Practical 3

Name : Dhanshree Dharpure
Roll No. : 03
CSE B1

Aim: To design a Class Diagram to represent the structural view of the system.

Theory:

Class scenario name: User

Purpose: Represents a user of the food safety portal, who can register, log in, update account settings, update preferences, and perform various actions like searching, exploring, and planning events in restaurants.

Detailed explanation:

- The User class has several properties like username, password, settings, preferences, and favorite_restaurants.
- The User class has methods like register(), login(), updateAccountSettings(), updatePreferences(), searchRestaurants(), exploreRestaurantDetails(), planGroupEvent(), and saveFavoriteRestaurant().
- The User class has a relationship(composition) with the AccountSettings, UserPreferences, and (aggregation) with Restaurant classes.

Class scenario name: AccountSetting

Purpose: Represents the account settings of a user, such as dietary restrictions, security question, and security answer.

Detailed explanation:

- The AccountSettings class has properties like dietaryRestrictions, securityQuestion, and securityAnswer.
- The AccountSettings class has a relationship (composition) with the User class.

Class scenario name: UserPreferences

Purpose: Represents the user preferences, such as email, totalUsers, and totalRestaurants.

Detailed explanation:

- The UserPreferences class has properties like email, totalUsers, positiveFeedback, regularCustomers, and totalRestaurants.
- The UserPreferences class has a relationship(composition) with the User class.

Class scenario name: Restaurant

Purpose: Represents a restaurant, which has opening hours, promotions, ratings, and menus.

Detailed explanation:

- The Restaurant class has properties like openingHours, promotions, ratings, and menu.
- The Restaurant class has methods like getOpeningHours(), getPromotions(), and getMenu().
- The Restaurant class has a relationship(dependency) with the Menu class.

Class scenario name: Menu

Purpose: Represents a menu, which has menuItems and a name.

Detailed explanation:

- The Menu class has properties like name and menuItems.
- The Menu class has a relationship with the MenuItem class.

Class scenario name: OpeningHours

Purpose: Represents the opening hours of a restaurant, which has days and hours.

Detailed explanation:

● The OpeningHours class has properties like days and hours. ● The OpeningHours class has a relationship(composition) with the Restaurant class.

Class scenario name: Promotion

Purpose: Represents a promotion, which has a name, description, and start and end dates.

Detailed explanation:

- The Promotion class has properties like name, description, startDate, and endDate.
- The Promotion class has a relationship(composition) with the Restaurant class.

Class scenario name: Rating

Purpose: Represents a rating, which has a user, restaurant, rating value, and review.

Detailed explanation:

- The Rating class has properties like user, restaurant, value, and review.
- The Rating class has a relationship(composition) with the User and Restaurant classes.

Class scenario name: OnlineBooking

Purpose: Represents an online booking, which has a user, restaurant, date, time, and payment method.

Detailed explanation:

- The OnlineBooking class has properties like user, restaurant, date, time, and payment.
- The OnlineBooking class has methods like confirm() and cancel(). ● The OnlineBooking class has a relationship(aggregation) with the User, Restaurant, and Payment classes.

Class scenario name: Payment

Purpose: Represents a payment method.

Detailed explanation:

- The Payment class has properties like method and amount.
- The Payment class has a relationship(composition) with the OnlineBooking class.

Class scenario name: Feedback

Purpose: Represents feedback, which has a user, content, and rating.

Detailed explanation:

- Feedback class has properties like user, content, and rating. ● The Feedback class has a relationship(aggregation) with the User class.

Class scenario name: FeedbackAnalysis

Purpose: Represents feedback analysis, which has total feedback, positive feedback, and negative feedback.

Detailed explanation:

- The FeedbackAnalysis class has properties like totalFeedback, positiveFeedback, and negativeFeedback.
- The FeedbackAnalysis class has a relationship(composition) with the Feedback class.

Class scenario name: SystemPerformance

Purpose: Represents system performance, which has bookings processed and system uptime.

Detailed explanation:

- The SystemPerformance class has properties like bookingsProcessed and systemUptime.

Class scenario name: Admin

Purpose: Represents an admin user, who can manage user accounts and view system performance.

Detailed explanation:

- The Admin class has properties like username and password. ● The Admin class has methods like manageUserAccounts() and viewSystemPerformance().
- The Admin class has a indirect relationship(association) with the AccountSettings, UserPreferences, and SystemPerformance classes.

