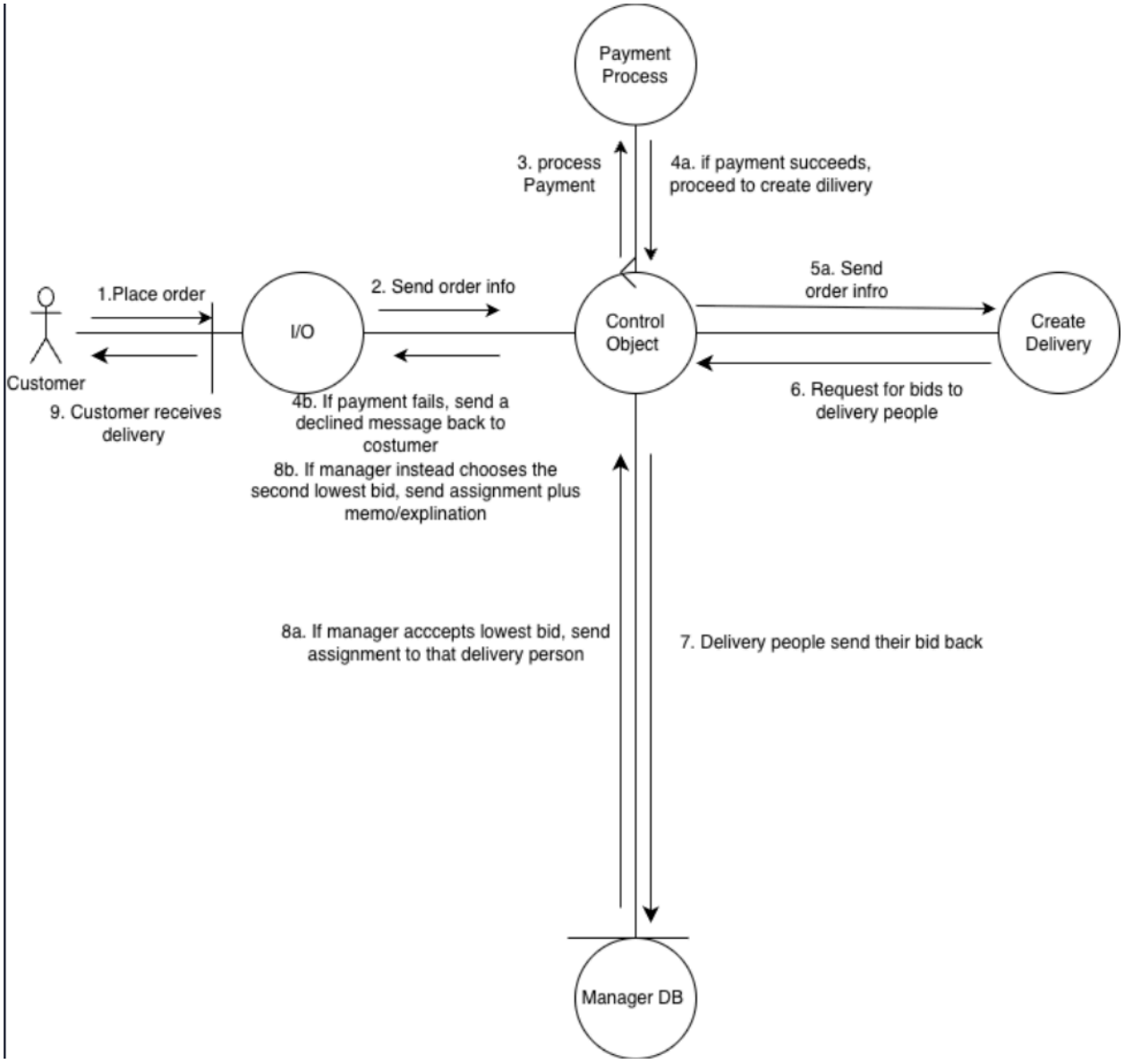
grade: 90
comments: cannot just jump to diagrams in a report, need introduction and other descriptions as a context. The overall class diag is not good, only covered one actor; the use cases are given with good details, the petri-nets are fine, but we need more than 1. The E/R diag is a sloppy rendentaion that is impossible to read by guessing I don't see any "isa" relation, which should group chefs/deli/manger as one "employee" entity to inherit from.

Team S

 # All use cases have diagram files that can be found in the github for further viewing

 # https://github.com/aldaneeeee/Byte-Bite

Class Collaboration Diagram:



why only customer but no all other actors?

Use Cases:

Use Case 1: Browse Menu

Actors: Visitor, Customer, VIP Customer

Purpose: Allow actors to browse and search for food items with pictures, prices, and ratings shown.

Precondition: System is active, menu data is available in the database.

Main Flow:

1. The actor opens the application.
2. The system displays a menu overview with food pictures, reviews, and prices. Also displays the most ordered chef and top rated chef, and an AI chat box is available to ask questions about the restaurant and food.
3. Menu also displays and allows browsing of discussions, reviews, and chef/delivery person profiles
4. The food being shown depends as follows:
    - For visitors and newly registered customers, a viewing of the most ordered and top rated dishes.
    - For registered and VIP customers, a personalized menu based on their prior orders will be shown. VIP customers will also be shown VIP dishes.
5. Actors can sort, filter. or search for dishes based on popularity, rating, and price.

Postcondition: A GUI that displays food to be chosen.

Exceptions:

1. Missing image for the food:
    - In place of the image, there will be text stating “No image available”
2. Missing description for the food:
    - In place of the description, there will be text stating “No description is available”
3. No ratings for the food:
    - The food will be rated 0 stars, and the reviews page will indicate there are no reviews.

 # This use case has a petri net, available in the github to view

Use Case 2: Register

Actors: Visitor, Manager

Purpose: Enable visitors to create accounts

Precondition: User is not logged in, and they do not have an existing account

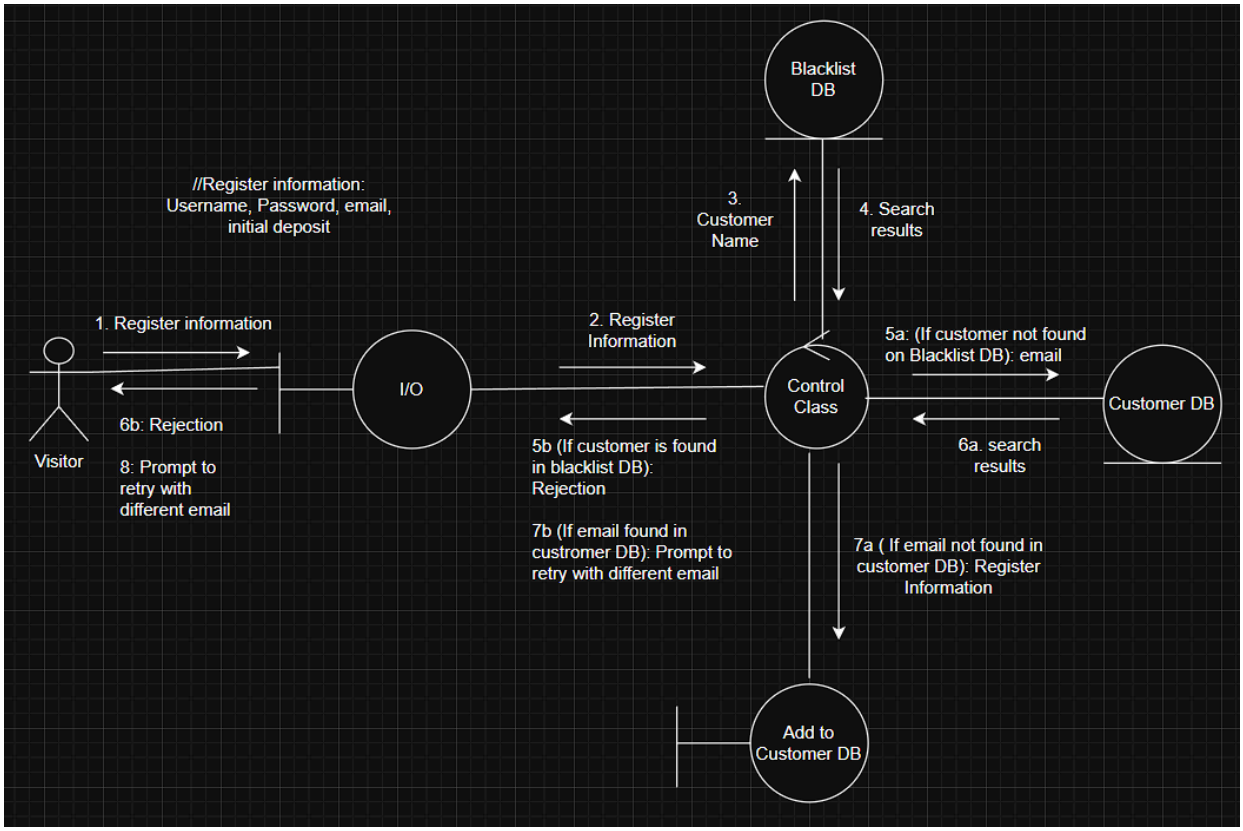
Main Flow:

1. The visitor opens the application and selects the register button
2. The system requests username, password, email, and initial deposit amount.
3. The system checks a blacklist and the user is approved if they don't appear on it.
4. User credentials are stored securely in the system.

