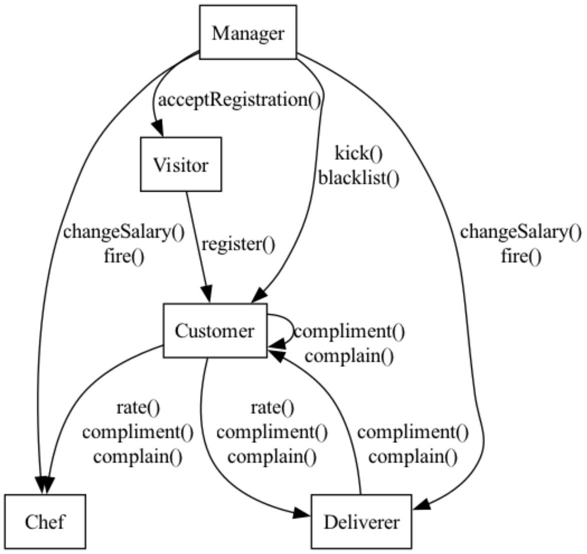
AI Restaurant Phase 2 Report

Team R

Introduction

This project aims to create an online digital restaurant with AI and a complex reputation system to ensure that all customers are friendly and reasonable, and all employees have excellent quality. Users can discuss dishes, chefs, and anything else related to the restaurant in a separate forum. The system has five types of users: Managers, visitors, customers, deliverers, and chefs.



Managers

For simplicity, in this project, we assume one manager who raises and cuts the other employees' salary and hires them and fires them. The manager also decides if complaints are valid, and decides to admit new customers. (As an optional feature, we might require that the person registering share their location and/or IP address, and we can confirm if the user is a new user and not an alternate account made by a kicked-out user). By our decision, managers will also be in charge of adding the chefs' new dishes and deciding their prices.

Visitors

Visitors can only browse the site and view menus. However, visitors can apply to become customers.

Customers

Customers can browse the menu and order items, and ask questions to the AI bot/knowledge base. They can send messages/posts, rate dishes and deliveries (but not the chefs or deliverers themselves, according to our interpretation of the spec). They operate under a system of warnings. After three warnings, the customer is deregistered. Customers get a warning each time they attempt to spend more money than they have, or when they make a complaint deemed without merit by the manager. Customers can also contribute to the local knowledge base, and can compliment or complain about chefs, deliverers, and other customers, and can dispute complaints about them. Based on our interpretation of the spec, complaints about a customer only matter with regard to gaining a VIP status.

VIP Status

A customer automatically becomes VIP if they have no valid complaints on them when they finish spending $100 total or make a 3rd order. That way, customers do not lose their VIP status retroactively, and customers with complaints about them are no longer eligible to gain VIP status. VIP customers get a 5% discount on ordinary orders, get access to special dishes, get free delivery every three orders, (i.e. starting from the third order after gaining VIP status), and their compliments and valid complaints are worth two points instead of one. VIP customers get warnings like ordinary customers, except that they get demoted to ordinary customers after two warnings, with warnings then reset.

Employees

The same rules apply to both chefs and deliverers, both called “employees”, regarding ratings, compliments, complaints, cuts, bonuses, hiring, and firing.
Customers can compliment or complain about employees, and rate their dishes and deliveries. As mentioned above, the manager verifies all complaints. We define an employee’s *score* to be $score(good, bad) = good_{NOT\ VIP} - bad_{NOT\ VIP} + 2(good_{VIP} - bad_{VIP})$ where $good$ is the number of received compliments, bad is the number of received valid complaints, and the subscripts VIP and $NOT\ VIP$ indicate the VIP status of those customers. Employees’ salaries are lowered if their score drops below or equal to -3, and if their score goes back up and drops below or equal -3 again, they are fired. When an employee’s score increases to 3 or higher, they get a bonus. Similarly, an employee whose average rating of their deliveries/dishes drops below or equal to 2 gets demoted, and if it goes up and drops below or equal to 2 again gets fired, and an increase to 4 gives the employee a bonus. (We’ll rely on the manager’s discretion to prevent employees from bribing people to swing his ratings above and below 4 repeatedly). Employees can dispute complaints starting from when they are created up until they are verified.
Here are the only differences:

Deliverers

Deliverers place deliveries made by customers. They bid how much they’ll charge to deliver it, and the lowest bidder wins. Deliverers can compliment or complain about a customer. Deliverers do not get warnings for invalid complaints, but only valid complaints affect customers.

