history, and user data saved to {filename}.")

    def load_from_file(self, filename):
        if os.path.exists(filename):
            try:
                with open(filename, 'r') as file:
                    data = json.load(file)
                    self.stock = data.get("stock", {})
                    self.history = data.get("history", {})
                    self.users = data.get("users", {})
                print(f"Stock data, history, and user data loaded from {filename}.")
except Exception as e:
            print(f"Error loading data from {filename}: {e}")
        else:
            print(f"{filename} not found. Starting with an empty stock, history, and user
data.")

    def register_user(self, username, password):
        self.users[username] = password

    def login(self, username, password):
        return self.users.get(username) == password

    def save_users_to_file(self, filename):
        with open(filename, 'w') as file:
            json.dump(self.users, file)
        print(f"User data saved to {filename}.")
```

CODE

```
def load_users_from_file(self, filename):
        if os.path.exists(filename):
            try:
                with open(filename, 'r') as file:
                    self.users = json.load(file)
                print(f"User data loaded from {filename}.")
            except Exception as e:
                print(f"Error loading user data from {filename}: {e}")
        else:
            print(f"{filename} not found. Starting with an empty user database.")

    def view_history(self):
        if not self.history:
            print("\nHistory is empty.")
        else:
            print("\nStock History:")
            for item_name, actions in self.history.items():
                print(f"{item_name}:")
                for timestamp, action in actions:
                    print(f"  {timestamp}: {action}")
```

```
def main():
    manager = StockManager()
    file_path = ""
    script_directory = os.path.dirname(__file__)
    user_file = os.path.join(script_directory, "users.json")
    stock_data_file = os.path.join(script_directory, "stock_data.json")
    if not os.path.exists(user_file):
        with open(user_file, 'w') as file:
            json.dump({}, file)
    if not os.path.exists(stock_data_file):
        with open(stock_data_file, 'w') as file:
            json.dump({"stock": {}, "history": {}, "users": {}}, file)
    manager.load_from_file(stock_data_file)
    manager.load_users_from_file(user_file)

    while True:
        print("\nOptions:")
        print("1. Register")
        print("2. Login")
        print("3. Exit")
```

CODE

```
choice = input("Enter your choice: ")

    if choice == "1":
        username = input("Enter username: ")
        password = input("Enter password: ")
        manager.register_user(username, password)
        manager.save_users_to_file(user_file)
        print(f"User {username} registered successfully.")

    elif choice == "2":
        username = input("Enter username: ")
        password = input("Enter password: ")
        if manager.login(username, password):
            print(f"Welcome, {username}!")

            while True:
                print("\nStock Options:")
                print("1. Add stock")
                print("2. Group stock")
                print("3. Search stock")
                print("4. Delete stock")
                print("5. Stock report")
                print("6. View History")
                print("7. Logout")
```

```
stock_choice = input("Enter your choice: ")

                if stock_choice == "1":
                    item_name = input("Enter item name: ")
                    quantity = int(input("Enter quantity: "))
                    user = username
                    manager.insert_stock(item_name, quantity, user)
                    print(f"{quantity} {item_name} added to stock.")
                elif stock_choice == "2":
                    grouped_stock = manager.group_stock()
                    print("\nGrouped Stock:")
                    for item_type, items in grouped_stock.items():
                        print(f"{item_type}: {items}")

                elif stock_choice == "3":
                    search_item = input("Enter item name to search: ")
                    found_item = manager.search_stock(search_item)
                    if found_item:
                        print(f"{search_item} found in stock: {found_item}")
                    else:
                        print(f"{search_item} not found in stock.")
```

CODE

```
elif stock_choice == "4":
                delete_item = input("Enter item name to delete: ")
                delete_quantity = int(input("Enter quantity to delete: "))
                user = username
                manager.delete_stock(delete_item, delete_quantity, user)

            elif stock_choice == "5":
                total_quantity = sum(manager.stock.values())
                print("\nStock Report:")
                print(f"Total Items: {len(manager.stock)}")
                print(f"Total Quantity: {total_quantity}")

            elif stock_choice == "6":
                manager.view_history()

            elif stock_choice == "7":
                print("Logging out.")
                break

            else:
                print("Invalid choice. Please try again.")

        else:
            print("Invalid username or password.")
```

```
    elif choice == "3":
            manager.save_to_file(stock_data_file)
            break

        else:
            print("Invalid choice. Please try again.")

if __name__ == "__main__":
    main()
```

REASON WHY WE APPLIED THIS ALGORITHMS?

Algorithms can provide a clear and automated record of who borrowed tools or took items out of the company. The tracking process can be streamlined by algorithms, which makes it simpler to keep track of and control the movement of tools and other items. The time and effort needed for manual record-keeping are decreased as a result.