Postcondition: A new customer account is made and details are stored in the system database.

Exceptions:

1. Email is already used for an account:
    - An error will be printed, stating "An account using this email has already been registered."
2. Visitor is on the blacklist:
    - An error will be printed, stating "User is on a blacklist and barred from making an account."
3. User payment method doesn't work:
    - The user's account still gets made, and their balance is set to 0. The customer is alerted that their payment did not work.



Use Case 3: Login

Actors: Registered Customer, VIP Customer

Purpose: Allow registered customers and vip customers to log into the application

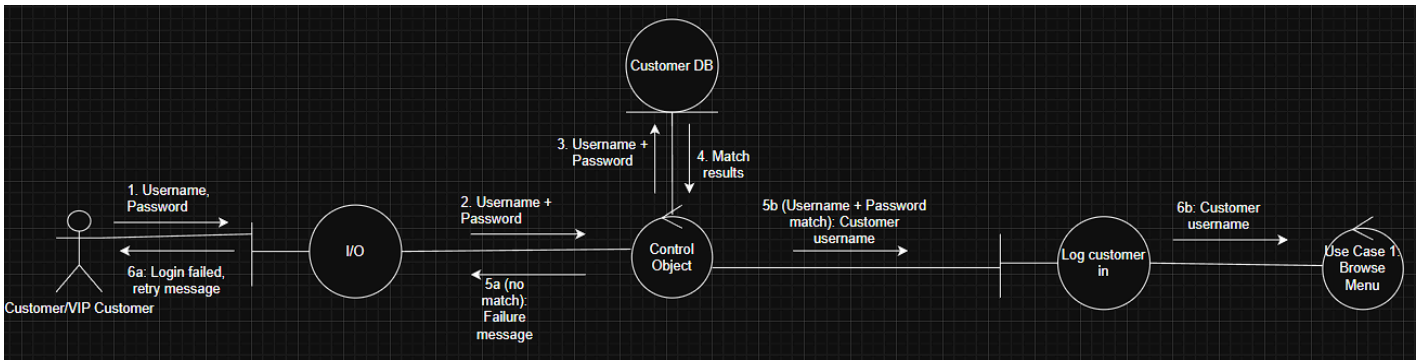
Precondition: User is not logged in, and an account has been made

1. The customer opens the application and selects the login button
2. The system requests username and password. If correctly inputted, the user is redirected to their personalized menu screen.

Postcondition: Existing customer is logged in successfully.

Exceptions:

1. Incorrect username and password combination:
    - The system prints an error stating “incorrect username or password”
2. Missing username or password:
    - The system prints an error stating “All fields must be filled.”



Use Case 4: Ordering Food

Actors: Customer, VIP Customer, Chef, Delivery Person, Manager

Purpose: Enables customers to place and complete food orders with payment and delivery.

Precondition: User is logged in and has sufficient deposit balance.

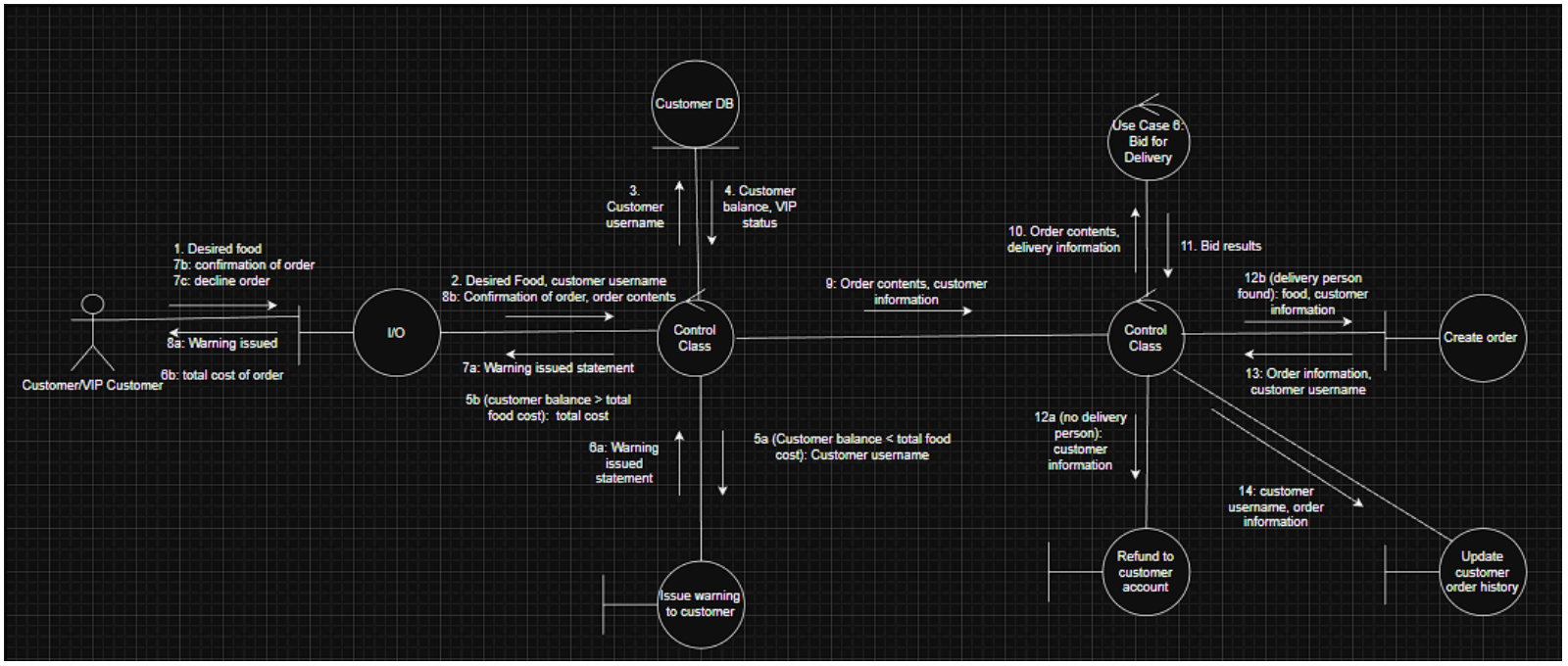
Main Flow:

1. The customer selects desired dishes and adds them to the cart.
2. The system displays the total cost, factoring in VIP discounts and showing the discount if applicable.
3. The customer confirms the order.
4. The system checks the balance of the customer. If it is sufficient, the cost is deducted and the order is confirmed.The order is assigned to the appropriate chef for preparation.
5. Delivery people bid for the job, and the manager assigns the delivery to the lowest bid, or to a higher bid with a justifiable reason as a memo in the system.
6. The delivery person delivers the order to the customer, and the order is logged for customer history and VIP eligibility.

Postcondition: The order is processed, payment is deducted, and customer history is updated

Exceptions:

1. Customer tries checking out with an empty cart:
    - Checkout button does not work until at least one item is in the cart
2. Customer tries to check out with an insufficient balance:
    - The order is rejected and a warning for recklessness is issued. The system prints an error stating “insufficient balance. A warning has been issued for reckless behavior.”