Chefs

Chefs cook the dishes. When the chef wants to add a new dish, the manager decides the price and uploads the information to the site. Chefs can neither rate nor compliment nor complain about anyone.

Other stuff

We haven’t yet figured out how to store the data associated with messages, nor the knowledge base, but the plan so far is to have one general-purpose forum of threads with each thread having a title and many messages. The knowledge base will have its own dedicated table of questions and answers. If a question is found in the knowledge base that shares 90% of the words with those in the input question, the answer is returned. Otherwise, the program will ask the AI engine.

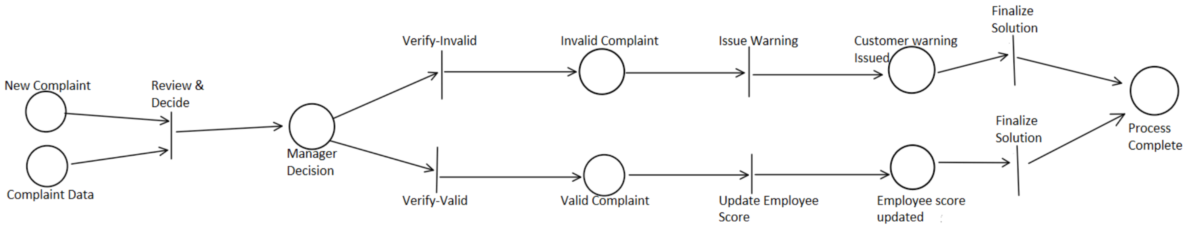
Website Structure

```
index.html
  ai_chat.html
  menu.html
  discussions.html
  register.html
  login.html
    dashboard.html
    cart.html
```

Work in progress. It’s not clear yet how many separate html pages we’re going to have, or if we’re going to cram everything into the dashboard. So far, we have a number of html pages that need to be combined into one, and some that need to be implemented.

Use-cases

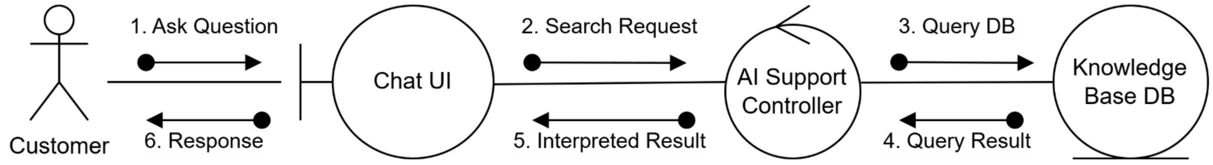
- The reputation management and warning system ensures customers are sane and respect the system, and that employees do their jobs well.



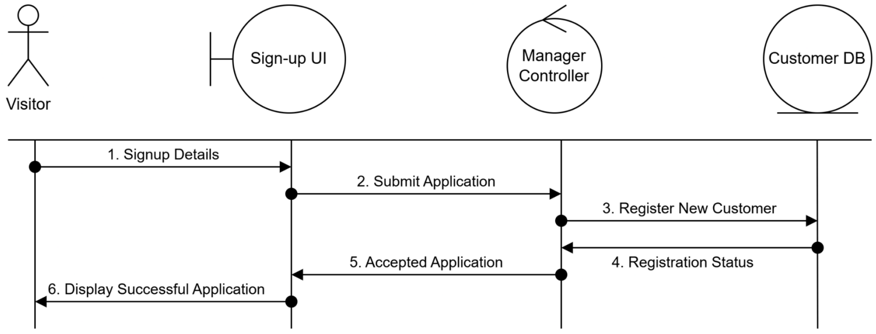
- Users can log in without a captcha.

