

README

ieats App

INDEX

Project Description

page 3

Key Features
Technologies Used

Installation

page 4 - 5

Installation
Requirements

Usage

page 6 - 8

Login or Sign In
Main Menu
Worker Dashboard

Further Improvements

page 9

Credits

page 10

PROJECT DESCRIPTION

The ieats App was created to solve the problem of long lines and limited time during breaks at IE University. This app allows students to pre-order food and drinks, ensuring they can enjoy their breaks without the hassle of waiting in line.

Key Features

- **Pre-Order System:** Place orders for food and beverages ahead of time.
- **Flexible Payment Options:** Accepts cash on pickup, PayPal, Apple Pay, and other payment options.
- **Optimal Pickup Locations:** Orders can be picked up from convenient locations within the IE Tower.
- **Real-Time Order Management:** Ensures orders are completed within a 30-minute window.
- **User-Friendly Interface:** Simple and intuitive design for easy navigation.

Technologies Used

- **Python:** Core programming language for backend logic.
- **Dictionary Structures:** Manages products, shopping carts, and past orders.
- **Datetime Module:** Handles timestamps for orders and past orders.
- **Input/Output Handling:** Implements interactive features for user login, browsing, and checkout.
- **Development and Testing:** Created using Windows and tested on both macOS and Windows.

Challenges and Learnings

Challenges

- Managing product categories and dynamically adding selected items to the cart.
- Handling user input validation to ensure a seamless experience.
- Simulating a real-world ordering system with Python dictionaries and logic.

Future Enhancements

- Develop a front-end interface for better usability (e.g., React or Flutter).
- Integrate with a database to persist user data and past orders.
- Add order notifications and tracking features.
- Implement a recommendation system for frequently ordered items.

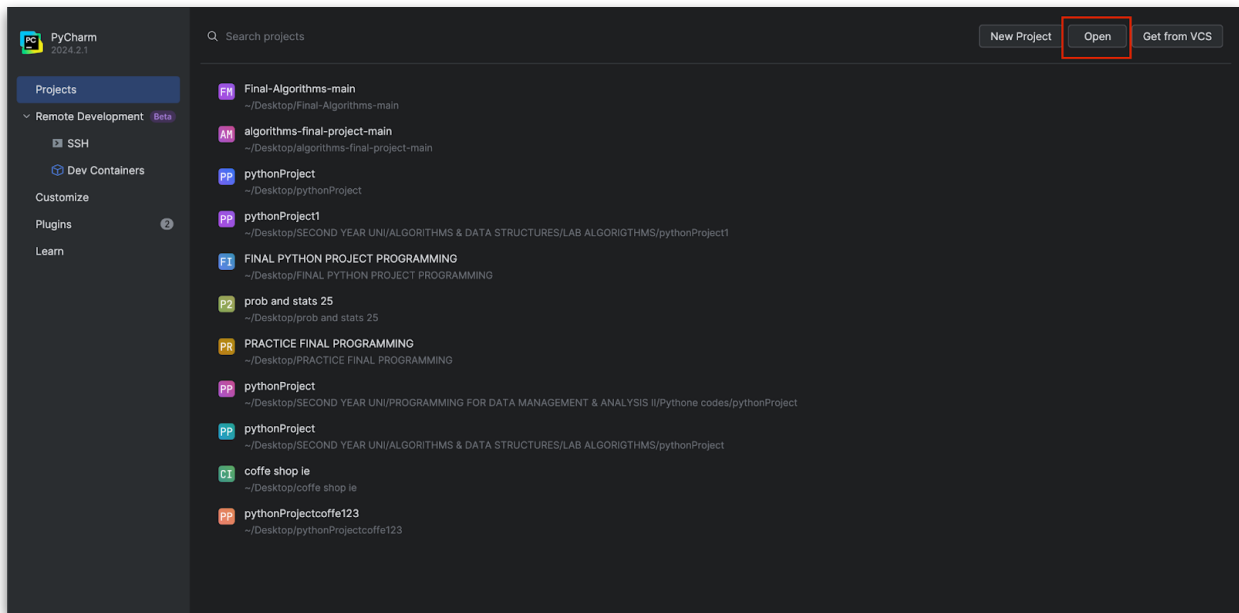
Why This Solution?

