ARE YOU HUNGRY? JUST SEARCH

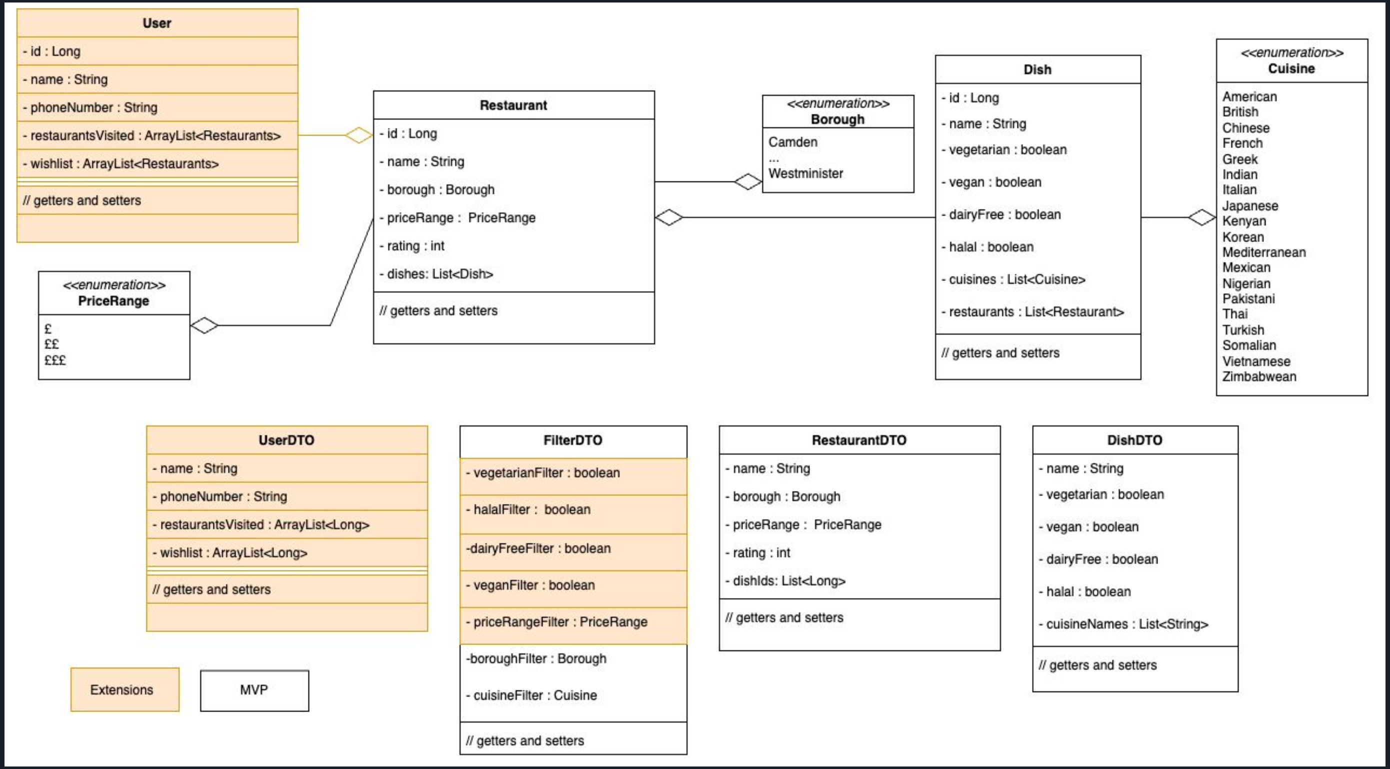
Anna’s Restaurant Recommender By Team Bad WiFi

