

DELI-CIOUS - JAVA CLI ORDERING APP

•••

BUILD YOUR SANDWICH, BYTE BY BYTE

Custom Orders, Programmed Perfectly!

WELCOME TO DELI-CIOUS



YOUR CLI-POWERED SANDWICH SHOP

This app was built in Java as part of my Capstone 2 project. It's a menu-driven command-line system that lets you build custom sandwich orders with drinks, chips, and receipts!

LET'S BUILD A SANDWICH

The image shows a smartphone displaying a Java development environment (IDE) and a terminal window. The IDE shows a project structure for 'DELI-cious' with files like Application.java, MenuOptions, Order, ReceiptWriter, Sandwich, and UserInterface. The terminal window shows the application's command-line interface:

```
C:\Users\Student\.jdks\corretto-17.0.14\bin>
=====
        🥪 WELCOME TO DELI-cious 🥪
        Build Your Sandwich, Byte by Byte!
=====
What would you like to do?
1) 📲 Start a New Order
0) ❌ Exit

Enter your choice:
```

PRES ENTATION ROADMAP

• • •

01.

OPENING

Quick overview of what DELI-cious is and what problem it solves.

02.

HOW IT WORKS (OOP + FEATURES)

Breakdown of the app structure and design using Java classes.

03.

LIVE DEMO

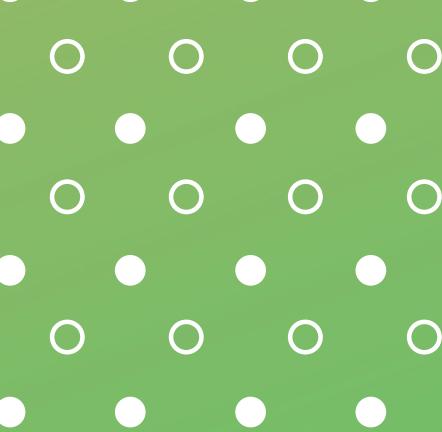
Walkthrough of a full order sandwich, drink, chips, and receipt generation.

04.

CODE SPOTLIGHT + REFLECTION

Highlight of our favorite method

WHAT WE'LL COVER TODAY



HOW DELI-CIOUS WAS BUILT (OOP STYLE)

• • •



[SANDWICH] CLASS

Stores sandwich options like bread, size, toppings, and toast. Calculates total price based on selections.



[UI] CLASS

Handles user prompts and menus. Connects the user to the app's logic flow.

The image shows two smartphones side-by-side. The left phone displays a Java code editor with the code for the [SANDWICH] CLASS and [UI] CLASS. The right phone displays a terminal window showing the output of the program, which includes the sandwich details and a total price of \$11.55.

```
package com.example.deli;

import java.util.ArrayList;
import java.util.List;

public class Sandwich {
    private String breadType; //they can choose from white or wheat
    private int size; //Customers can choose from small, medium, or large
    private boolean isToasted; //2 usages
    private List<String> meats; //4 usages
    private List<String> cheeses; //4 usages
    private List<String> regularToppings; //3 usages
    private List<String> sauces; //3 usages
    private double totalPrice; //9 usages

    public Sandwich(String breadType, int size, boolean isToasted, List<String> meats, List<String> cheeses, List<String> regularToppings, List<String> sauces) {
        this.breadType = breadType;
        this.size = size;
        this.isToasted = isToasted;
        this.meats = new ArrayList<>();
        this.cheeses = new ArrayList<>();
        this.regularToppings = new ArrayList<>();
        this.sauces = new ArrayList<>();
    }

    public void addMeat(String meat) {
        meats.add(meat);
    }

    public void addCheese(String cheese) {
        cheeses.add(cheese);
    }

    public void addRegularTopping(String topping) {
        regularToppings.add(topping);
    }

    public void addSauce(String sauce) {
        sauces.add(sauce);
    }

    public void calculateTotalPrice() {
        totalPrice = size * 5.0 + (isToasted ? 1.0 : 0.0) + meats.size() * 2.0 + cheeses.size() * 1.5 + regularToppings.size() * 0.5 + sauces.size() * 1.0;
    }

    public String toString() {
        return "Order Details:\n" +
                "4\" White (Toasted)\n" +
                "Meats: [Steak]\n" +
                "Cheeses: [American]\n" +
                "Toppings: [Lettuce, Tomato, Onion, Lettuce]\n" +
                "Sauces: [Ranch]\n" +
                "Price: $7.55\n" +
                "\n" +
                "Drinks: [medium Lemonade]\n" +
                "Chips: [Classic]\n" +
                "Total Price: $11.55";
    }
}
```



[RECEIPT WRITER]