//Petri net

<u>Use Case 5: Image-Based Food Search</u>

Actors: Visitor, Customer, VIP Customer

Purpose: Find desired food based on the provided image.

Precondition: A picture is uploaded to reference for the search.

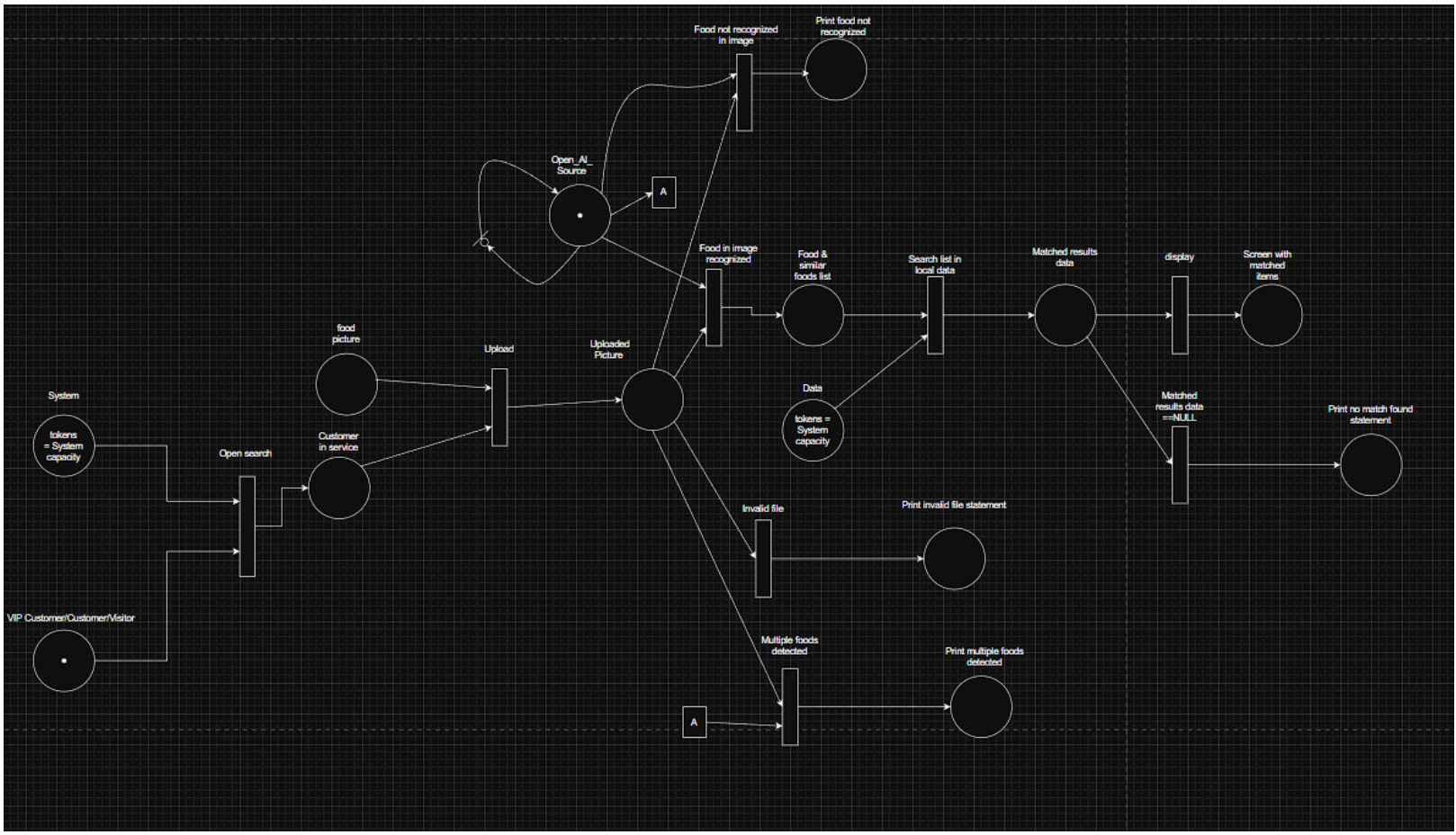
Main Flow:

1. The user uploads the image to the input box.
2. The system uses OpenAI to detect the dish and identify any matches.
3. The system displays similar dishes that are available in the menu.
4. The user can view and order these available dishes.

Postcondition: The customer has received options for the food they wanted.

Exceptions:

1. Invalid file:
    - The system will display an error message letting the user know the file is invalid and display allowed file types and file size.
2. No match found in the database:
    - The system will let the user know that there was no match in the database.
3. Food not recognized in the image:
    - The system will display an error message stating that the food was not recognized.
4. Multiple foods detected:
    - The system will display an error stating that multiple food types were detected
5. The file is too large:



    - The system will display an error stating the file was too large

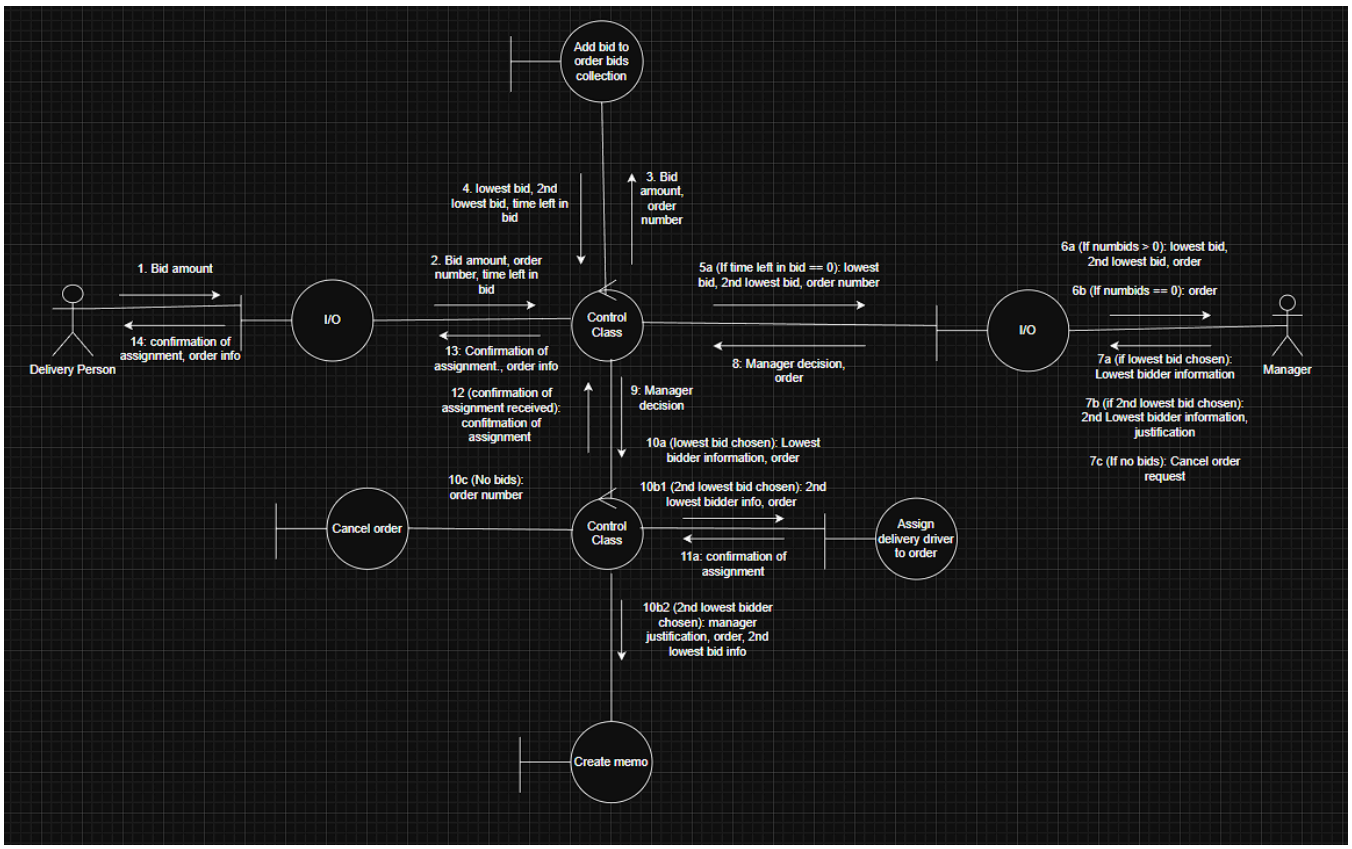
Use Case 6: Delivery Bidding

Actors: Manager, Delivery Person
Purpose: Allow delivery people to bid for new delivery orders and enable the manager to assign jobs fairly.
Precondition: A food order has been placed.
Main Flow:

1. A new delivery order is opened for bidding.
2. Available delivery people place a bid.
3. After a time limit, all bids are displayed to the manager and the bid is closed. The listing is removed from the bidding menu.
4. The manager selects the lowest bid.
    - If the manager wants to, they can select a higher bid but they must provide justification in the form of a memo.

