Team name: meal reviews in ranking
Leader: Ibuki Nakamura

Web Page

Homepage:

- ordering by an average menu rating (aka. restaurant rating) (high->low)

Review details page:

- Restaurant name
    - Username (If not input gonna show as anonymous)
    - Menu name (Optinal)
    - Menu category
    - Menu rating
    - Price
    - Comment
    - Map
    - Pictures
- Restaurant rating (an average menu rating)
- Category (Show distinct menu category)

Giving review page:

- Restaurant name
    - search from existing restaurant in the database
    - If not exist add new restaurant (name, map url)
        - If we use Place API, it might is easy to get map url at restaurant
        - https://qiita.com/yoshii0110/items/7938085151f569427af6
- Menu name (Optinal)
- Menu rating
- Price
- Comment
- Pictures (maximum ?? pics)

Development

Development tools:

- image storage: S3
- database: firebase

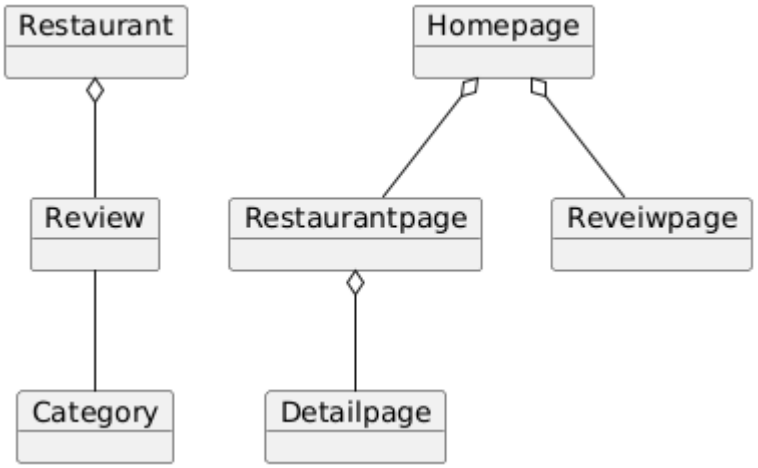
Programming language:

- Frontend: React (Typescript)
- Backend: NodeJS (Typescript)

Functional requirements

- Homepage should show restaurant list order by an average menu rating.
- Homepage has search functions for restaurants.
- User should be able to click on restaurant which links to restaurant details page.
- User should be able to click on the “review” button on the home page and link to giving review page.
- On the giving review page, user should be able to search the restaurant name.
- On the giving review page, user should be able to create a new restaurant if it does not exist in the database.
- On the giving review page, user should be able to input review information.
- On the giving review page, it should show an error if the user does not input all the required information.
- User should be able to see multiple reviews on a restaurant page.
- Restaurant page should display restaurant information along with the reviews that giving by users.

Domain model




```
@startuml
object Review
object Restaurant
object Category

Restaurant o-- Review
Review -- Category

@enduml
```

```
@startuml
object Homepage
object Restaurantpage
object Reveiwpage
object Detailpage

Homepage o-- Restaurantpage
Homepage o-- Reveiwpage
Restaurantpage o-- Detailpage

@enduml
```