Customer

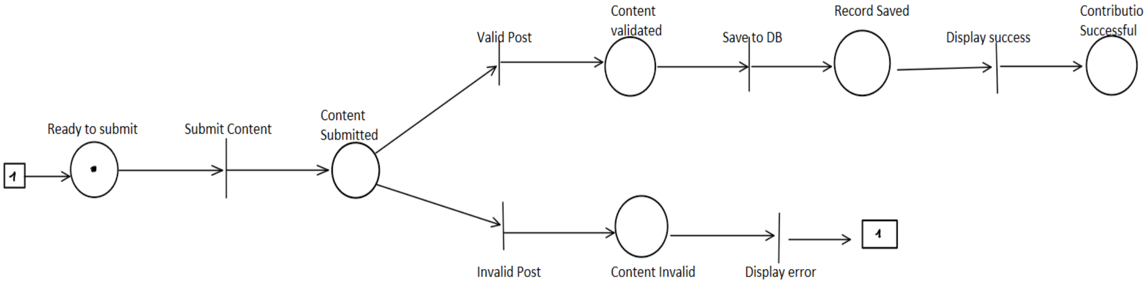
- Customers can ask questions and receive answers from the local knowledge base, and get AI answers as a fallback.



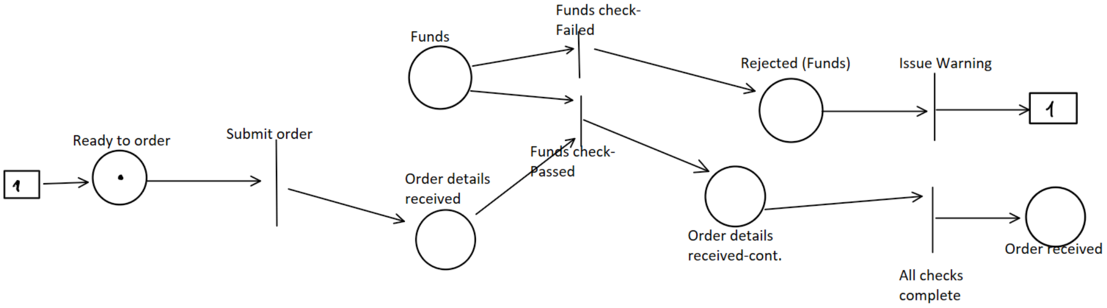
- Customers are manually admitted after registration.



- Customers can discuss anything on a dedicated forum, and can discuss deliverers and chefs.

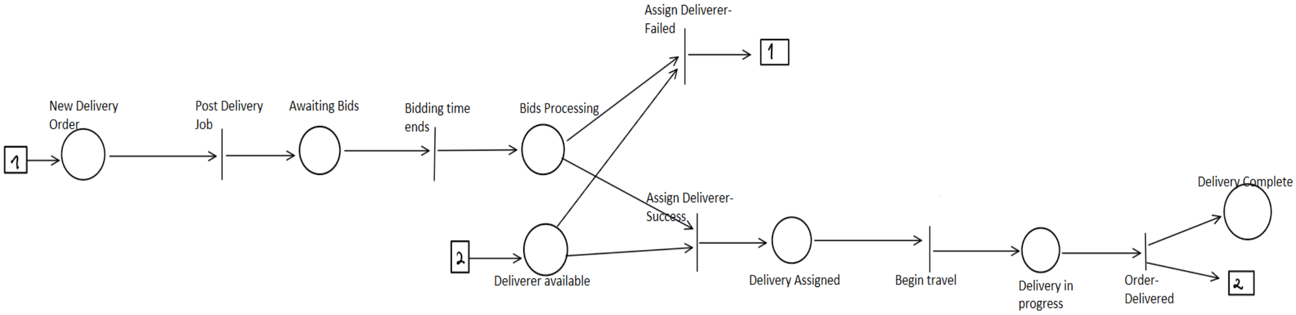


- Customers can order online.