Postcondition: The order is assigned to a delivery person, and a memo is posted if necessary.
Exceptions:

1. Nobody bids before the bidding phase ends:
    - The order is canceled, and the customer is notified.
2. Multiple delivery people bid at the same price and the price is the lowest:
    - The delivery person who made the first bid receives the order.



Use Case 7: Rate and Review Chefs and Delivery People

Actors: Customer, VIP Customer, Delivery Person, Manager, Chef

Purpose: Provide structured feedback on the food and delivery service.

Precondition: A delivery is completed.

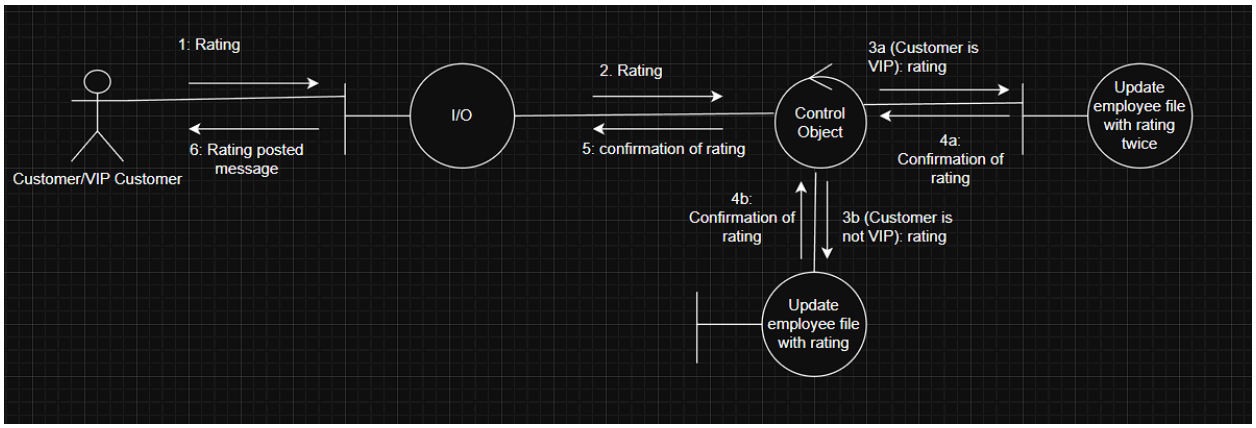
Main Flow:

1. After a delivery completion, the system prompts the customer to rate the food quality and delivery person quality, each on a scale from 1-5 stars. The customer may also add a compliment or complaint separately to the chef and/or delivery person that will be handled by the manager.
2. The system updates the chef's and delivery person's reputation based on the newly added rating, with a VIP customer's feedback counting for twice as much.

Postcondition: New ratings are stored, chef and delivery driver reputations are updated, and bonuses/demotions are handled accordingly.

Exceptions:

1. The delivery gets canceled:
    - Nobody is able to rate anyone on the order.
2. The reviewer doesn't leave any comments with their review:
    - The system returns an error and states that there must be a description in their rating.
3. The reviewer chooses to not leave a review:
    - No adjustment is made to any rating.



Use Case 8: Complaint and Compliment Management

Actors: Customer, VIP Customer, Delivery Person, Manager, Chef

Purpose: The manager handles compliments or complaints filed from customer to chef, customer to customer, customer to delivery person, or delivery person to customer, and hands out warnings as necessary.

Precondition: A compliment or complaint is filed.

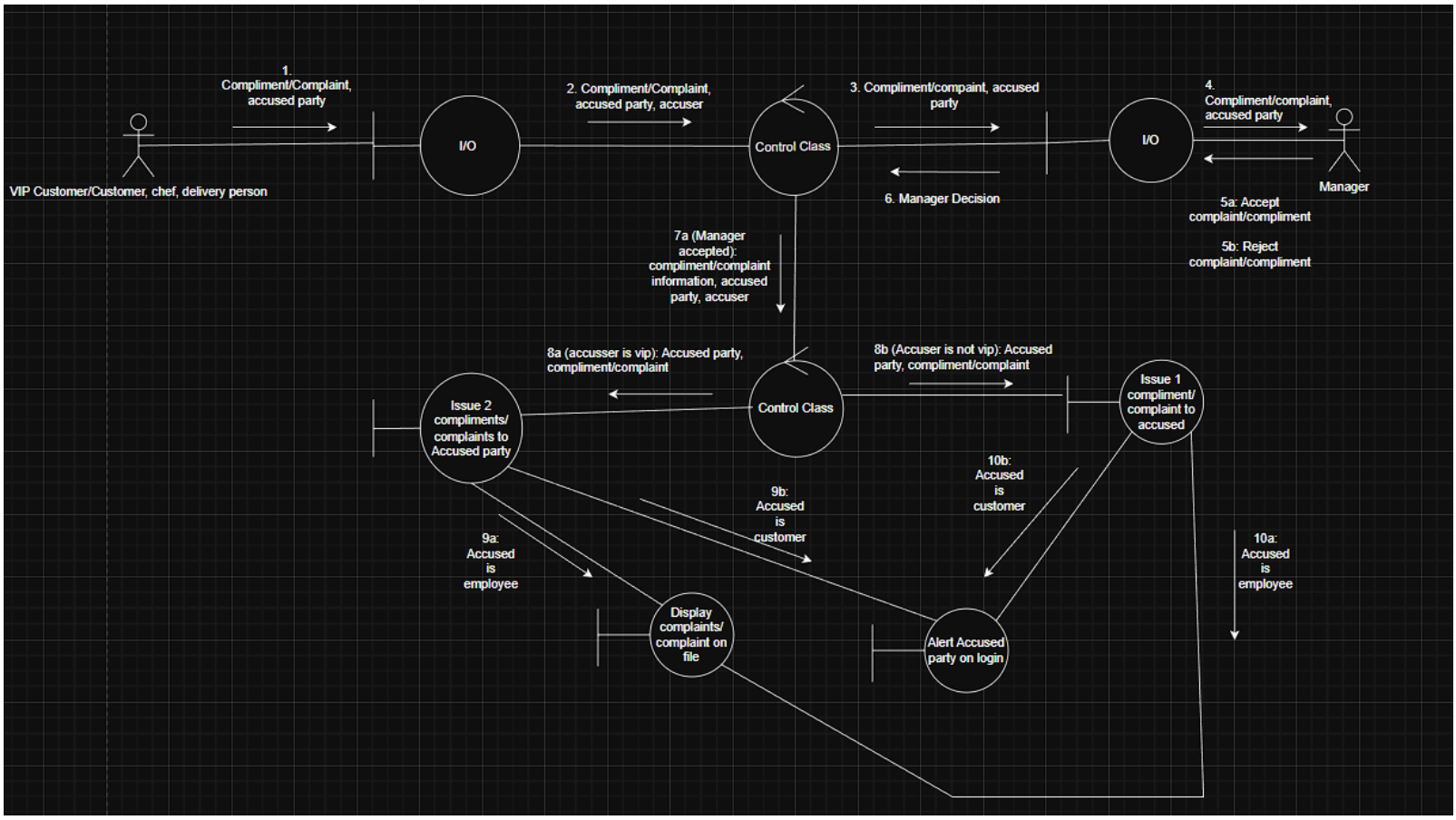
Main Flow:

1. The compliment/complaint gets sent to the manager and shows details regarding who filed it, the reasoning, and who it is about.
2. The party receiving a complaint has the option to refute it.
    - If the complaint is valid, the accused party receives a warning.
    - If a complaint is decided to be without merit, the accuser receives a warning.
3. The manager makes the final decision and the system updates the relevant party's information. If the accuser is a VIP customer, the compliment or complaint holds double the weight.
4. Any warnings or compliments will appear on file for employees, and will be displayed on login for customers.