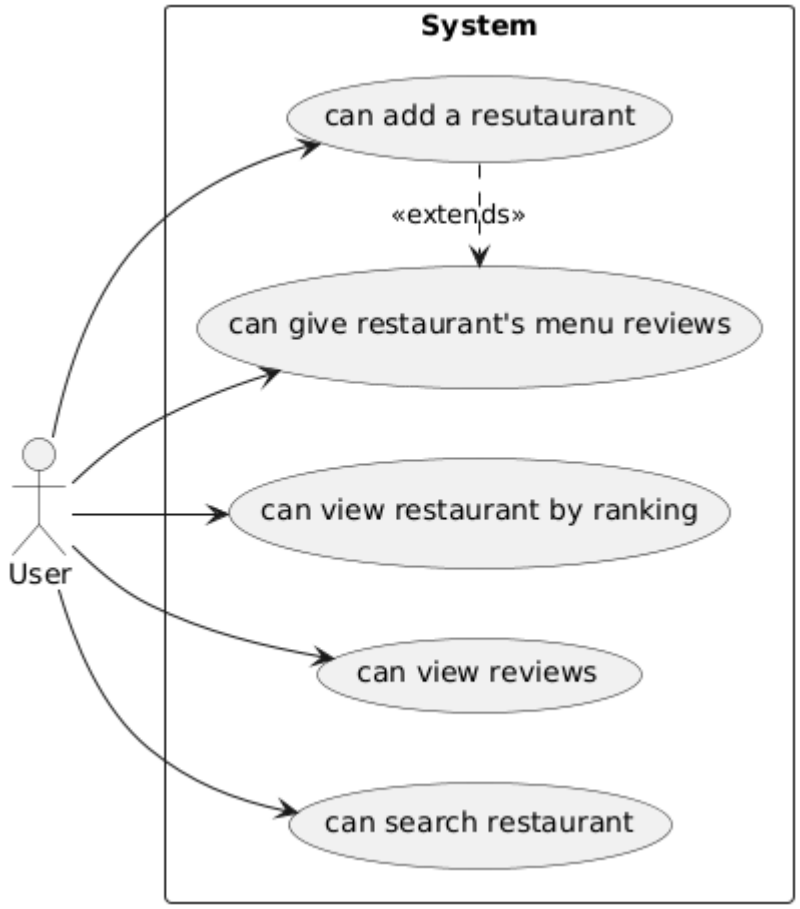
Use case analysis

- Home Page
    - basic flow
        - User landing on homepage.
        - User can see a list of restaurant rankings.
        - User can access to Restaurant Page and Review Page
    - alternate flow:
        - The homepage displays ‘no reviews’ when there are no reviews in the database.
    - alternate flow:
        - User click on the ‘review’ button. Navigating user to Review Page.
    - alternate flow:
        - User click on the restaurant. Navigating user to Restaurant Page.

- Review Page
    - basic flow
        - User search the restaurant.
        - User fill in review information.
    - alternate flow
        - User search for the restaurant and the restaurant does not exist in the database, User should add the restaurant.
    - alternate flow
        - User does not input all the required review information. Shows error message.

- Restaurant Page
    - basic flow
        - User landing on Restaurant Page
        - Page shows reviews of menu

Use case diagram




```
@startuml
left to right direction
actor "User" as user
rectangle System {
  usecase "can view restaurant by ranking" as UC1
  usecase "can give restaurant's menu reviews" as UC2
  usecase "can add a resutaurant" as UC3
  usecase "can view reviews" as UC4
  usecase "can search restaurant" as UC5
}
user --> UC1
user --> UC2
user --> UC3
user --> UC4
user --> UC5
UC3 .> UC2 : <<extends>>
@enduml
```

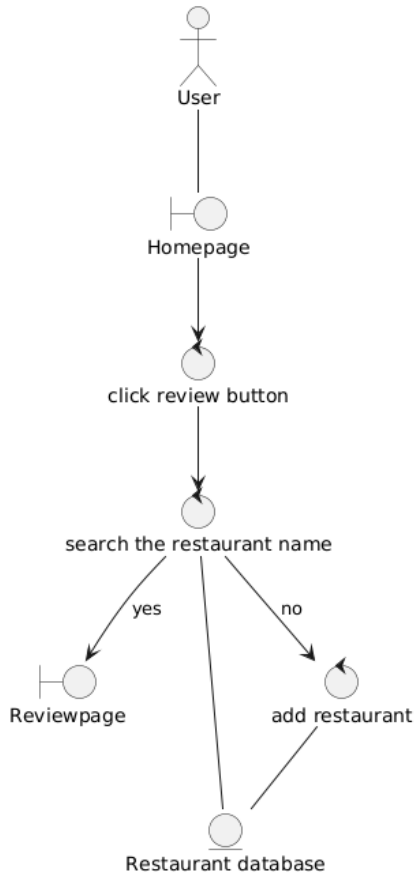
Robustness diagrams

Review

- Basic course:
    - User click review button
    - User search the restaurant name
    - Search restaurant name in the database whether exist or not
    - if yes, shows reviews form
    - User input review informa
    - tionUser submit review
    - Check if review include all required information
    - If yes, review information save in the database
    -
- Alternate course 1:
    - if cannot find restaurant name in the database
    - User add restaurant
- Alternate course 2:
    - If user does not input all the required information
    - Shows error message