This app was created to optimise the brief and precious time students have between classes. By enabling pre-orders and offering a streamlined pickup process, the Uni Coffee Order Project enhances the overall university experience. It also serves as a practical use case for applying Python programming skills to solve real-world problems.

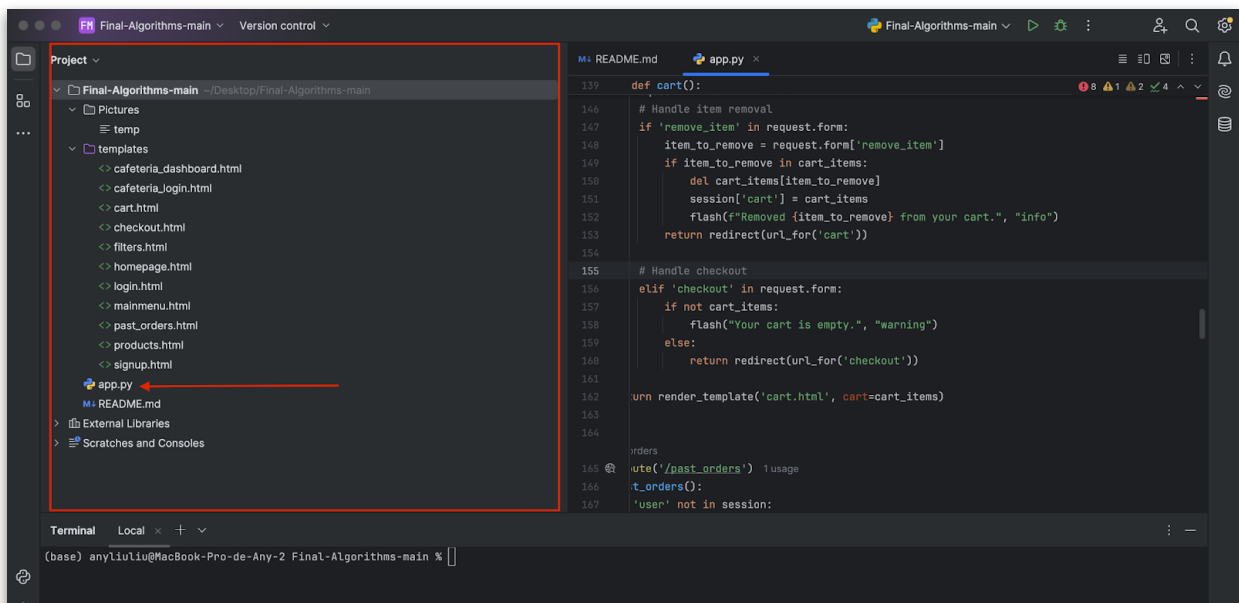
INSTALLATION

Installation

1. **Download the Zip File:** Save the folder to a known location on your computer.
2. **Recommended IDEs:** Use PyCharm or Visual Studio Code.
3. **Open the Folder:** Open the downloaded folder in your chosen IDE.