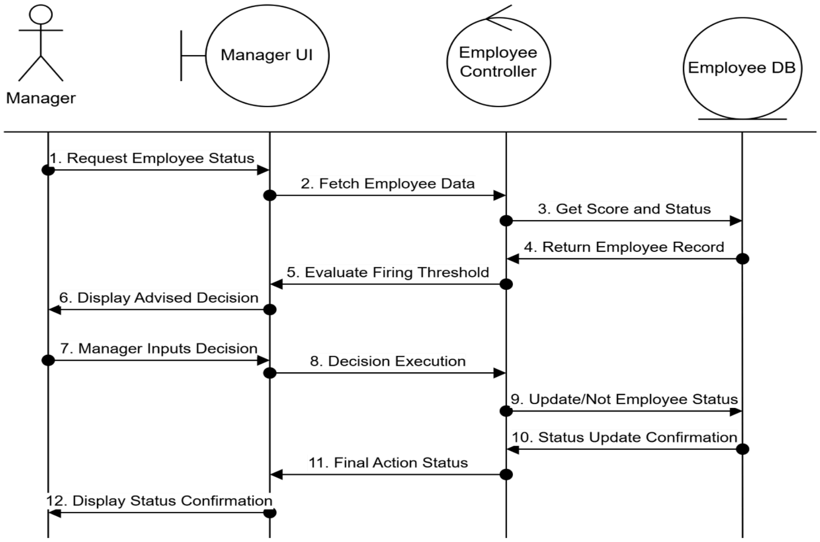
Deliverers

- Deliverers can rate their customers.
- The deliverer who charges the lowest price is hired to save the customer money.



Managers

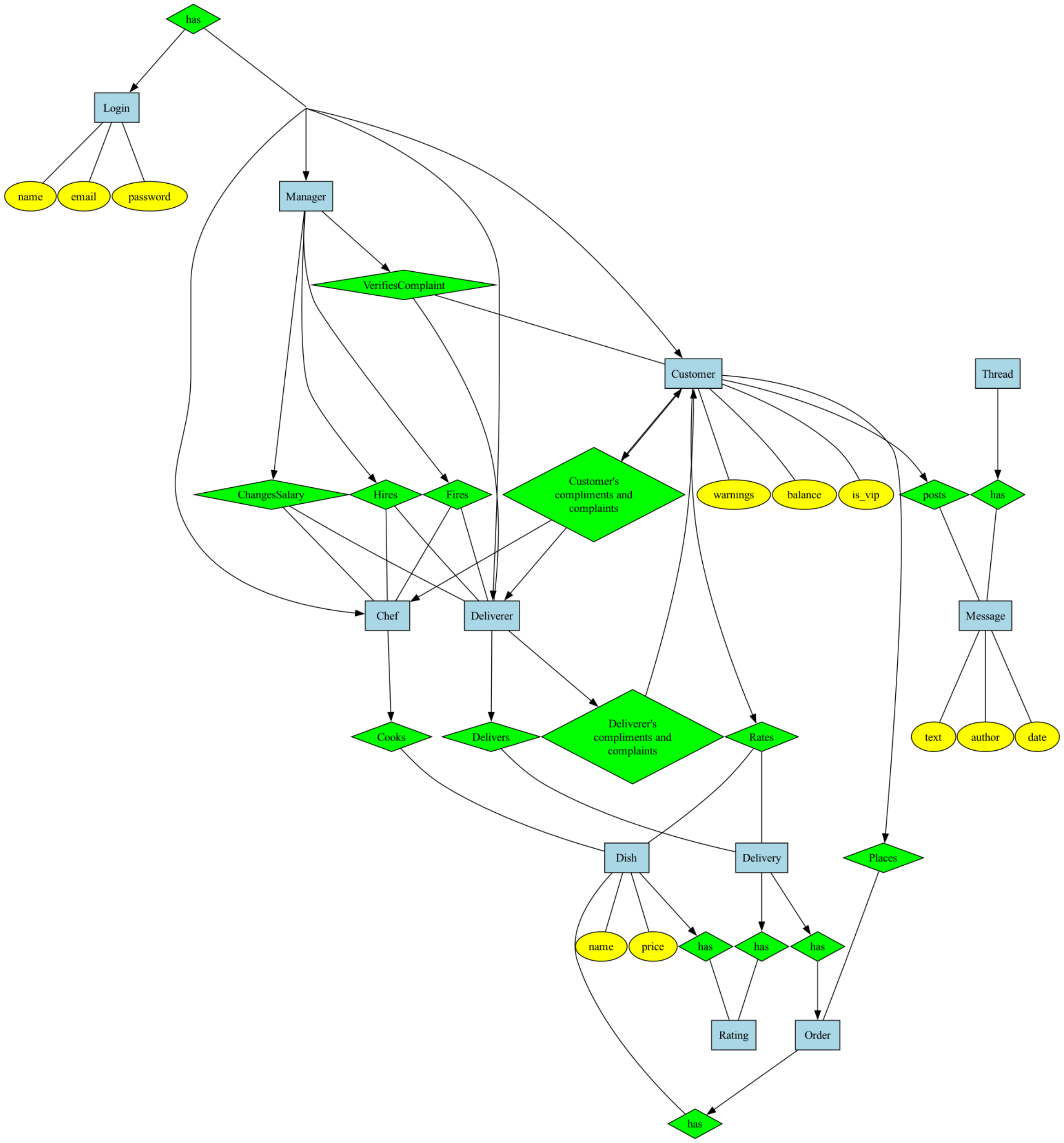
- Managers are given full control over the system, and can manually manage employees' salaries.