Postcondition: All compliments and complaints are handled and all warnings are displayed.

Exceptions:

1. A complaint is submitted without any reasoning:
    - The complaint is immediately disregarded.
2. Any field is left empty while completing a compliment or complaint form:
    - The system returns a message stating that all fields must be filled.
3. No appeal of a complaint is filed:
    - A warning is issued to the accused party



<u>Use Case 9: Rewards and Punishments</u>

Actors: Chef, Delivery Person, Manager, Customer, VIP Customer

Purpose: The manager rewards or punishes the chefs and delivery people based on reviews, compliments, and complaints. The manager also rewards and punishes customers and VIP customers based on compliments and complaints.

Precondition: A compliment, complaint, or review has been finalized on an account.

Main Flow:

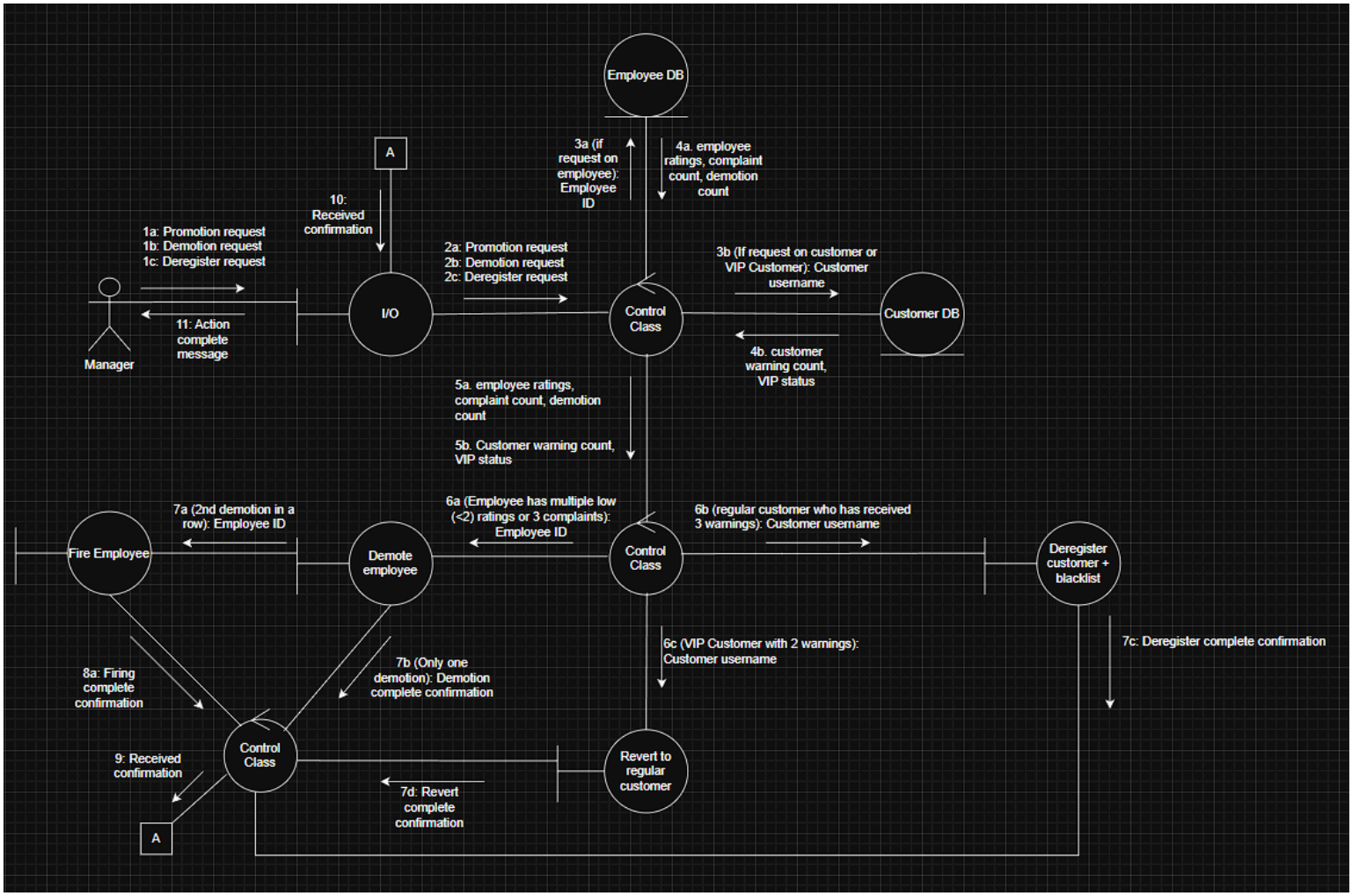
1. The manager checks to see if any action is needed against a party.
    - If a chef or delivery person receives repeated low ratings (<2 stars) or receives 3 complaints, they are demoted. Two demotions lead to firing. If a chef or delivery person receives multiple high ratings (>4) or 3 compliments, they will receive a raise. One complaint cancels out one compliment and vice versa.

- If a registered customer receives 3 complaints/warnings, they are deregistered. If a VIP customer receives 2 complaints/warnings, they are reverted to regular customers and their warnings are cleared.

Postcondition: All reviews, complaints, and compliments cause the affected party to be rewarded or punished accordingly.

Exceptions:

1. The manager tries to demote a regular customer:
   - The system returns an error stating that this is an illegal action.
2. A VIP customer receives 3 complaints to be reviewed and the manager approves the first two:
   - The third complaint under review is removed.



Use Case 10: Account Management

Actors: Customer, Manager, VIP Customer

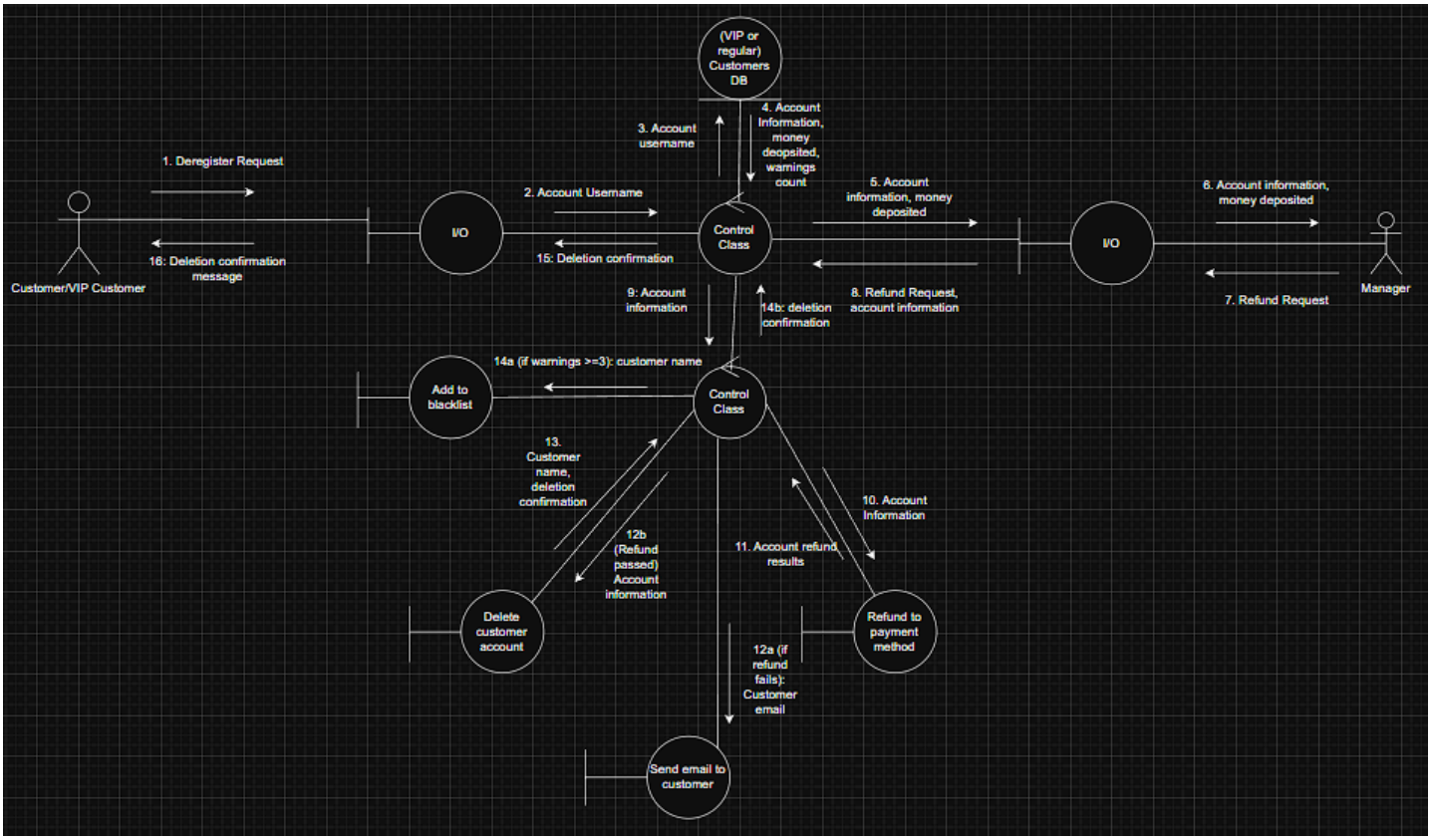
Purpose: Manage account closures and blacklisting.