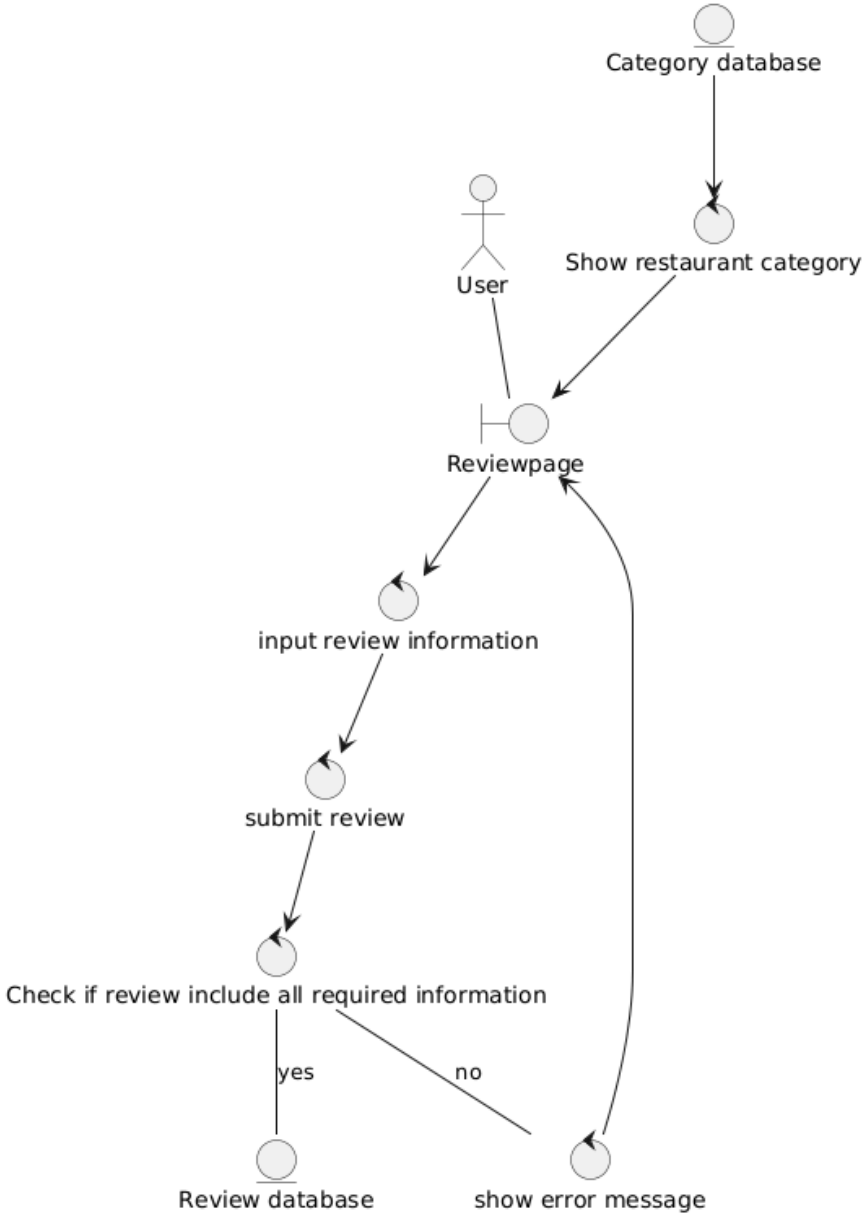
Review menu usecase:

1. search the restaurant




```
@startuml

actor "User" as user
boundary Homepage
boundary Reviewpage

control "click review button" as clikButton
control "search the restaurant name" as searchRestaurant
entity "Restaurant database" as restaurantDatabase
control "add restaurant" as addRestaurant

user -- Homepage
Homepage --> clikButton
clikButton --> searchRestaurant
searchRestaurant -- restaurantDatabase
searchRestaurant --> Reviewpage : yes
searchRestaurant --> addRestaurant : no
addRestaurant -- restaurantDatabase

@enduml
```