Class scenario name: City Employee

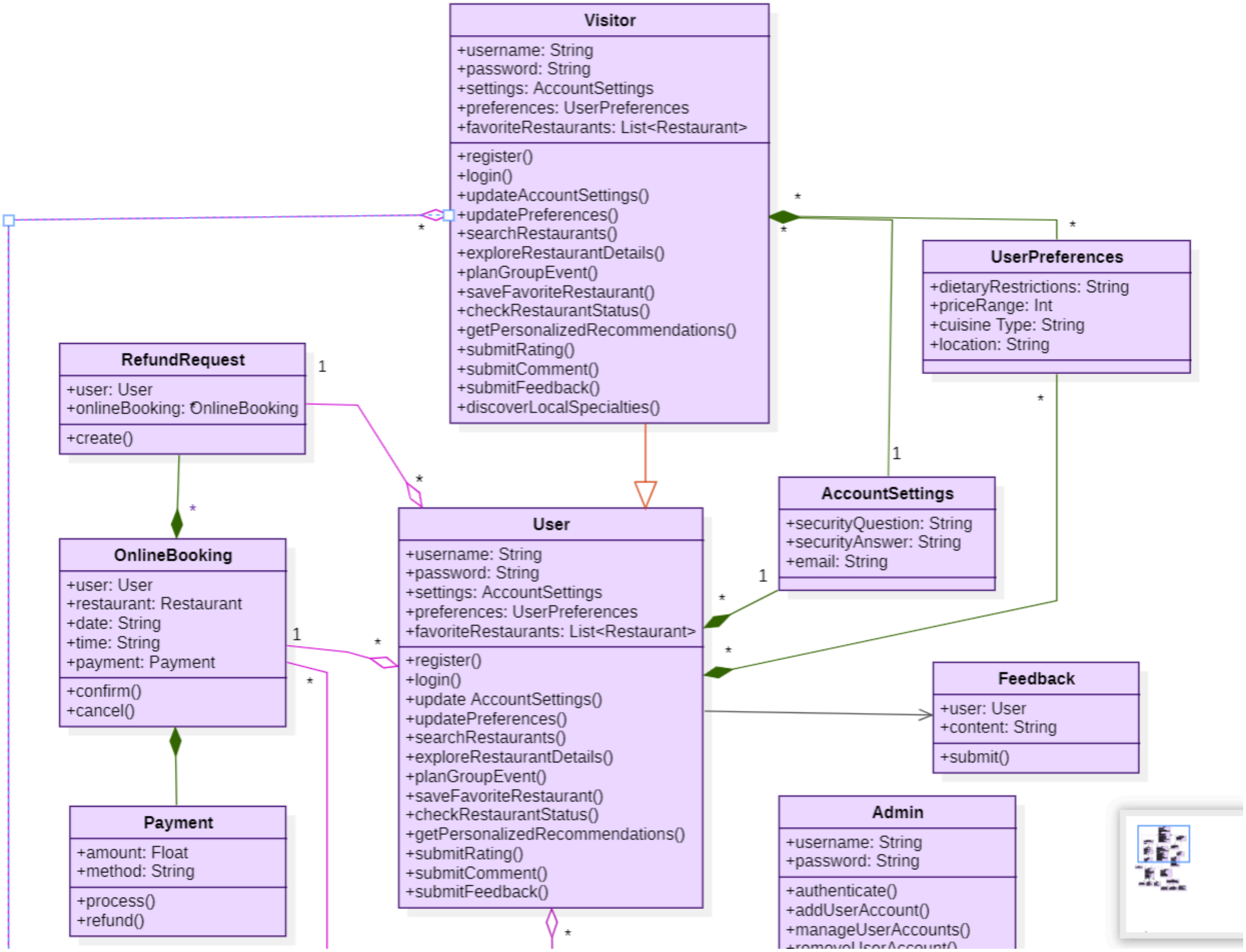
Purpose: Represents a city employee who can view food safety inspection reports for restaurants in the city and update their status.