Precondition: A customer receives 3 warnings or chooses to deregister from the system.
Main Flow:

1. The manager will check to see if the customer has any deposited balance in their account. They will then refund the deposit and close the account.
2. If the account had 3 warnings, the person is also added onto the blacklist. They cannot register for the site again.

Postcondition: Account has been deleted, and any banned customers are added to the blacklist.
Exceptions:

1. The refund for a customer whose account is being closed fails:
    - The system prints an error stating the refund failed, and sends an email to the customer stating to input a form of refund payment within a time period or their account will be closed without a refund.
2. Manager tries to deregister a customer's account that does not have 3 warnings and has not opted to be deregistered:
    - The system returns an error and states that the account does not meet the requirements to be deregistered.
3. A customer who opts to be deregistered receives three warnings before their account is closed:
    - The customer's account is deleted and they are added to the blacklist.



Use Case 11: VIP Promotion:

Actors: Customer

Purpose: Promote a customer to a VIP customer once the conditions have been met.

Precondition: Customer spends more than $100, or makes 3 orders without outstanding complaints.

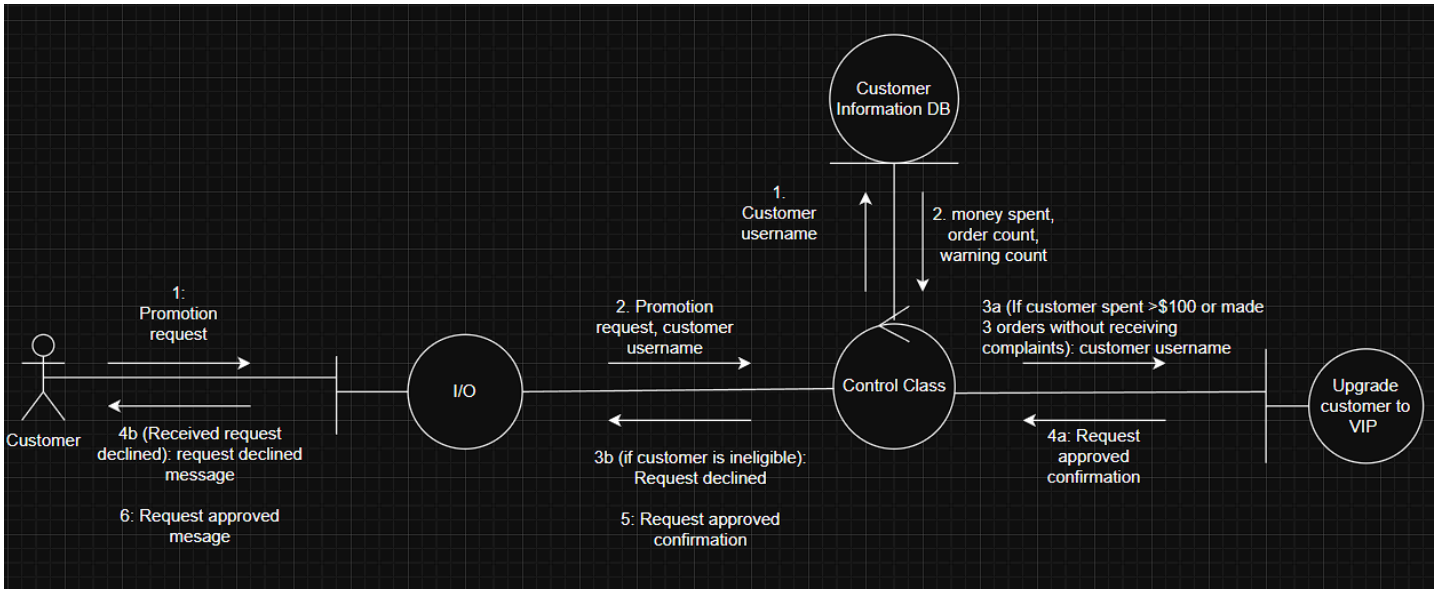
Main Flow:

1. The customer becomes eligible to be promoted to a VIP customer, and requests a promotion to VIP status.
2. The system turns their account into a VIP one, with the following benefits:
    - 5% discount on all orders
    - 1 free delivery every 3 orders
    - access to exclusive dishes
3. The user receives a notification that they are now a VIP customer.

Postcondition: Registered customer is now a VIP customer.

Exceptions:

1. A normal customer is promoted to VIP customer while they have two warnings:
    - The customer is demoted to a regular customer again.
2. An order is canceled and refunded, causing the money spent to be below $100 and they have less than three orders:
    - The customer retains their VIP status.
3. A normal customer applies for VIP without being eligible:



Use Case 12: AI Q&A Service

Actors:Visitor, Customer, VIP Customer, Employee, Manager, AI Assistant

Purpose: Provide automated question answering using local knowledge base and a fallback LLM.

Precondition: A question is asked to the AI assistant chatbot.