2. submit review



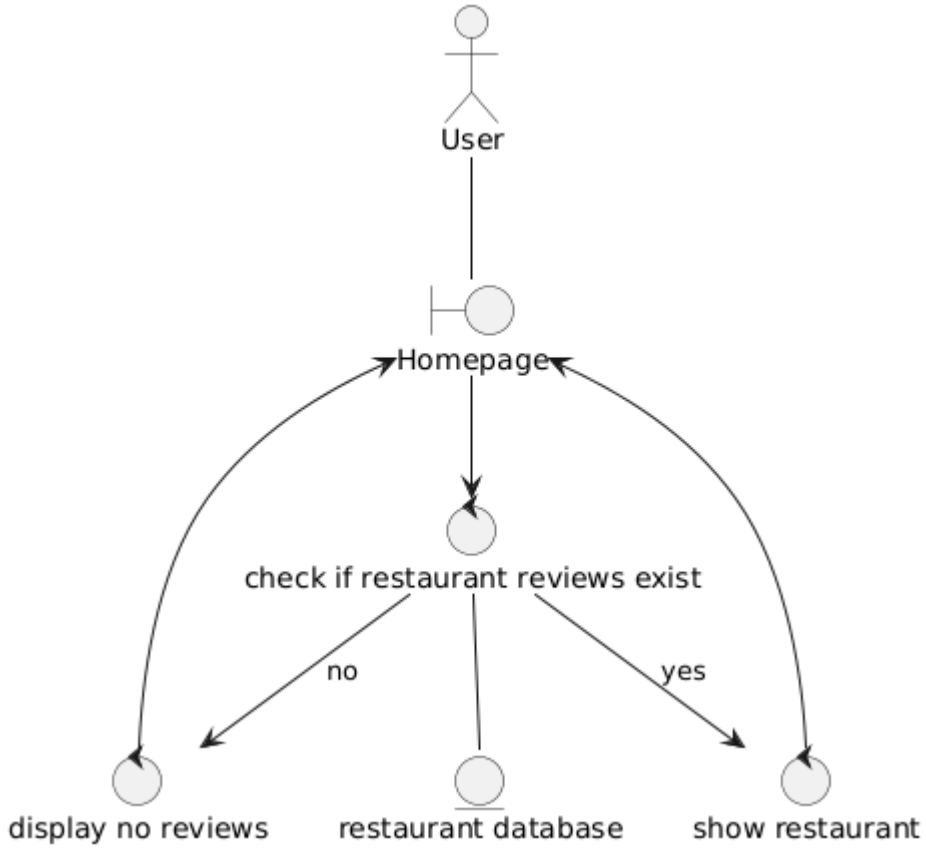
@startuml

actor "User" as user
boundary Reviewpage

control "input review information" as InputReveiw
control "submit review" as SubmitReview
control "Check if review include all required information" as check
control "Show restaurant category" as showCategory
entity "Review database" as reviewDatabase

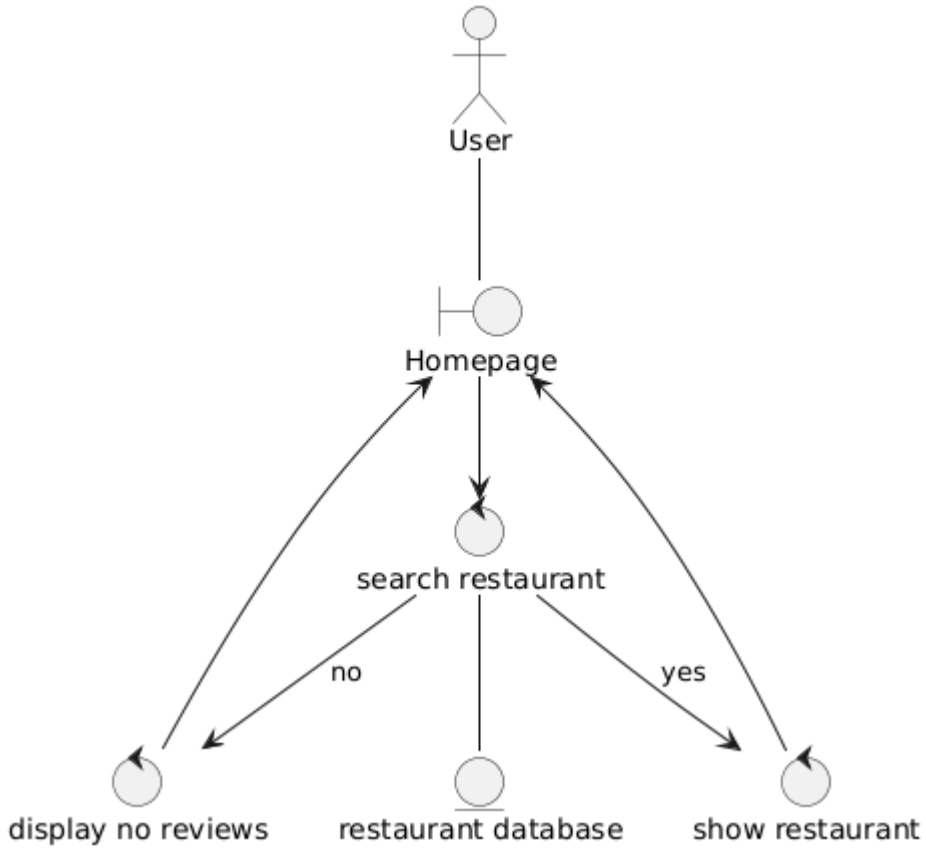
```
entity "Category database" as categoryDatabase
control "show error message" as showError

user -- Reviewpage
Reviewpage --> InputReveiw
InputReveiw --> SubmitReview
SubmitReview --> check
check -- reviewDatabase : yes
check -- showError : no
showError --> Reviewpage
categoryDatabase --> showCategory
showCategory --> Reviewpage
@enduml
```