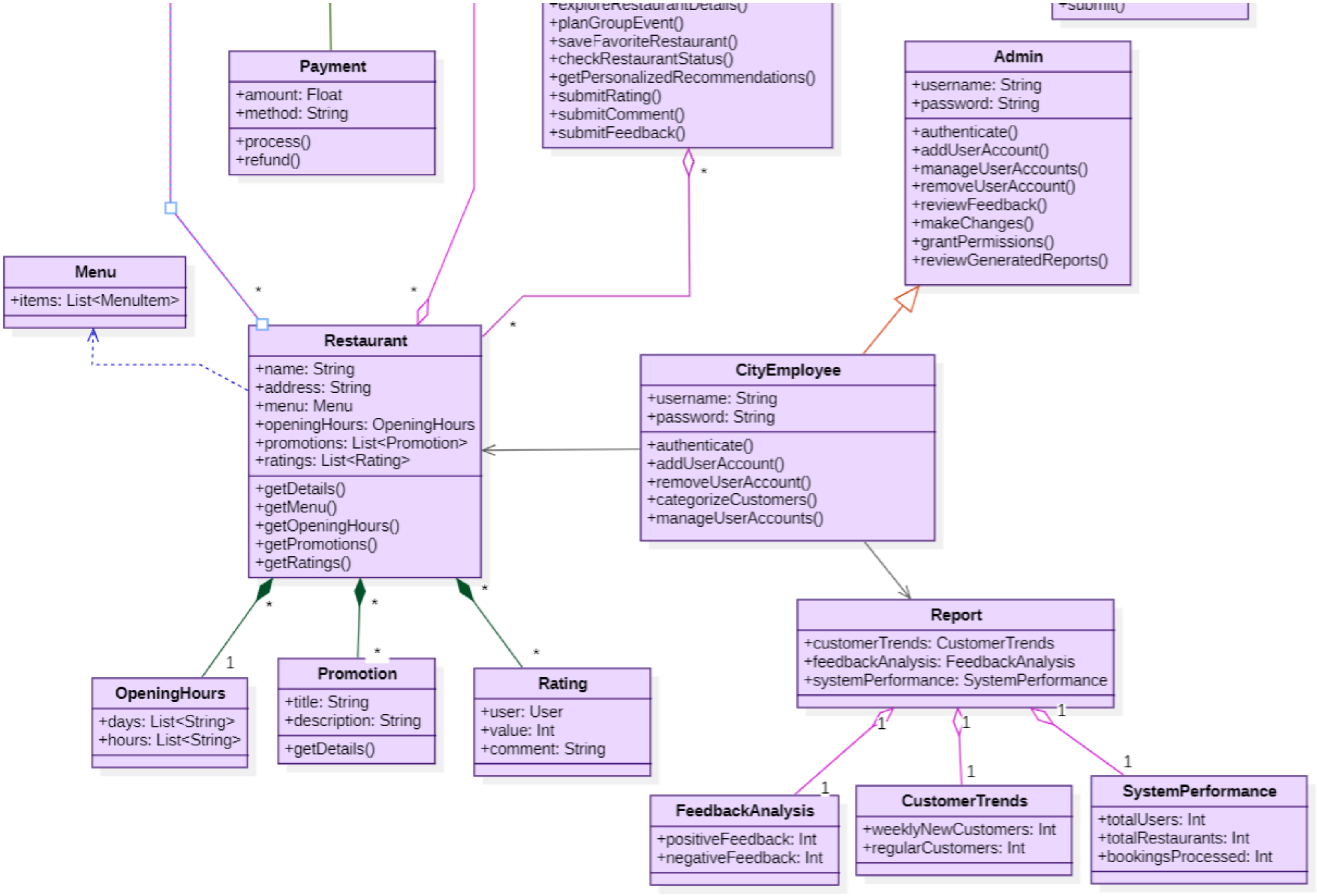
Detailed explanation:

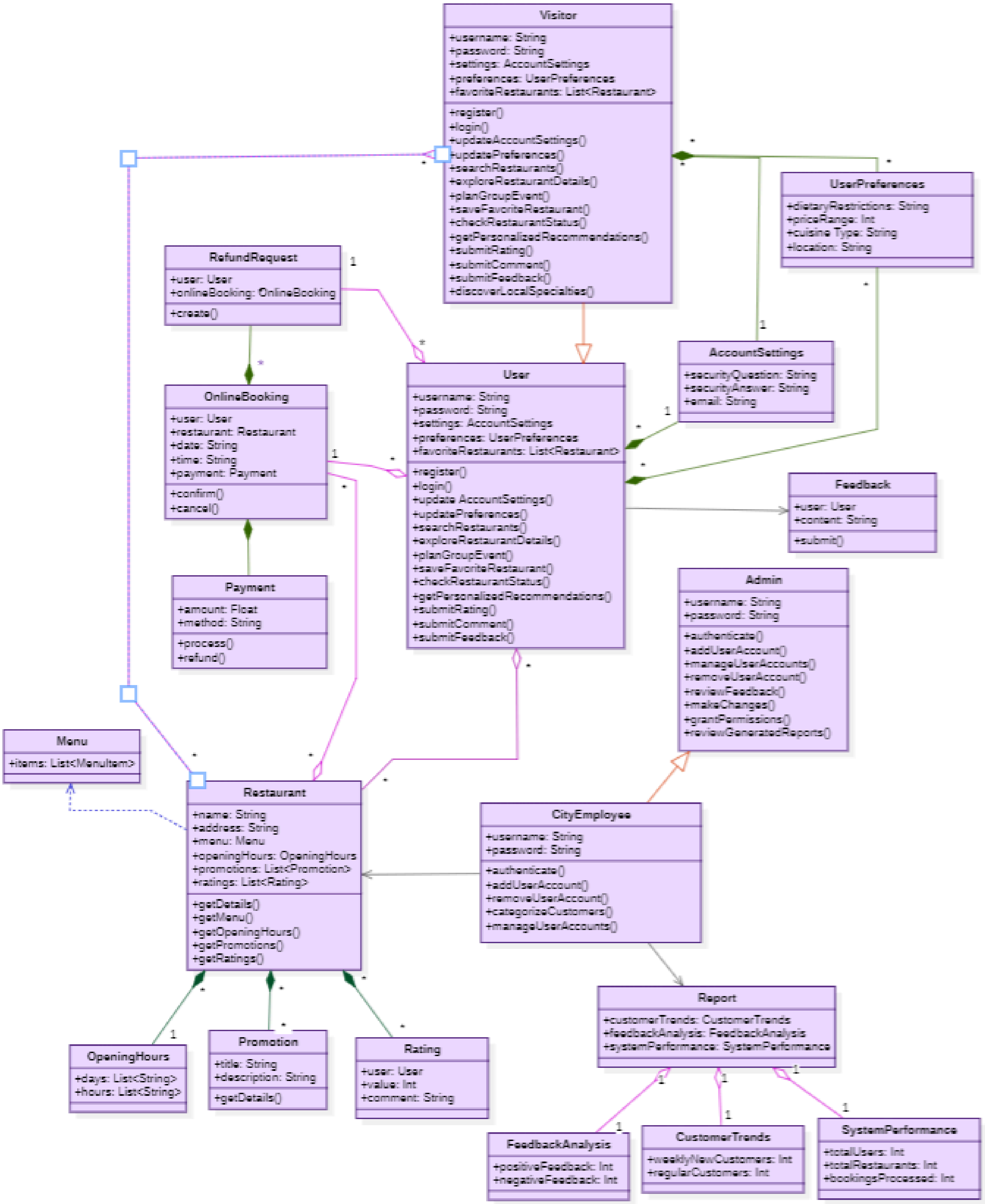
- The CityEmployee class has properties like username and password.
- The CityEmployee class has methods like viewReports() and updateRestaurantStatus().
- The CityEmployee class has a relationship(association) with the Restaurant, Report class.

Result: Class diagram has been designed and studied.

(Screenshot is attached below and also jpeg is uploaded with this docx)


Visitor
+username: String
+password: String
+settings: AccountSettings
+preferences: UserPreferences
+favoriteRestaurants: List<Restaurant>
+register()
+login()
+updateAccountSettings()
+updatePreferences()
+searchRestaurants()
+exploreRestaurantDetails()
+planGroupEvent()
+saveFavoriteRestaurant()
+checkRestaurantStatus()
+getPersonalizedRecommendations()
+submitRating()
+submitComment()
+submitFeedback()
+discoverLocalSpecialties()
UserPreferences
+dietaryRestrictions: String
+priceRange: Int
+cuisine Type: String
+location: String
RefundRequest
+user: User
+onlineBooking: OnlineBooking
+create()
AccountSettings
+securityQuestion: String
+securityAnswer: String
+email: String
User
+username: String
+password: String
+settings: AccountSettings
+preferences: UserPreferences
+favoriteRestaurants: List<Restaurant>
+register()
+login()
+update AccountSettings()
+updatePreferences()
+searchRestaurants()
+exploreRestaurantDetails()
+planGroupEvent()
+saveFavoriteRestaurant()
+checkRestaurantStatus()
+getPersonalizedRecommendations()
+submitRating()
+submitComment()
+submitFeedback()
OnlineBooking
+user: User
+restaurant: Restaurant
+date: String
+time: String
+payment: Payment
+confirm()
+cancel()
Payment
+amount: Float
+method: String
+process()
+refund()
Feedback
+user: User
+content: String
+submit()
Admin
+username: String
+password: String
+authenticate()
+addUserAccount()
+manageUserAccounts()
1
*
1
*
1
1
*
*