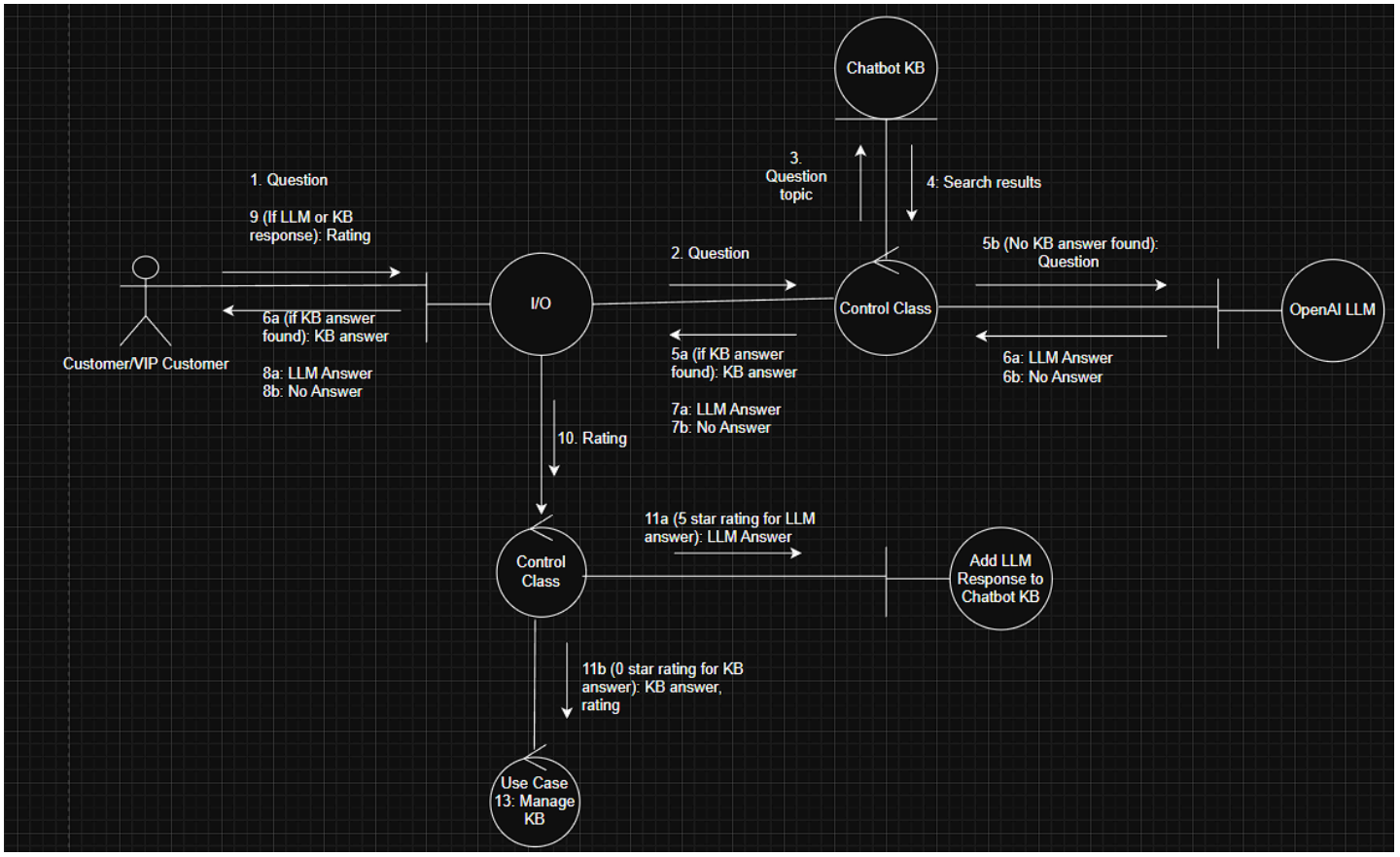
Main Flow:

1. Any user types a question in the chat box.
2. The system searches the local knowledge base for a match. If a match is found, the AI assistant prints the found answer in the chat box.
3. If the answer was found from the local knowledge base, the assistant then asks for a rating from 0-5 based on how helpful the answer is. If the rating is 0, the answer is flagged for the manager to check, and if the content is really bad, the answer will be removed from the KB and the author will not be allowed to add any answers to the KB again.
4. If not, the system will delegate the answer to the OpenAI LLM API, and the found answer will be added to the local KB if it is rated a 5.

Postcondition: The user receives an answer to their question and the local KB is updated.

Exceptions:

1. No suitable answer in the KB and the LLM fails to give an answer:
    - The system will return a message stating “no available answer at this time. Please try again.”
2. System finds multiple answers in the KB regarding the same topic:
    - The system prints out the first found relevant entry, and asks if it answers the question. If not, it continues to the next entry. If no entry answers their question, the system defers to the LLM.
3. System finds a KB entry that is corrupted/missing information:
    - The system deletes that entry and defers to the LLM if no other relevant entry is found.



<u>Use Case 13: Manage Knowledge Base</u>

Actors: Customer, VIP Customer, Manager, AI Assistant

Purpose: Maintain and clean up the local AI knowledge base.

Precondition: An AI answer from the local knowledge base is rated a 0.

Main Flow:

1. The system flags any answers that receive a 0 rating from a customer or VIP customer and reports it to the manager.
2. The manager reviews the flagged entry for accuracy and if it is offensive.
3. If the entry is offensive or incorrect, the entry is removed from the knowledge base.
    - If the entry is removed, the author of said entry is banned from making entries in the knowledge base again.

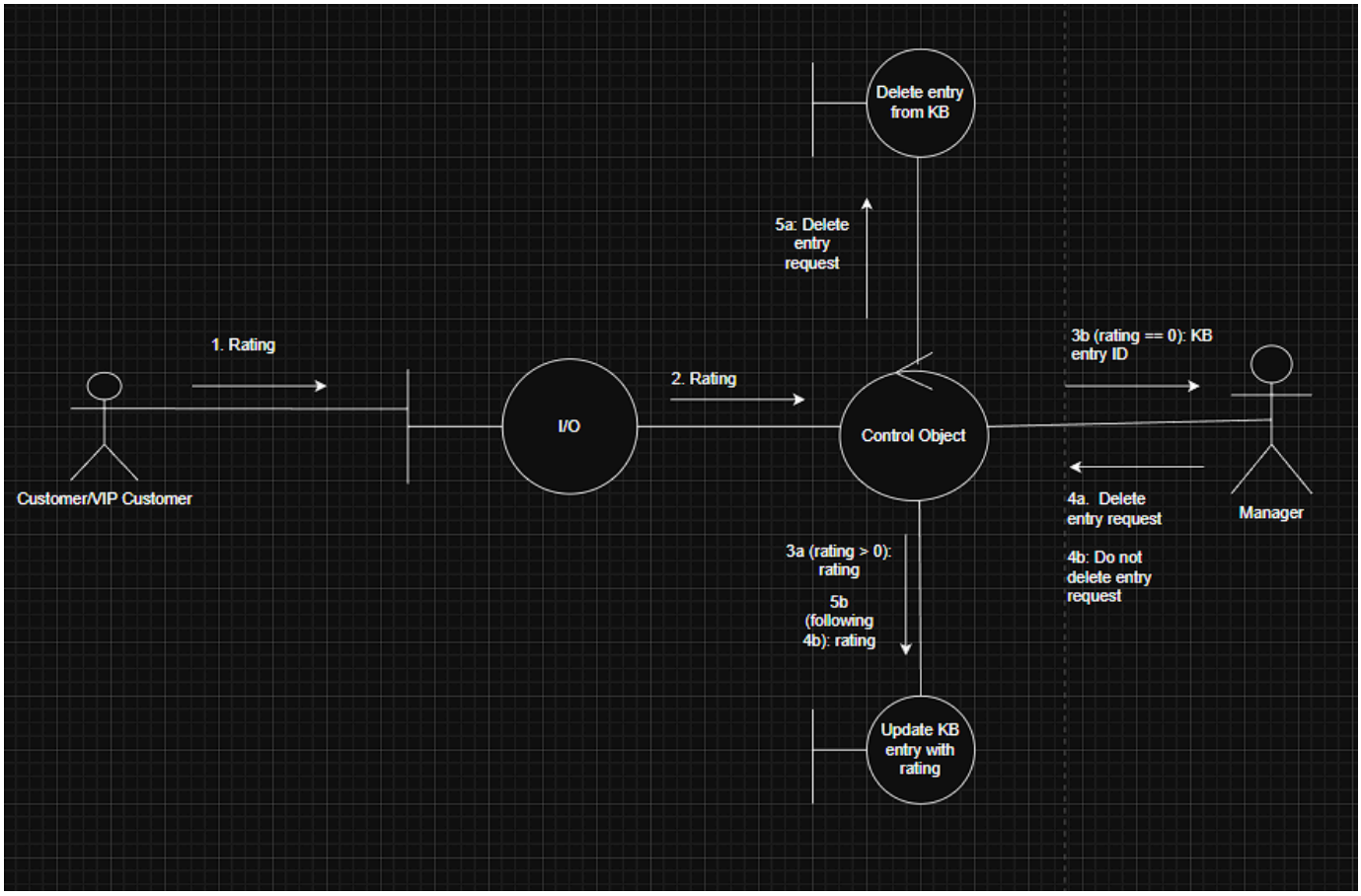
Postcondition: The knowledge base is kept updated with incorrect and offensive content removed.

Exceptions:

1. The flagged entry cannot be retrieved:

- The manager must manually check the database for the entry, and if it is not there the report is ignored.