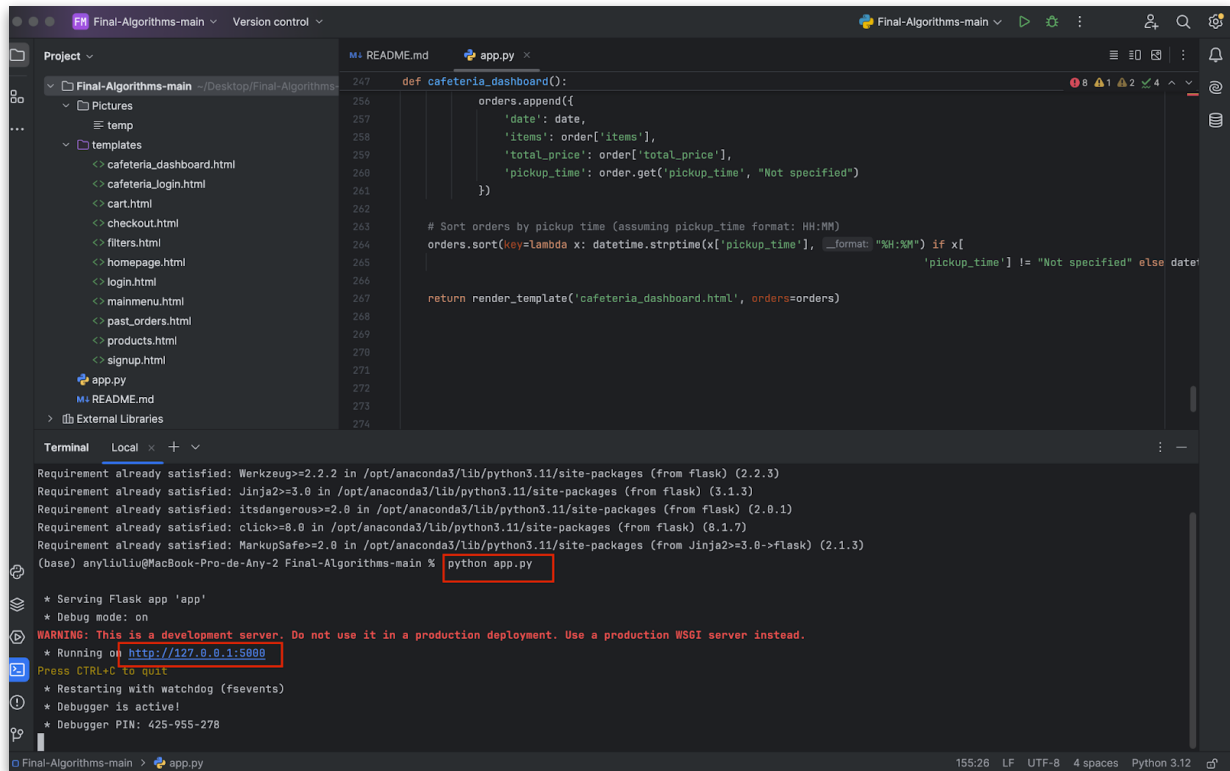


4. **View Project Structure:** On the left side of the screen, you will see all the elements inside the zip file.



5. Run the Application:

- Open app.py.
- In the terminal, install Flask if you don't have it: `pip install flask`
- Run the application: `python app.py`
- Click on the provided HTTP link to start using the program.



The screenshot shows a code editor with a project named 'Final-Algorithms-main'. The file explorer on the left shows a directory structure with 'templates' and 'app.py'. The main editor window displays the code for the `cafeteria_dashboard()` function in `app.py`. The terminal window at the bottom shows the command `python app.py` being executed, and the output indicates that the application is running on `http://127.0.0.1:5000`. The terminal also shows a warning about using a development server and a debugger PIN.

```
def cafeteria_dashboard():
    orders.append({
        'date': date,
        'items': order['items'],
        'total_price': order['total_price'],
        'pickup_time': order.get('pickup_time', "Not specified")
    })

    # Sort orders by pickup time (assuming pickup_time format: HH:MM)
    orders.sort(key=lambda x: datetime.strptime(x['pickup_time'], '%H:%M') if x['pickup_time'] != "Not specified" else date)

    return render_template('cafeteria_dashboard.html', orders=orders)
```

```
Requirement already satisfied: Werkzeug>=2.2.2 in /opt/anaconda3/lib/python3.11/site-packages (from flask) (2.2.3)
Requirement already satisfied: Jinja2>=3.0 in /opt/anaconda3/lib/python3.11/site-packages (from flask) (3.1.3)
Requirement already satisfied: itsdangerous>=2.0 in /opt/anaconda3/lib/python3.11/site-packages (from flask) (2.0.1)
Requirement already satisfied: click>=8.0 in /opt/anaconda3/lib/python3.11/site-packages (from flask) (8.1.7)
Requirement already satisfied: MarkupSafe>=2.0 in /opt/anaconda3/lib/python3.11/site-packages (from Jinja2>=3.0->flask) (2.1.3)
(base) anyliuliu@MacBook-Pro-de-Amy-2 Final-Algorithms-main % python app.py

* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with watchdog (fsevents)
* Debugger is active!
* Debugger PIN: 425-955-278
```