+exploreRestaurantDetails()
+planGroupEvent()
+saveFavoriteRestaurant()
+checkRestaurantStatus()
+getPersonalizedRecommendations()
+submitRating()
+submitComment()
+submitFeedback()
+submit()
Payment
+amount: Float
+method: String
+process()
+refund()
Admin
+username: String
+password: String
+authenticate()
+addUserAccount()
+manageUserAccounts()
+removeUserAccount()
+reviewFeedback()
+makeChanges()
+grantPermissions()
+reviewGeneratedReports()
Menu
+items: List<MenuItem>
*
*
*
*
Restaurant
+name: String
+address: String
+menu: Menu
+openingHours: OpeningHours
+promotions: List<Promotion>
+ratings: List<Rating>
+getDetails()
+getMenu()
+getOpeningHours()
+getPromotions()
+getRatings()
CityEmployee
+username: String
+password: String
+authenticate()
+addUserAccount()
+removeUserAccount()
+categorizeCustomers()
+manageUserAccounts()
*
*
*
1
*
*
OpeningHours
+days: List<String>
+hours: List<String>
Promotion
+title: String
+description: String
+getDetails()
Rating
+user: User
+value: Int
+comment: String
Report
+customerTrends: CustomerTrends
+feedbackAnalysis: FeedbackAnalysis
+systemPerformance: SystemPerformance
1
1
1
1
1
1
FeedbackAnalysis
+positiveFeedback: Int
+negativeFeedback: Int
CustomerTrends
+weeklyNewCustomers: Int
+regularCustomers: Int
SystemPerformance
+totalUsers: Int
+totalRestaurants: Int
+bookingsProcessed: Int


Visitor
+username: String
+password: String
+settings: AccountSettings
+preferences: UserPreferences
+favoriteRestaurants: List<Restaurant>
+register()
+login()
+updateAccountSettings()
+updatePreferences()
+searchRestaurants()
+exploreRestaurantDetails()
+planGroupEvent()
+saveFavoriteRestaurant()
+checkRestaurantStatus()
+getPersonalizedRecommendations()
+submitRating()
+submitComment()
+submitFeedback()
+discoverLocalSpecialties()
UserPreferences
+dietaryRestrictions: String
+priceRange: Int
+cuisine Type: String
+location: String
RefundRequest
+user: User
+onlineBooking: OnlineBooking
+create()
AccountSettings
+securityQuestion: String
+securityAnswer: String
+email: String
OnlineBooking
+user: User
+restaurant: Restaurant
+date: String
+time: String
+payment: Payment
+confirm()
+cancel()
User
+username: String
+password: String
+settings: AccountSettings
+preferences: UserPreferences
+favoriteRestaurants: List<Restaurant>
+register()
+login()
+update AccountSettings()
+updatePreferences()
+searchRestaurants()
+exploreRestaurantDetails()
+planGroupEvent()
+saveFavoriteRestaurant()
+checkRestaurantStatus()
+getPersonalizedRecommendations()
+submitRating()
+submitComment()
+submitFeedback()
Feedback
+user: User
+content: String
+submit()
Payment
+amount: Float
+method: String
+process()
+refund()
Admin
+username: String
+password: String
+authenticate()
+addUserAccount()
+manageUserAccounts()
+removeUserAccount()
+reviewFeedback()
+makeChanges()
+grantPermissions()
+reviewGeneratedReports()
Menu
+items: List<MenuItem>
Restaurant
+name: String
+address: String
+menu: Menu
+openingHours: OpeningHours
+promotions: List<Promotion>
+ratings: List<Rating>
+getDetails()
+getMenu()
+getOpeningHours()
+getPromotions()
+getRatings()
CityEmployee
+username: String
+password: String
+authenticate()
+addUserAccount()
+removeUserAccount()
+categorizeCustomers()
+manageUserAccounts()
Report
+customerTrends: CustomerTrends
+feedbackAnalysis: FeedbackAnalysis
+systemPerformance: SystemPerformance
OpeningHours
+days: List<String>
+hours: List<String>
Promotion
+title: String
+description: String
+getDetails()
Rating
+user: User
+value: Int
+comment: String
FeedbackAnalysis
+positiveFeedback: Int
+negativeFeedback: Int
CustomerTrends
+weeklyNewCustomers: Int
+regularCustomers: Int
SystemPerformance
+totalUsers: Int
+totalRestaurants: Int
+bookingsProcessed: Int