2. The author for a banned entry is not found:
    - Nobody is banned for that particular report.



<u>Use Case 14: Manage Finances</u>

Actors: Customer, Manager

Purpose: Manage deposits, payments, and refunds efficiently.

Main Flow:

1. The customer deposits funds into their system wallet.
2. During orders, the system deducts costs automatically from the deposit.
3. During an order, if the balance is less than the total order amount, the order is rejected and a warning is issued.
4. The manager can clear the deposit for customers who quit or are deregistered.
5. There is a financial log maintained for auditing.

Postcondition: Account balances, payments, and refunds are accurately recorded.

Exceptions:

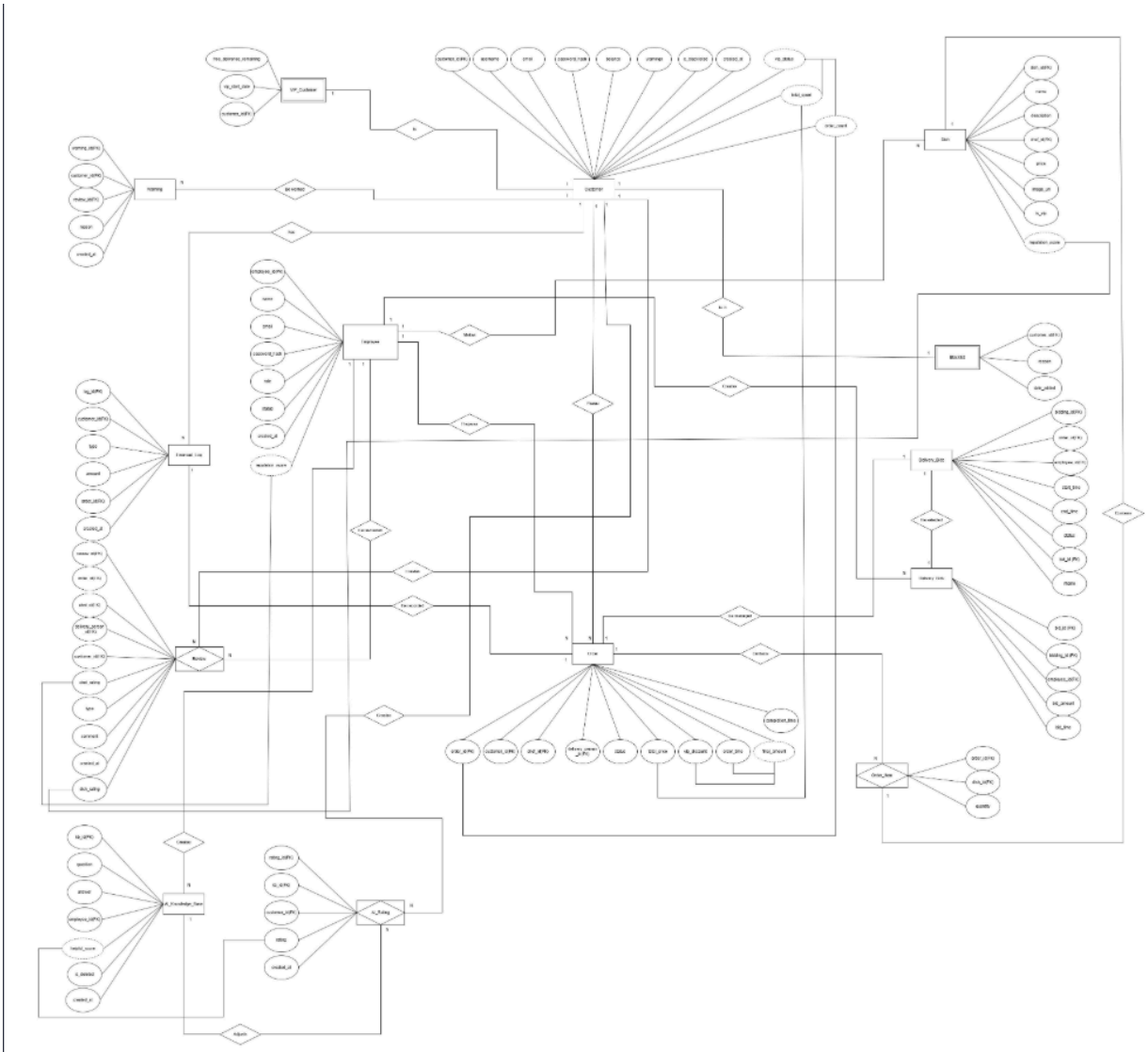
1. Deposit fails or declines:

- The attempt is not logged onto the financial log, and the system returns an error message stating the transaction declined.

2. Refund fails:
    - The system prints an error stating the refund failed, and sends an email to the customer stating to input a form of refund payment within a time period or their account will be closed without a refund.
3. An order is canceled and the customer is refunded:
    - The log gets updated with an entry for the refund.

 # There is a petri net available for this use case on the github

E/R Diagram:



Detailed Design:

Authentication of accounts
RegisterUser Input : username/email, password Output: error/ entry If email exist in the database Then return error (avoiding duplicates) Else Create password Save user email, password Return entry
LoginUser Input: email/username, password Output:user session/ error If user not found , return error If password is invalid, return error Create a session for user Return user session
LogoutUser Input : session token Output: successful log out Delete session token Return success
ResetPassword Input: email, code, new password Output: success/error If code valid and not expired Update user password Return success Else Return error
User profile

GetProfile

input : user ID
Output profile

Get and return profile for user ID

---

updateUserProfile

Input: user ID, profile data
Output: updated profile

Get existing profile
Update either (name, phone, email, preferences,password)
Save profile
Return updated profile

Menu

CheckMenu

Input: search words, filter (maybe)
Output: List of menu Items

Loading all menu items
If searched a key word
Then filter by key word or name
Return filtered list

---

getItem

Input Item ID
Output menu item

Find menu item by ID
Return it

---

createItem

Input : manager ID, item details
Output: confirmation

If manager ID
  Add new item to the database
Else
  Return error

---

updateItem

Input: manager ID, item ID, updates

output : update item

Verify manager/boss ID
Find item to update
Add the updates
Save the changes

Inventory

getInventory

Verify manager ID
Return current inventory list

UpdateInventory

Input: item ID, amount change
Output new amount

Get current amount
Add change value
Save new amount

Cart /checkout

addToCart

Input : item , quantity
Output: success

If item exist
  Add item to cart or Update quantity
else
  Add to cart and ask for quantity
Save cart

removeFromCart

Input: item
Output: success

Find cart for user
Remove item
Save cart

totalSum

Input:user id
output : subtotal , total after tax and delivery fees

For each item in cart (sum x quantity) Apply discount and promotions (maybe) Add tax and delivery fees Return total sum
CheckoutOrder (dummy) Input: userID,payment info, address Output: order confirmation/ error Validate cart and payment info
Order Management
UpdateOrderStatus Input :order ID, status Output
*Potentially might add more *

SYSTEM SCREENS (UI) in Figma:

https://www.figma.com/make/93qC0SYMahvyhGuv9P7cTm/Restaurant-Website?node-id=0-1&p=f&t=v5lKULo9bYpa25hw-0&fullscreen=1

Meeting Memos:

Meeting Memo 1 — Initial Planning

Date: November 8, 2025
Format: Discord Call

Summary:

Discussed the overall scope of the Byte & Bite system.

Assigned each team member to major sections of the Phase II Design Report (Use Cases, ER Diagram, Screens, Pseudocode, Diagrams).

Decisions:

Confirmed that we will use Flask + OpenAI API for final Phase implementation.

Meeting Memo 2 — Use Case & Diagram Review

Date: November 15, 2025
Format: Discord Chat + Quick chat check in

Summary:

Reviewed all major use cases: Order Creation, Payment Processing, Delivery Assignment, and Account Management.

Verified which use cases would require Petri-nets and which would have collaboration diagrams.

Discussed the order flow for the “Process Payment” and “Delivery Assignment” steps to ensure consistency across diagrams.

Did Pseudocode on the call , started a base for all use cases

Github Repo:

https://github.com/aldaneeeee/Byte-Bite