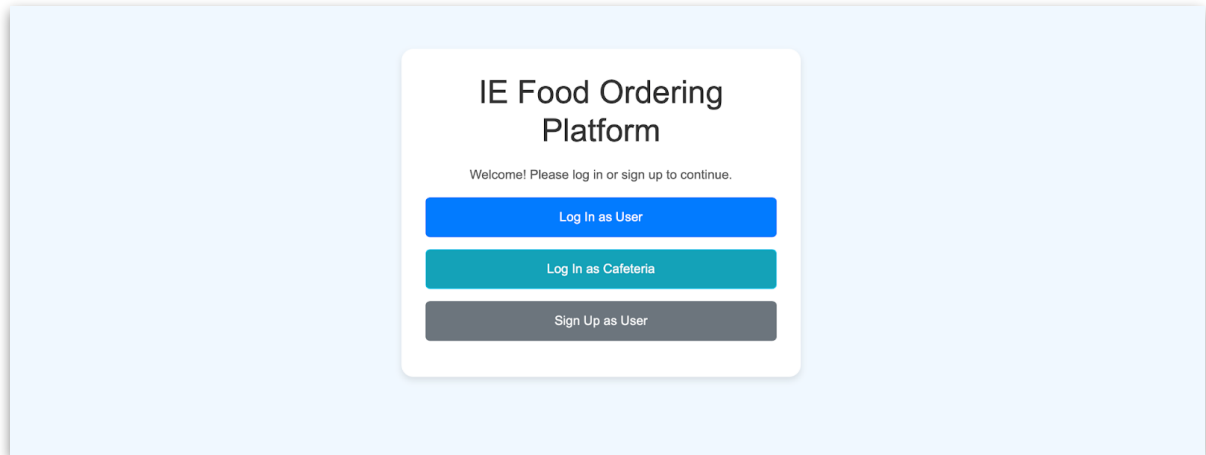
Requirements

- Operating Systems: Windows 10, 11, macOS, Linux.
- Python: Version 3.6 or higher.
- Libraries: Flask

USAGE

Login or Sing Up

At the start, you will be asked whether you are a buyer or a worker.



Buyer:

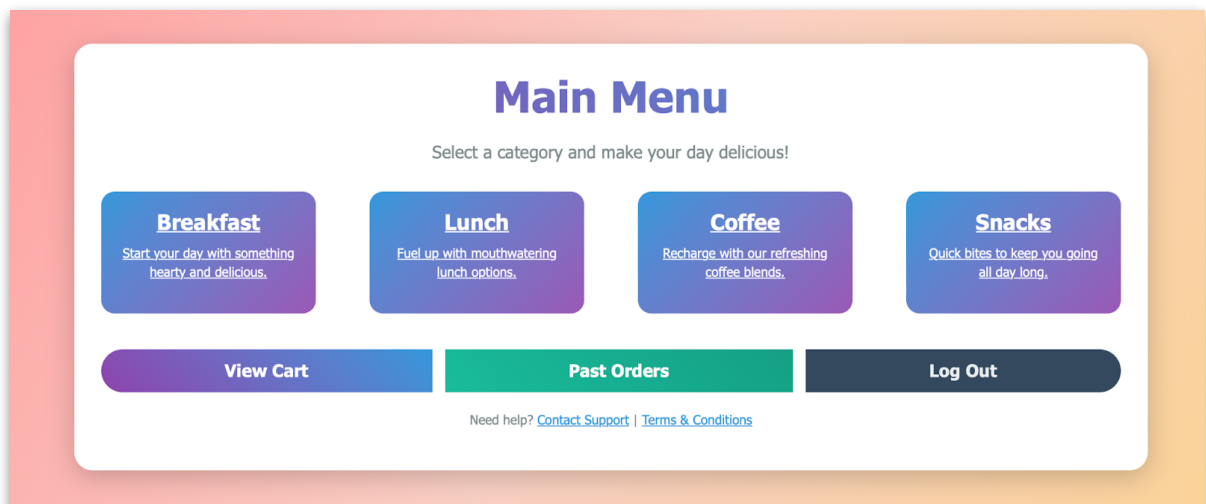
- You will be asked whether you want to log in or sign up.
- Login: Enter your email and password. If they are correct, you will be logged in to the store.
- Sign Up: Enter a new email and password to create an account.

