PasoEats Design Explanation

Encapsulation:
- All fields are private. Variables are only accessible through getter and setter methods.

Inheritance:
- Customer, Driver, and Admin extend User to reuse its class attributes (`UUID id, String username, String name`).

Polymorphism:
- For our command line interface we will use different fields from the User class for each user type.

Abstraction:
- User class is abstract and has abstract methods for each user type. This design enforces a common interface for all user types while abstracting away the unique implementation details of their specific actions.
- The OrderManager class exposes place(), acceptNext(), markStatus() and get(), to the user while hiding it's complex internal data structures such as Map and ArrayDeque. This approach ensures users interact only with necessary functionality, insulating them from internal changes to the storage implementation.

Data Structures:
- `byId: Map<UUID, Order>` - to store orders by their ID and have O(1) lookup time.
- `intake: ArrayDeque<UUID>` - to store orders in the order they were placed and have O(1) insertion and removal time. FIFO (First In, First Out).
- `List<OrderItem>` (inside Order) - to store the items in the order.
- `UUID` (User, Order, Driver, Customer, Admin) - to identify each object uniquely and control access to them in a secure way.

Overall Complexity:

Time Complexity: All Methods have a time complexity of O(1) except for .contains in the ArrayDeque. Similarily the HashMap will take O(n) time for the following functions: .put .remove .get .containsKey.

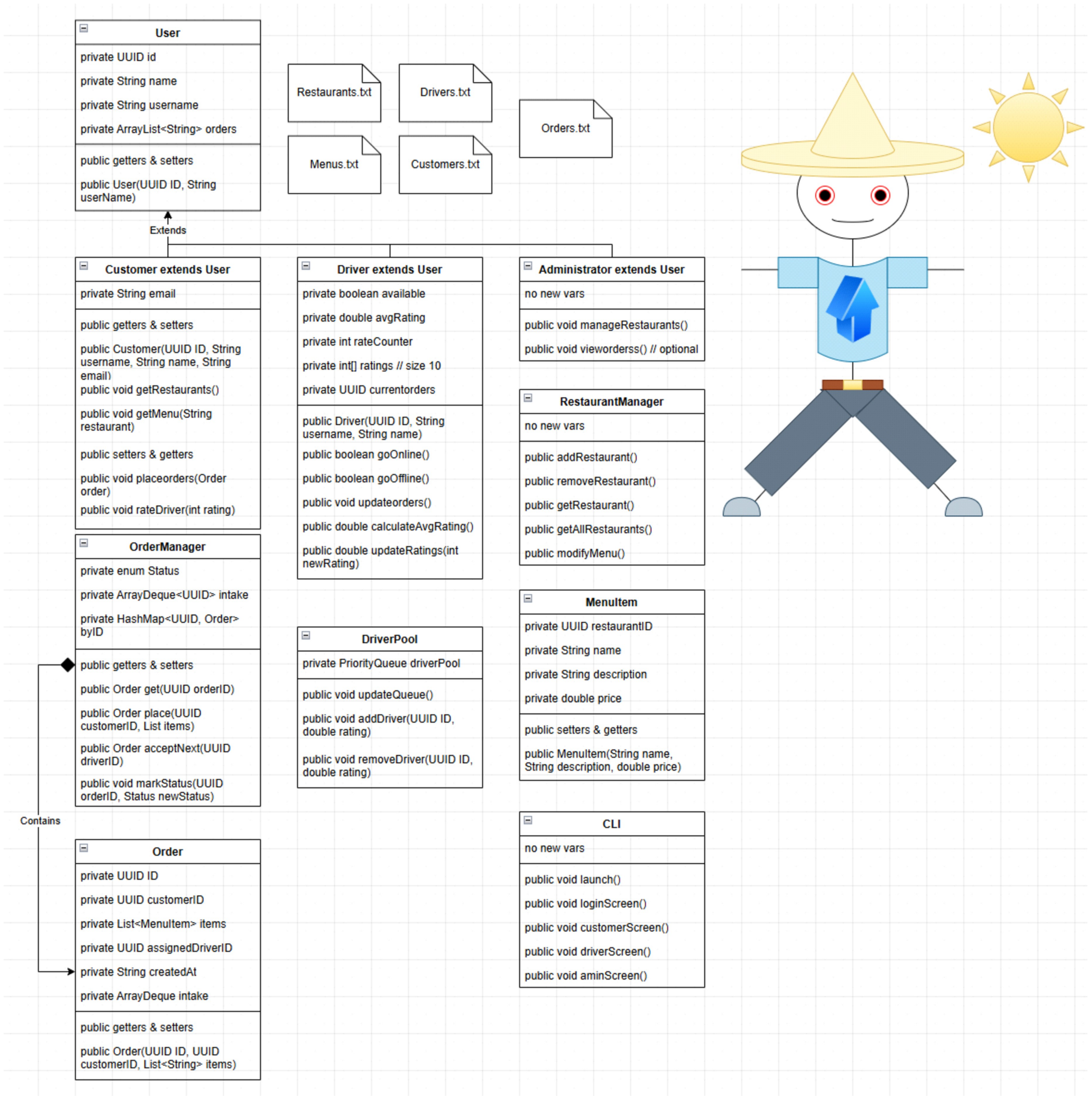
Space Complexity: O(n) for the map and queue.

Work Distribution:

- Reese
  - OrderManager
  - Order
- Connor
  - User

    - Driver
    - Customer
    - Adminstrator
- Brian
    - DriverPool
    - RestaurantManager
    - MenuItem
    - (potentially) Menu
    - *(optional, stretch goal) Graphical User Interface*
- Everybody
    - Command Line Interface

Class Diagram


User
private UUID id
private String name
private String username
private ArrayList<String> orders
public getters & setters
public User(UUID ID, String userName)
Restaurants.txt
Drivers.txt
Orders.txt
Menus.txt
Customers.txt
Extends
Customer extends User
private String email
public getters & setters
public Customer(UUID ID, String username, String name, String email)
public void getRestaurants()
public void getMenu(String restaurant)
public setters & getters
public void placeorders(Order order)
public void rateDriver(int rating)
Driver extends User
private boolean available
private double avgRating
private int rateCounter
private int[] ratings // size 10
private UUID currentorders
public Driver(UUID ID, String username, String name)
public boolean goOnline()
public boolean goOffline()
public void updateorders()
public double calculateAvgRating()
public double updateRatings(int newRating)
Administrator extends User
no new vars
public void manageRestaurants()
public void vieworderss() // optional
RestaurantManager
no new vars
public addRestaurant()
public removeRestaurant()
public getRestaurant()
public getAllRestaurants()
public modifyMenu()
OrderManager
private enum Status
private ArrayDeque<UUID> intake
private HashMap<UUID, Order> byID
public getters & setters
public Order get(UUID orderID)
public Order place(UUID customerID, List items)
public Order acceptNext(UUID driverID)
public void markStatus(UUID orderID, Status newStatus)
DriverPool
private PriorityQueue driverPool
public void updateQueue()
public void addDriver(UUID ID, double rating)
public void removeDriver(UUID ID, double rating)
MenuItem
private UUID restaurantID
private String name
private String description
private double price
public setters & getters
public MenuItem(String name, String description, double price)
Contains
Order
private UUID ID
private UUID customerID
private List<MenuItem> items
private UUID assignedDriverID
private String createdAt
private ArrayDeque intake
public getters & setters
public Order(UUID ID, UUID customerID, List<String> items)
CLI
no new vars
public void launch()
public void loginScreen()
public void customerScreen()
public void driverScreen()
public void aminScreen()