AGENDA

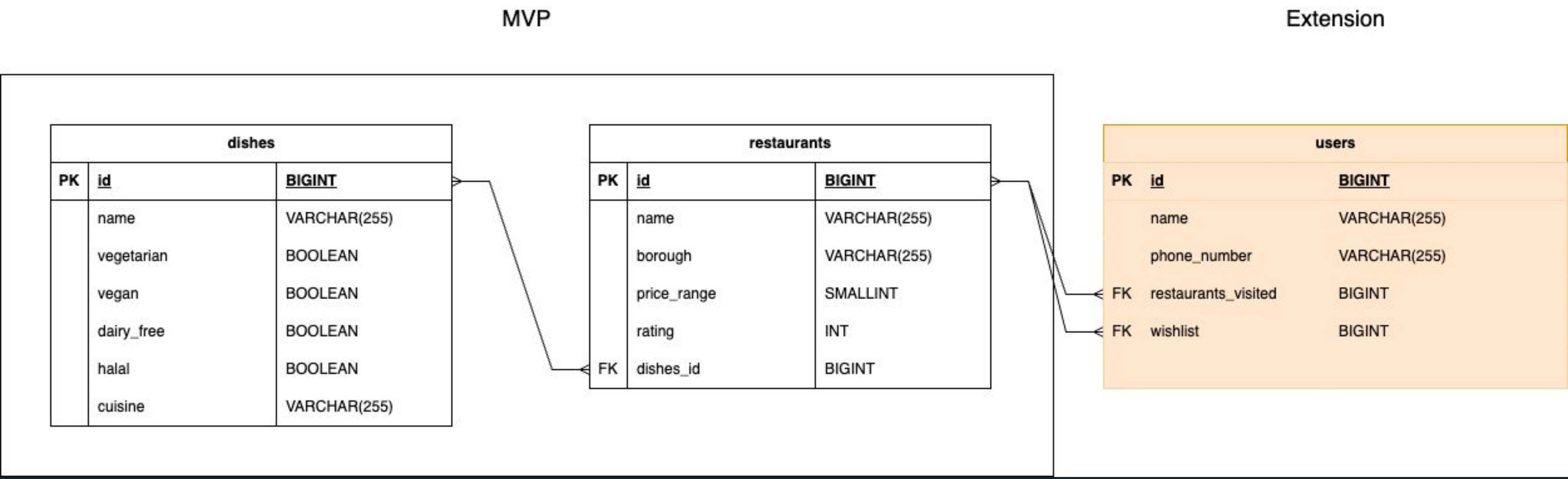
- An introduction to the theme and motivations behind it, followed by a discussion of the planning - Amelie
- A discussion through the code - Yihang
- Followed by a demo through the program - Faran
- An insight into bugs and problems we encountered - Callum
- To conclude, we will discuss the groups achievements and reflections - Sandra

MOTIVATIONS BEHIND OUR PROJECT

Class Diagram



Entity Relationship Diagram



dishes

PK	id	BIGINT
	name	VARCHAR(255)
	vegetarian	BOOLEAN
	vegan	BOOLEAN
	dairy_free	BOOLEAN
	halal	BOOLEAN
	cuisine	VARCHAR(255)

restaurants

PK	id	BIGINT
	name	VARCHAR(255)
	borough	VARCHAR(255)
	price_range	SMALLINT
	rating	INT
FK	dishes_id	BIGINT