Worker:

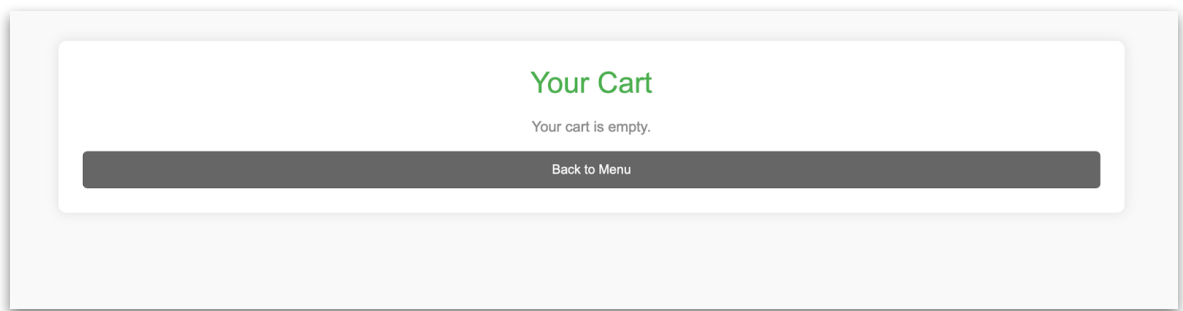
- Enter your worker email and password. If they are correct, you will be logged in to the worker dashboard.

Main Menu

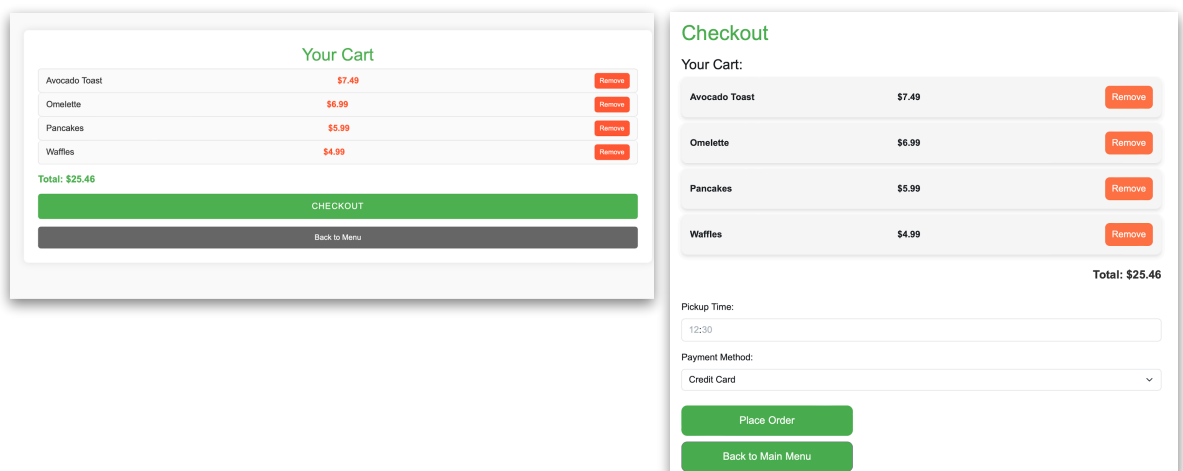
Once logged in as a buyer, you will have access to the main menu with the following options:



- **Filters:** Filter products by category.
 - Breakfast
 - Lunch
 - Coffee
 - Snacks
- **Shopping Cart:** View the products in your cart.
 - If you haven't added anything, it will tell you that your cart is empty.
 - You can view the items in your cart and the total price.