View restaurants ranking:



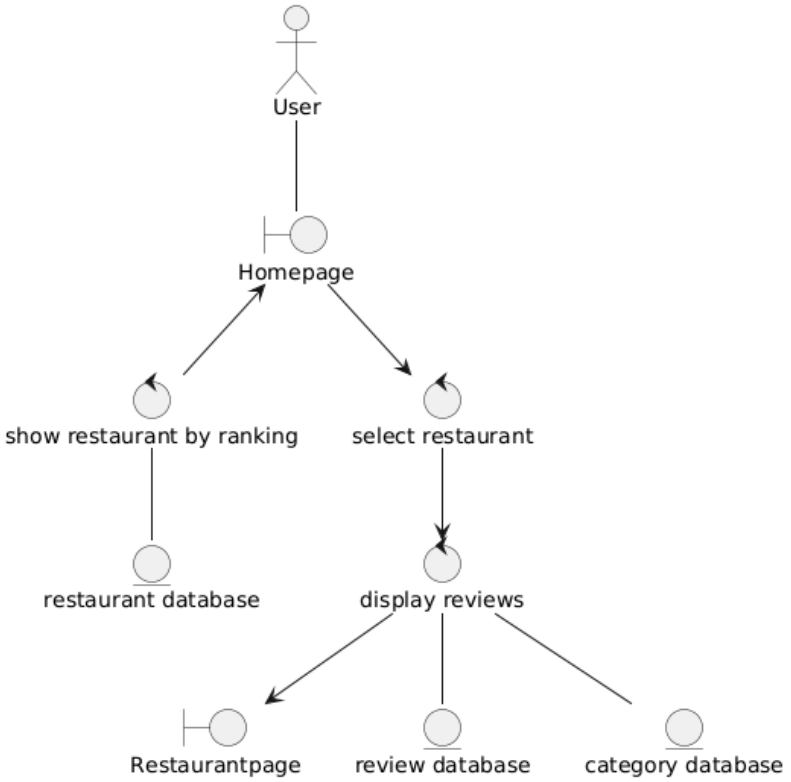

```
@startuml

actor "User" as user
boundary Homepage

control "check if restaurant reviews exist" as c1
entity "restaurant database" as restaurantDatabase
control "show restaurant" as c2
control "display no reviews" as c3

user -- Homepage
Homepage --> c1
c1 -- restaurantDatabase
c1 --> c2 : yes
c1 --> c3 : no
c2 --> Homepage
c3 --> Homepage

@enduml
```

Search restaurants on Home page:




```
@startuml

actor "User" as user
boundary Homepage

control "search restaurant" as c1
entity "restaurant database" as restaurantDatabase
control "show restaurant" as c2
control "display no reviews" as c3

user -- Homepage
Homepage --> c1
c1 -- restaurantDatabase
c1 --> c2 : yes
c1 --> c3 : no
c2 --> Homepage
c3 --> Homepage

@enduml
```

View reviews




```
@startuml

actor "User" as user
boundary Homepage

control "show restaurant by ranking" as c1
control "select restaurant" as c2
boundary Restaurantpage
control "display reviews" as c3
entity "restaurant database" as restaurantDatabase
entity "review database" as reviewDatabase
entity "category database" as categoryDatabase

user -- Homepage
Homepage <-- c1
c1 -- restaurantDatabase
Homepage --> c2
c2 --> c3
c3 --> Restaurantpage
c3 -- reviewDatabase
c3 -- categoryDatabase

@enduml
```