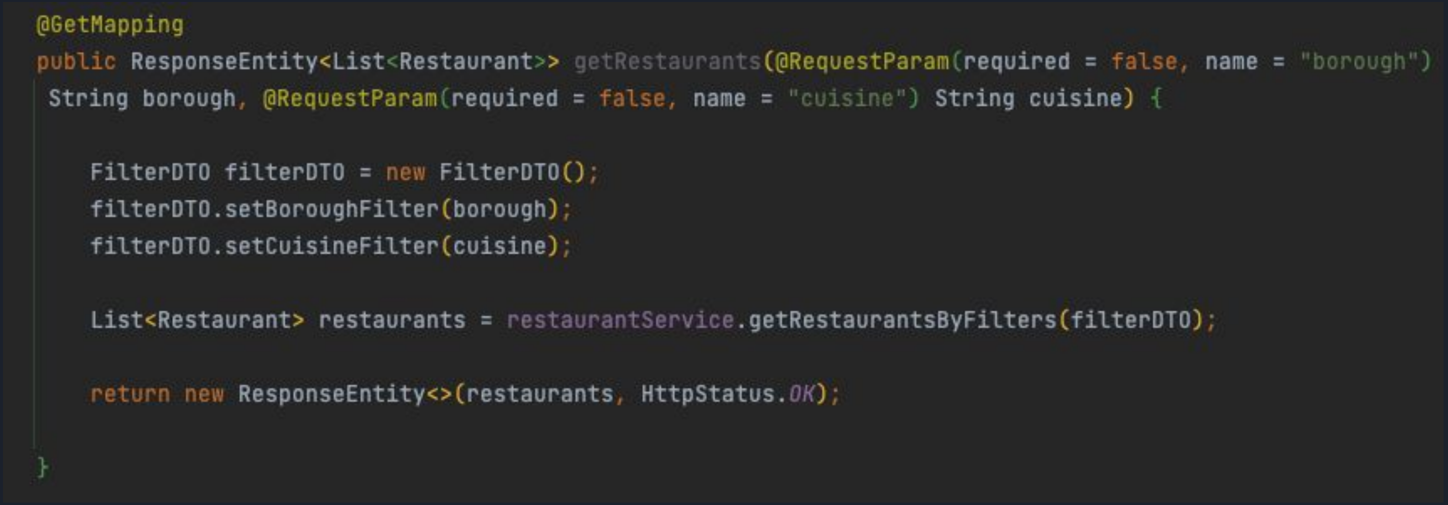
users

PK	id	BIGINT
	name	VARCHAR(255)
	phone_number	VARCHAR(255)
FK	restaurants_visited	BIGINT
FK	wishlist	BIGINT

CODE SAMPLE

Controller

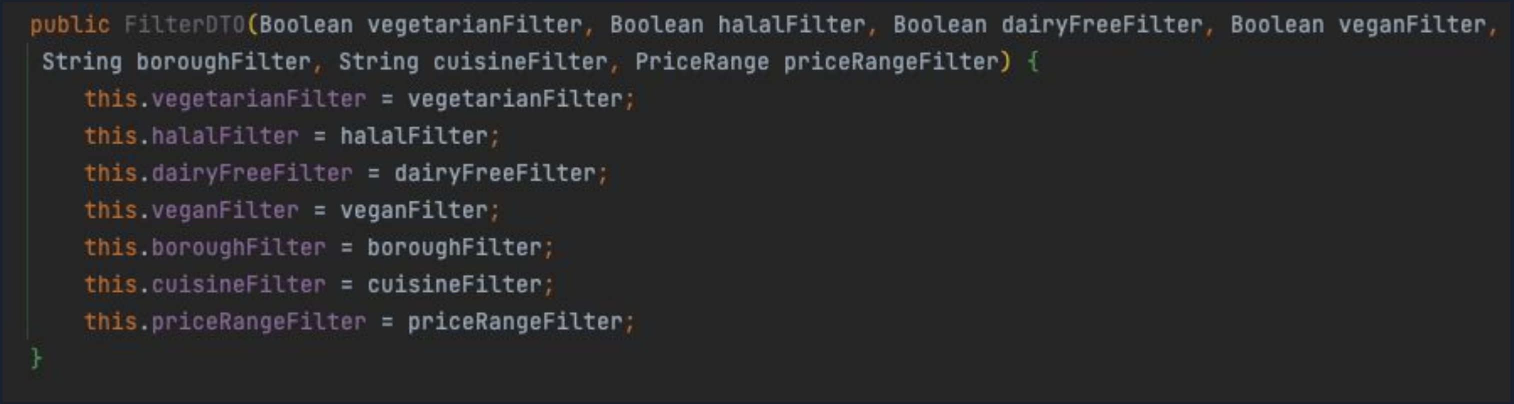
- GET restaurants endpoint with optional filter request parameters
- Response consists of corresponding restaurants
- DTO to help pass information from controller to service layer



```java
@GetMapping
public ResponseEntity<List<Restaurant>> getRestaurants(@RequestParam(required = false, name = "borough")
 String borough, @RequestParam(required = false, name = "cuisine") String cuisine) {

    FilterDTO filterDTO = new FilterDTO();
    filterDTO.setBoroughFilter(borough);
    filterDTO.setCuisineFilter(cuisine);

    List<Restaurant> restaurants = restaurantService.getRestaurantsByFilters(filterDTO);

    return new ResponseEntity<>(restaurants, HttpStatus.OK);

}
```