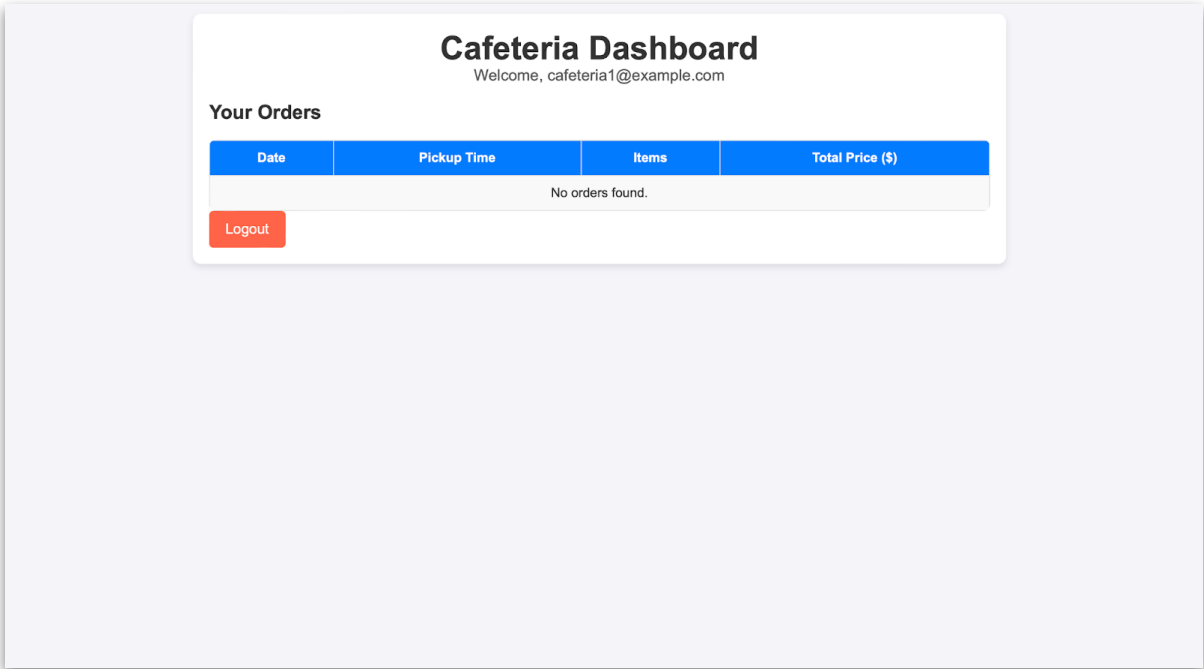
- **Past Orders:** View your past orders.
- **Checkout:** Proceed to pay for the products in your cart.
 - If you have products in your cart, you can proceed with checkout.
 - Choose a payment method: Apple Pay or PayPal.
 - Then, it will ask for the time you want to pick up your order.
 - The order will be placed, and the cart will be cleared after checkout.



For each category, you will be asked to enter the numbers of the products you want to add to your cart, separated by commas.

Worker Dashboard

If you log in as a worker, you can access the admin dashboard where you can see the orders placed by buyers. You can choose to either view the orders or exit the dashboard.





FURTHER IMPROVEMENTS

- . **Enhanced User Experience:** Use machine learning for personalized recommendations and add user reviews/ratings for informed choices.
- . **Advanced Order Management:** Allow order modifications and exportable order history.
- . **Integration & Compatibility:** Ensure cross-platform functionality (Android & iOS).



CREDITS

This project was developed by Team 6 members:

- Austin, Levi Singer
- Juzgado García-Aranda, Juan
- Liu, Any Wan Ying
- Zalba Montes, Pelayo
- Terry Sanz Pastor, Pedro
- Drake Saldaña, Miguel

Special thanks to Professor Antonio López Rosell for his guidance.

Algorithms and Data Structures Course

Institution: IE University, Madrid, Spain

This project represents a culmination of teamwork, critical thinking, and programming skills to address real-world challenges faced by students. Thank you for supporting and engaging with our work!