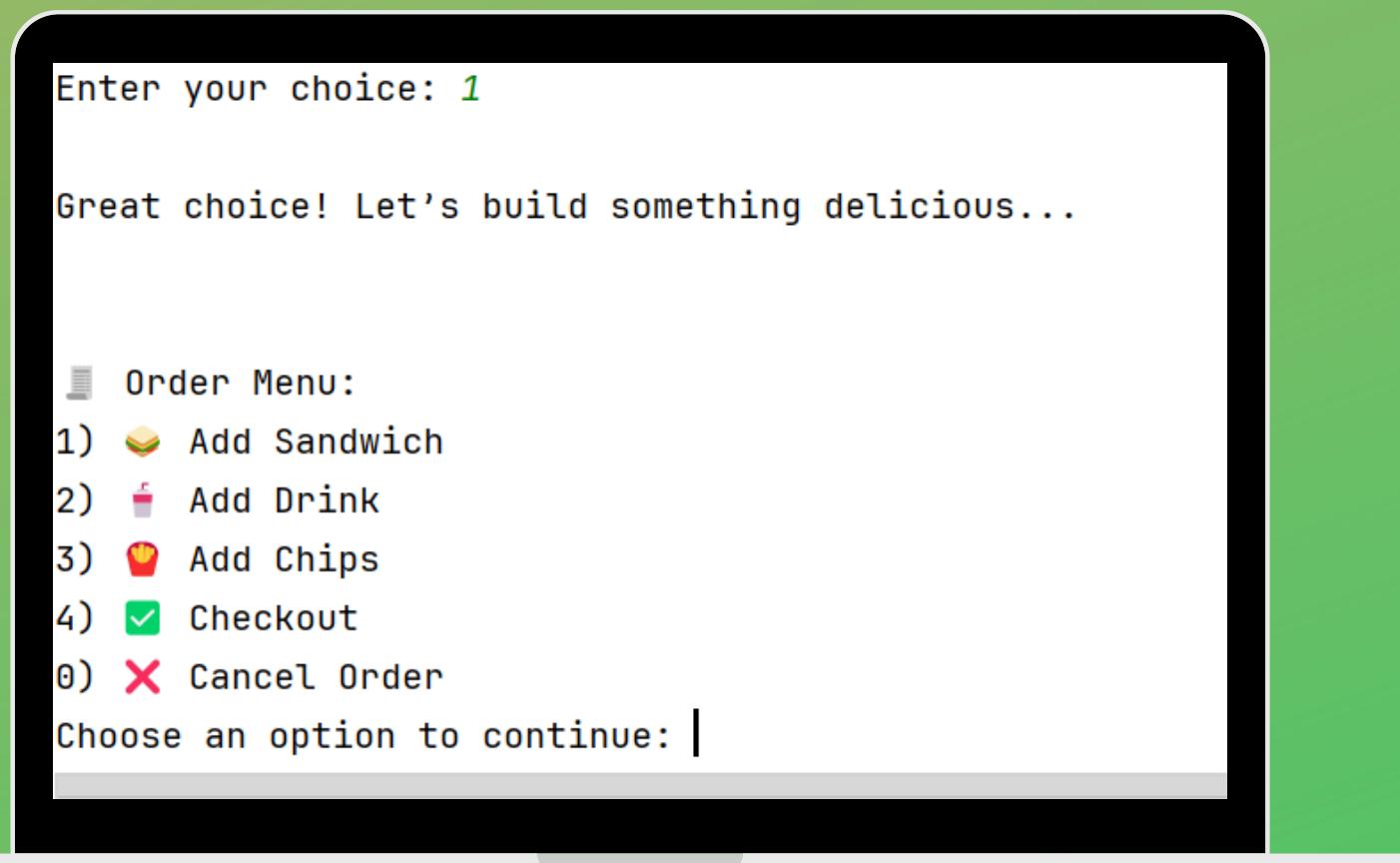
Saves the final order to a timestamped .txt file. Focuses only on receipt writing.



[ORDER] CLASS

Tracks everything in the order sandwiches, drinks, chips. Adds up prices and builds a full summary

HOW TO USE THE DELI-CIOUS APP



- **STEP ONE**

Open the terminal or command line in IntelliJ using Ctrl + Shift + F10 to run the program.

- **STEP TWO**

Choose 1 to start a new order or 0 to exit from the main menu.

- **STEP THREE**

Select sandwich size, bread, and whether it's toasted. Choose your meats, cheeses, toppings, and sauces. Also option to make it a combo by adding drinks and chips to your order!

- **STEP FOUR**

Review your full order summary and confirm. You'll see a detailed receipt printed to the screen.