DTO

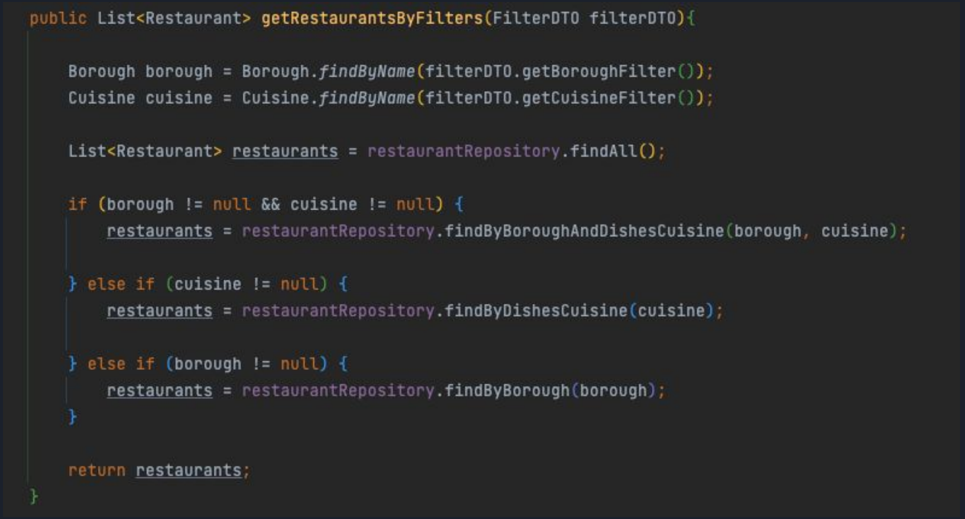
- Better understanding of DTOs
- Transfer of data from controller to service
- Additional filters for extension



```
public FilterDTO(Boolean vegetarianFilter, Boolean halalFilter, Boolean dairyFreeFilter, Boolean veganFilter,
 String boroughFilter, String cuisineFilter, PriceRange priceRangeFilter) {
    this.vegetarianFilter = vegetarianFilter;
    this.halalFilter = halalFilter;
    this.dairyFreeFilter = dairyFreeFilter;
    this.veganFilter = veganFilter;
    this.boroughFilter = boroughFilter;
    this.cuisineFilter = cuisineFilter;
    this.priceRangeFilter = priceRangeFilter;
}
```

Service

- Custom enum methods to find corresponding values from strings
- Conditionals to check presence of filters
- Makes use of derived queries



```java
public List<Restaurant> getRestaurantsByFilters(FilterDTO filterDTO){

    Borough borough = Borough.findByName(filterDTO.getBoroughFilter());
    Cuisine cuisine = Cuisine.findByName(filterDTO.getCuisineFilter());

    List<Restaurant> restaurants = restaurantRepository.findAll();

    if (borough != null && cuisine != null) {
        restaurants = restaurantRepository.findByBoroughAndDishesCuisine(borough, cuisine);

    } else if (cuisine != null) {
        restaurants = restaurantRepository.findByDishesCuisine(cuisine);

    } else if (borough != null) {
        restaurants = restaurantRepository.findByBorough(borough);
    }

    return restaurants;
}
```

DEMO

ENDPOINTS

INDEX

- `GET localhost:8080/restaurants`
- `GET localhost:8080/restaurants?borough={boroughName}`
- `GET localhost:8080/restaurants?cuisine={cuisineName}`
- `GET localhost:8080/restaurants?cuisine={cuisineName}&borough={boroughName}`
- `GET localhost:8080/dishes`