Manager
Manager UI
Employee
Controller
Employee DB
1. Request Employee Status
2. Fetch Employee Data
3. Get Score and Status
4. Return Employee Record
5. Evaluate Firing Threshold
6. Display Advised Decision
7. Manager Inputs Decision
8. Decision Execution
9. Update/Not Employee Status
10. Status Update Confirmation
11. Final Action Status
12. Display Status Confirmation

E/R Diagram

Note: The two arrows going between “customer” and “customer compliments and complaints” look like one double sided arrow because of the large scale.



Pseudocode

1. Module

1.1 login_user()

```
# Inputs: username/email, password
# Outputs: dashboard redirect or error

login_user(username_or_email, password):
     Locate user account
     If no account: show 'User not found'
     If wrong password: show 'Incorrect password'
     If customer: go to customer dashboard
     Else: go to staff dashboard
```

1.2 register_user()

```
# Inputs: username, email, password, extra fields
# Output: user created

register_user(user_data):
     Check if username or email exists
     If conflict: show error
     Else: create user and redirect to login
```

2. Customer Features

2.1 get_menu_items()

```
# Input: none
# Output: menu list

get_menu_items():
     Retrieve all dishes and return
```

2.2 update_cart()

```
# Inputs: user_id, dish_id, + or -
# Output: updated quantity
```

```
update_cart(user_id, dish_id, action):
     Find cart item
     If +: increase quantity
     If -: decrease or remove
     Return updated quantity
```

2.3 checkout()

```
# Input: user_id
# Output: order confirmed
checkout(user_id):
     If cart empty: show error
     Calculate total
     If insufficient balance: show error
     Else: place order, deduct balance, clear cart
```

3. Dashboard Methods

3.1 get_customer_stats()

```
# Input: user_id
# Output: balance, warnings, VIP
get_customer_stats(user_id):
     Retrieve and return account stats
```

4. Staff Features

4.1 get_staff_dashboard()

```
# Input: user_id
# Output: role-based dashboard
get_staff_dashboard(user_id):
     If manager: show complaints
     If chef: show ratings + feedback
     If deliverer: show pending orders
```

4.2 list_open_complaints()

```
list_open_complaints():
     Return all unresolved complaints
```

4.3 get_pending_delivery_orders()

```
get_pending_delivery_orders():
    Retrieve unassigned orders and annotate details
```

4.4 submit_bid()

```
# Inputs: deliverer_id, order_id, amount
# Output: confirmation
submit_bid(...):
    Record bid and confirm
```

5. User Profiles

5.1 get_user_profile()

```
# Input: username
# Output: profile data
get_user_profile(username):
    Get info, rating, feedback, return profile
```

6. Forum System

6.1 list_forums()

```
list_forums(search_query optional):
    If empty: return all forums
    Else: return matches
```

6.2 load_thread()

```
load_thread(thread_id):
    Return thread and all messages
```