- **STEP FIVE**

The receipt is automatically saved in the receipts/ folder, named with the date and time. Check your folder to see it saved as .txt.

MAKE IT A COMBO – A SMARTER WAY TO ORDER

AUTOMATIC COMBO BUILDER

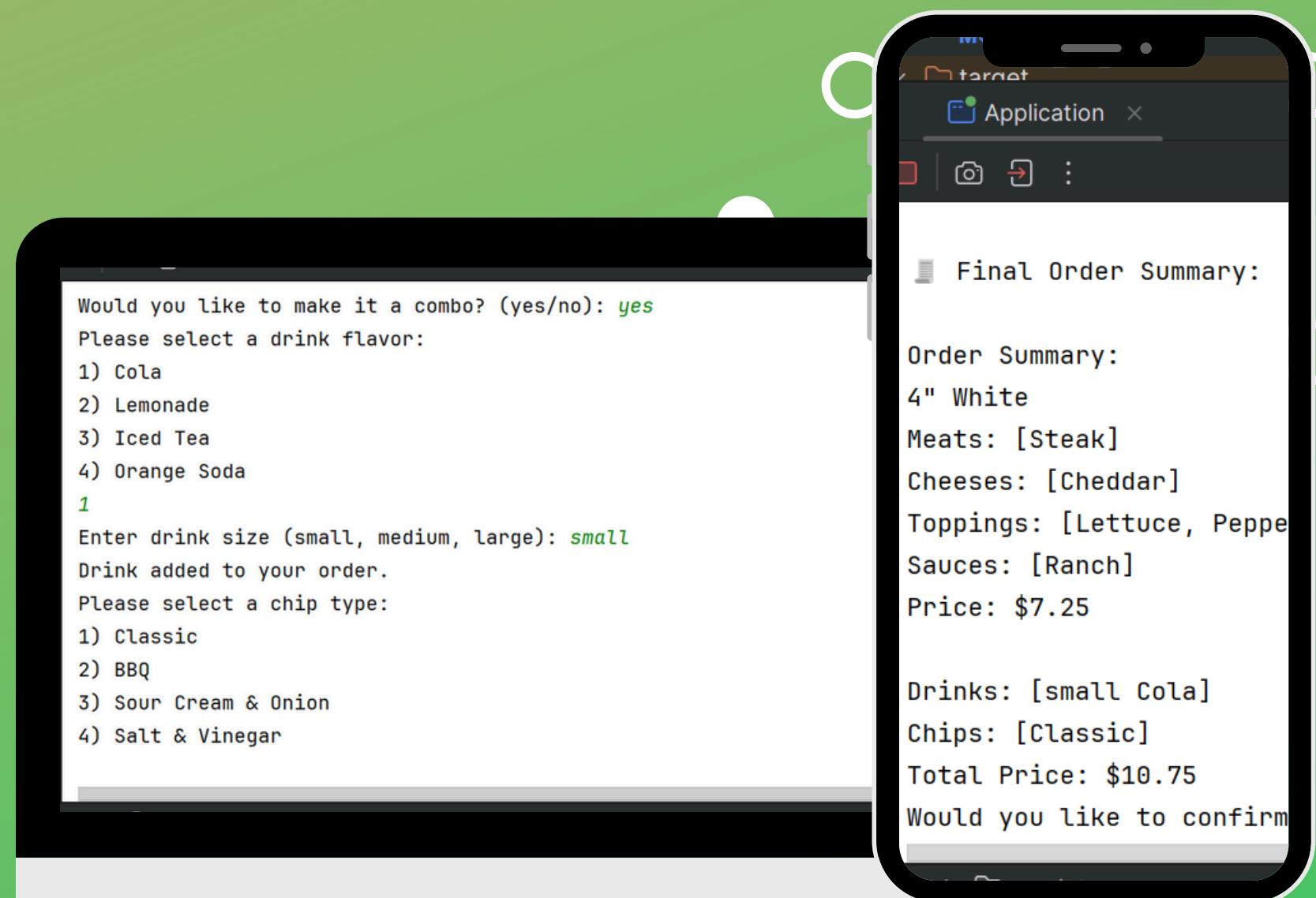
After a sandwich is built, users are asked if they want to make it a combo.

SMART PROMPTS

If yes, they select their drink size/flavor and chip type, all added instantly.

STREAMLINED EXPERIENCE

Reduces steps, adds value, and makes ordering faster and more complete.



THANKS FOR WATCHING



DELI-CIOUS JAVA APP

A CLI-powered sandwich builder designed with Object-Oriented Programming principles.

Custom orders, real-time pricing, and receipt generation all from the terminal!

Build Your Sandwich, Byte by Byte.