SHOW

- `GET localhost:8080/restaurants/{id}`
- `GET localhost:8080/dishes/{id}`

CREATE

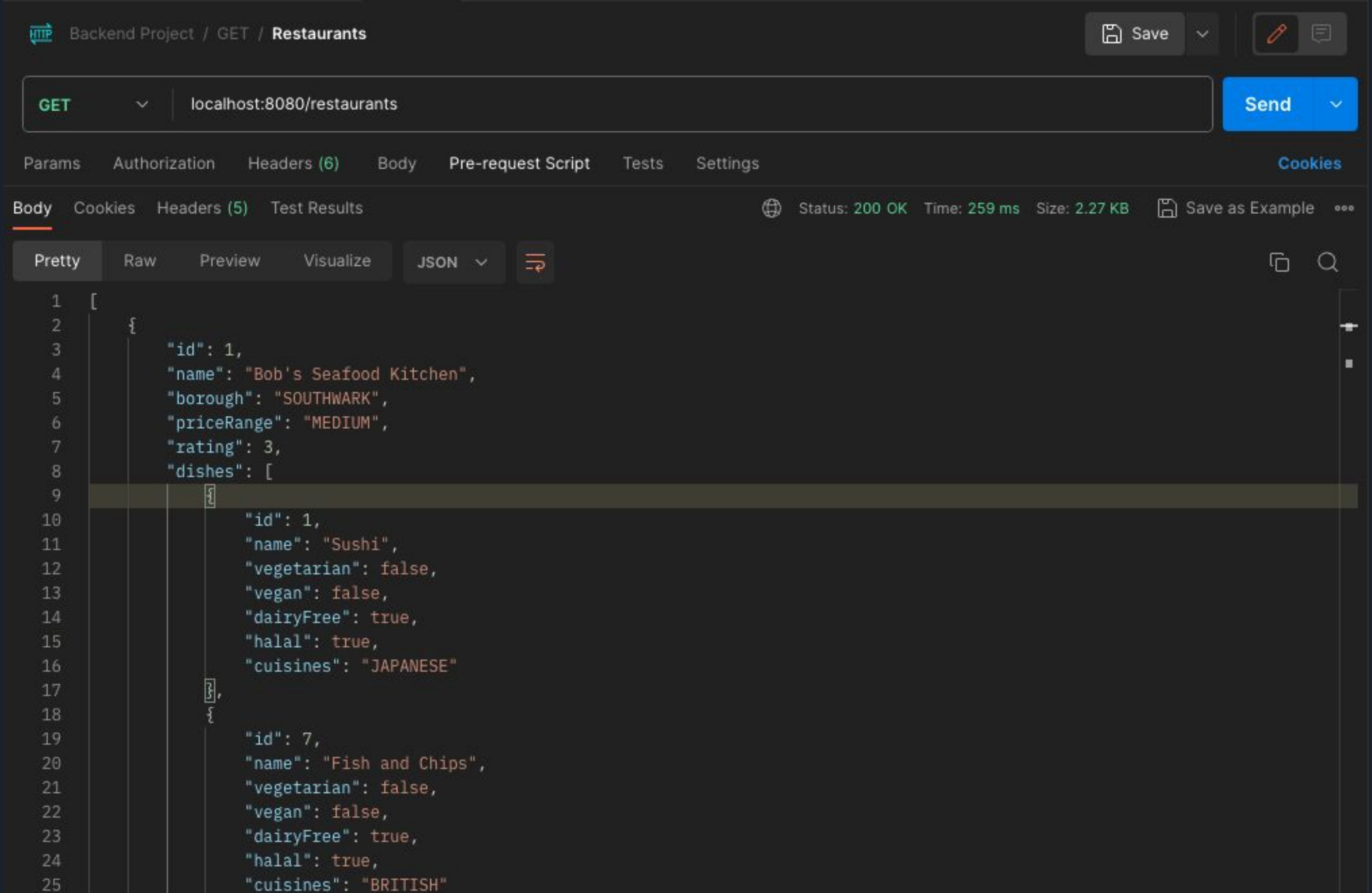
- `POST localhost:8080/restaurants`
- `POST localhost:8080/dishes`