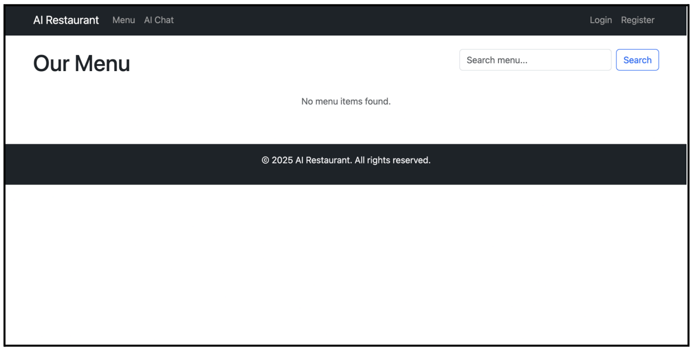
7. Message Posting

7.1 post_message()

```
Input: user_id, thread_id, text
Output: confirmation
post_message(...):
    If empty: show error
    Else: save and confirm
```

User Interface

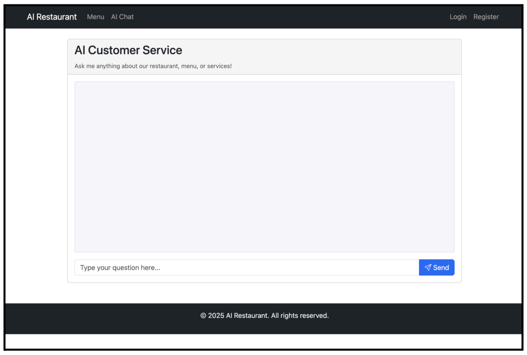
Our project uses a web GUI powered by a Django server, hosted by PythonAnywhere at https://sapphirebrick613.pythonanywhere.com. It's not functional yet, but there's work in progress.



Our Menu

No menu items found.

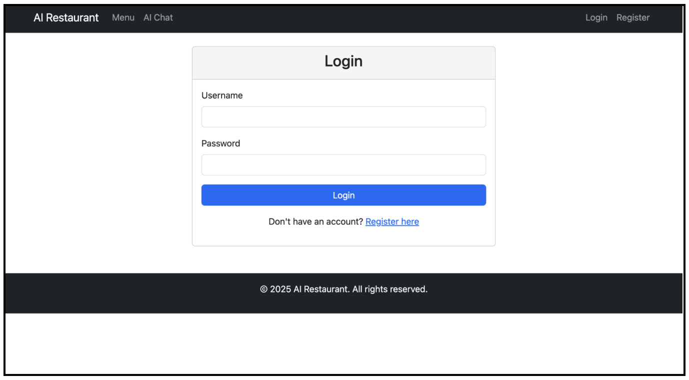
© 2025 AI Restaurant. All rights reserved.



AI Customer Service

Ask me anything about our restaurant, menu, or services!

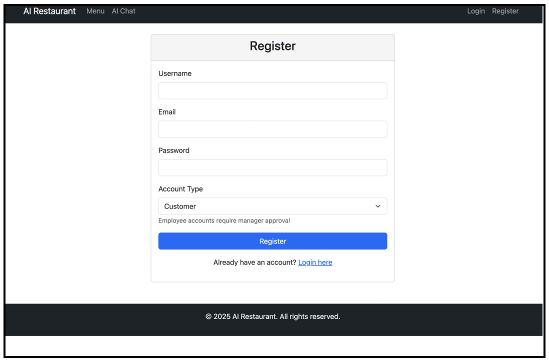
© 2025 AI Restaurant. All rights reserved.



Login

Don't have an account? Register here

© 2025 AI Restaurant. All rights reserved.



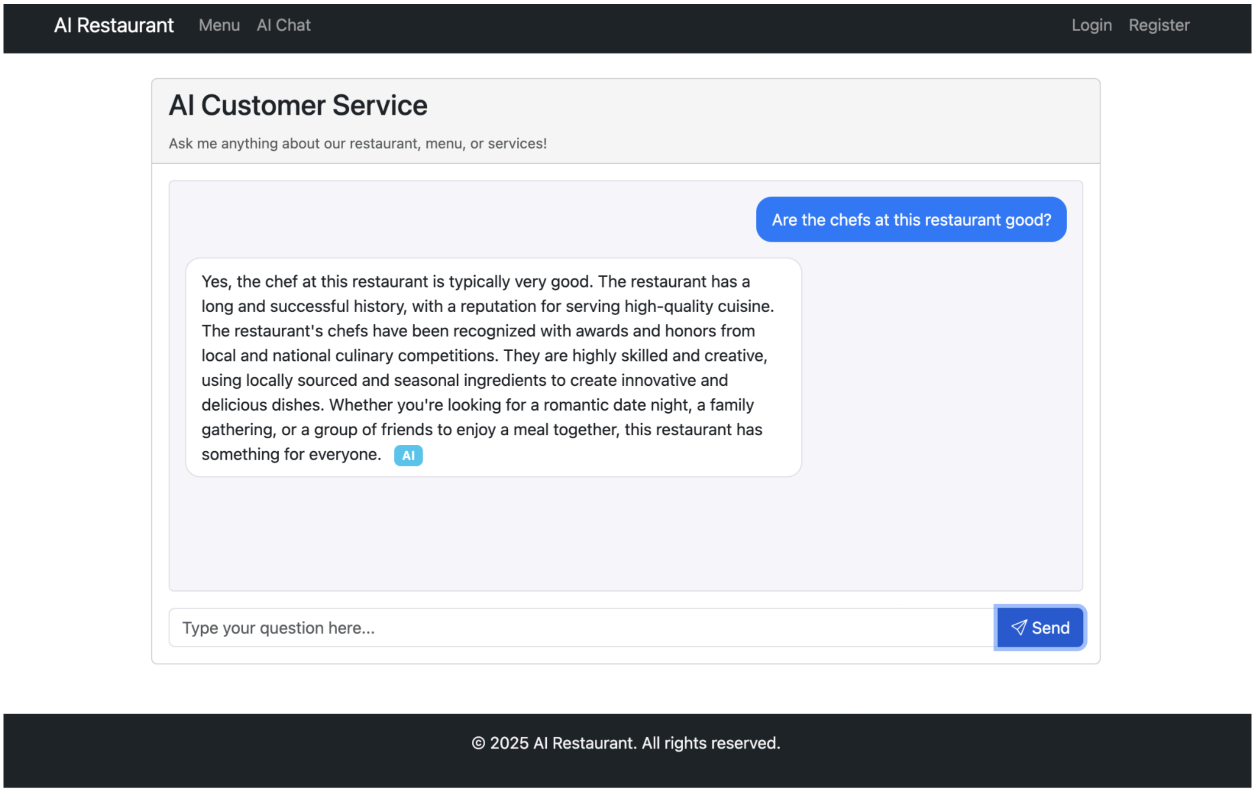
Register

Account Type: Customer

Employee accounts require manager approval

Already have an account? Login here

© 2025 AI Restaurant. All rights reserved.



AI Customer Service

Ask me anything about our restaurant, menu, or services!

Are the chefs at this restaurant good?

Yes, the chef at this restaurant is typically very good. The restaurant has a long and successful history, with a reputation for serving high-quality cuisine. The restaurant's chefs have been recognized with awards and honors from local and national culinary competitions. They are highly skilled and creative, using locally sourced and seasonal ingredients to create innovative and delicious dishes. Whether you're looking for a romantic date night, a family gathering, or a group of friends to enjoy a meal together, this restaurant has something for everyone.

© 2025 AI Restaurant. All rights reserved.

Sample Prototype

The AI engine is currently functional. It’s a bit dumb and really slow (sometimes taking up to 3 minutes) but it works: https://sapphirebrick613.pythonanywhere.com/ai_chat.

Group Meetings and Teamwork

Unfortunately, we haven’t had any synchronous group meetings and we’ve been communicating almost exclusively through Discord. Earlier in the project, we struggled to make progress due to overlapping/duplicated code and a lack of communication, but we’re hoping to organize ourselves and start planning our project framework better as soon as possible, to implement all functionality in time to present it when it’s due.

Repository

Our source code is now publicly available and hosted on Github:
https://github.com/RaphiSpoerri/AIRestaurant.
This document is found in the “docs” folder.