UPDATE

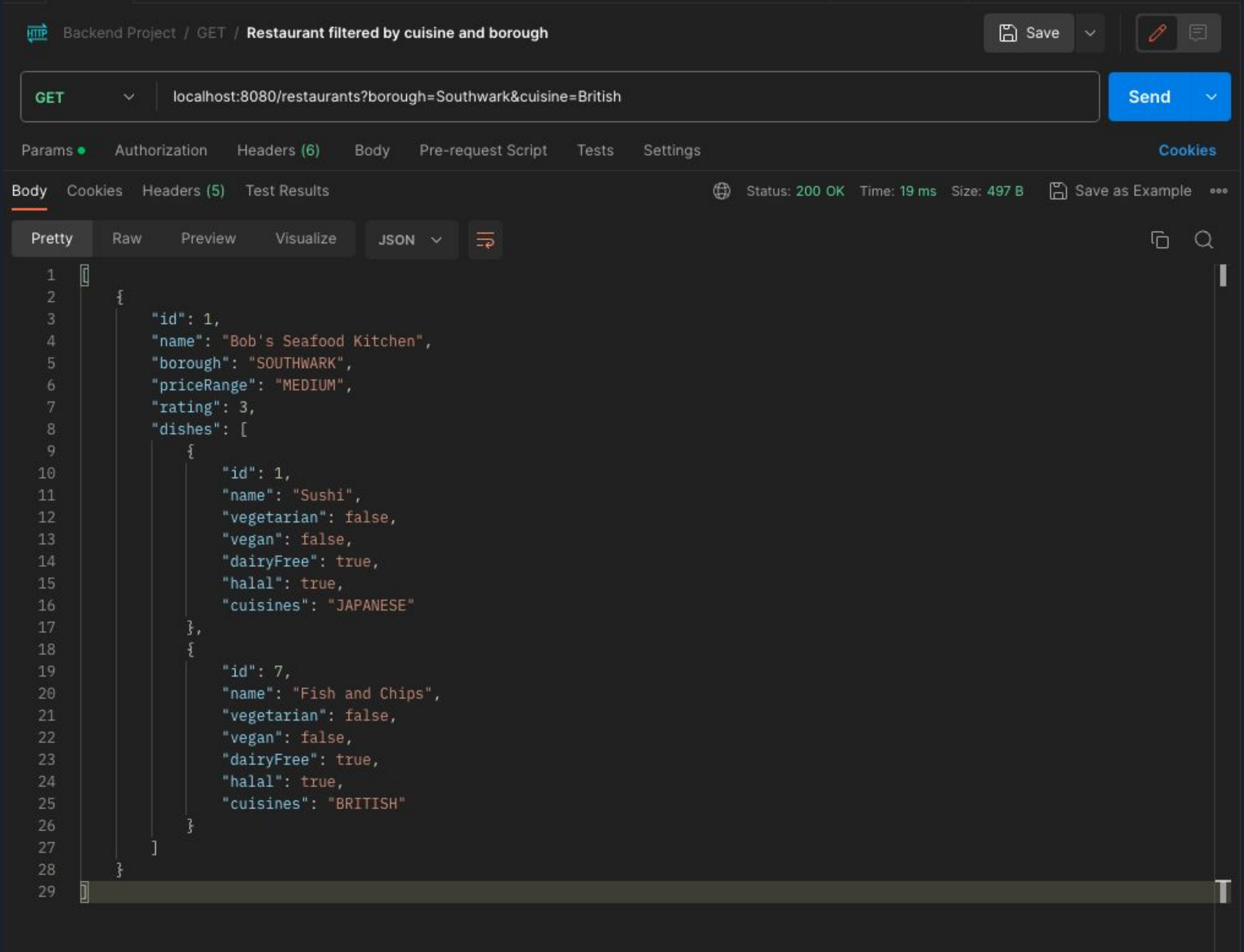
- `PUT localhost:8080/restaurants/{id}`

DESTROY

- `DELETE localhost:8080/restaurants/{id}`
- `DELETE localhost:8080/dishes/{id}`



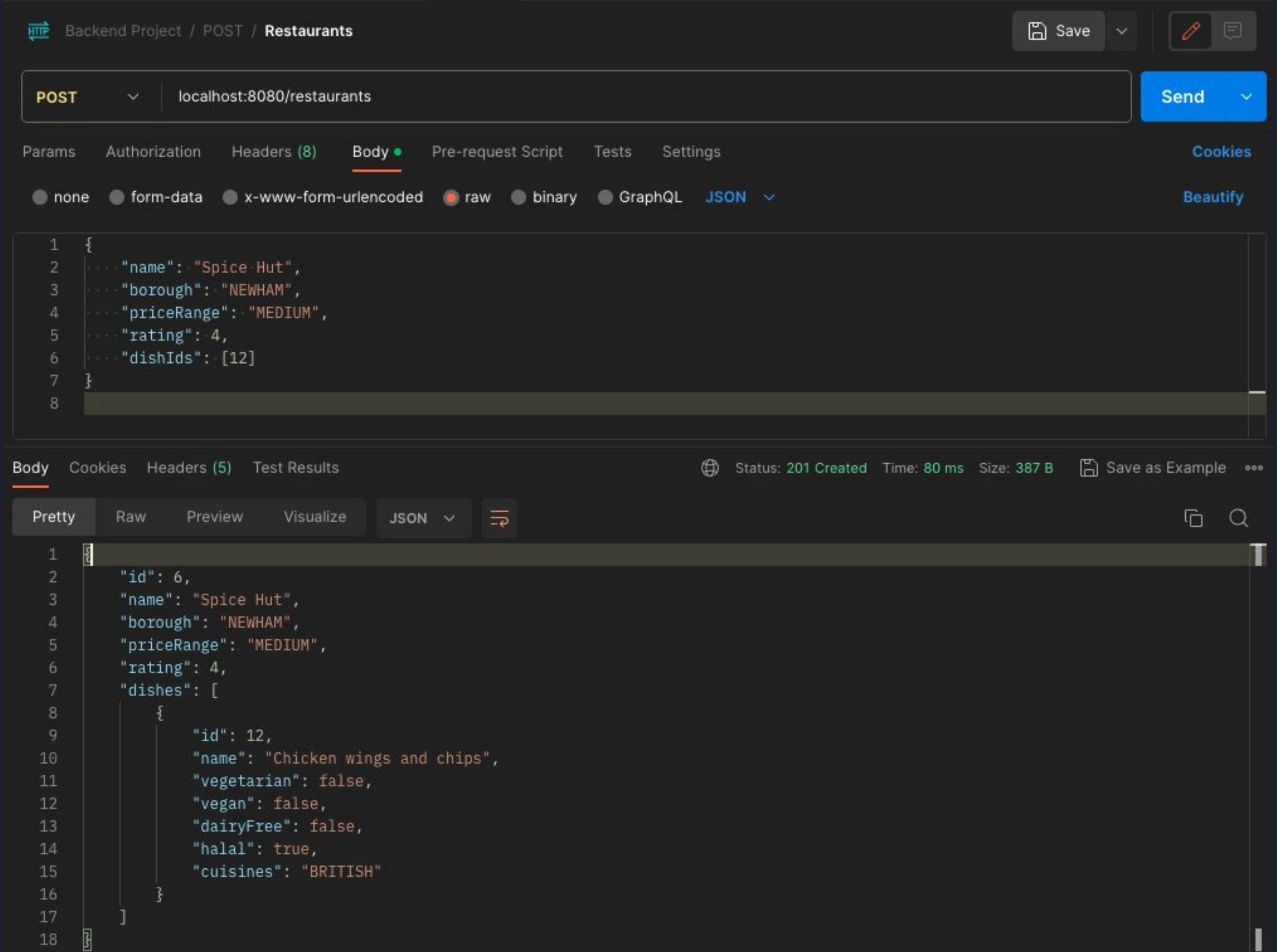
```
[
    {
        "id": 1,
        "name": "Bob's Seafood Kitchen",
        "borough": "SOUTHWARK",
        "priceRange": "MEDIUM",
        "rating": 3,
        "dishes": [
            {
                "id": 1,
                "name": "Sushi",
                "vegetarian": false,
                "vegan": false,
                "dairyFree": true,
                "halal": true,
                "cuisines": "JAPANESE"
            },
            {
                "id": 7,
                "name": "Fish and Chips",
                "vegetarian": false,
                "vegan": false,
                "dairyFree": true,
                "halal": true,
                "cuisines": "BRITISH"
```



localhost:8080/restaurants?borough=Southwark&cuisine=British

Status: 200 OK Time: 19 ms Size: 497 B

```
[
    {
        "id": 1,
        "name": "Bob's Seafood Kitchen",
        "borough": "SOUTHWARK",
        "priceRange": "MEDIUM",
        "rating": 3,
        "dishes": [
            {
                "id": 1,
                "name": "Sushi",
                "vegetarian": false,
                "vegan": false,
                "dairyFree": true,
                "halal": true,
                "cuisines": "JAPANESE"
            },
            {
                "id": 7,
                "name": "Fish and Chips",
                "vegetarian": false,
                "vegan": false,
                "dairyFree": true,
                "halal": true,
                "cuisines": "BRITISH"
            }
        ]
    }
]
```



POST localhost:8080/restaurants

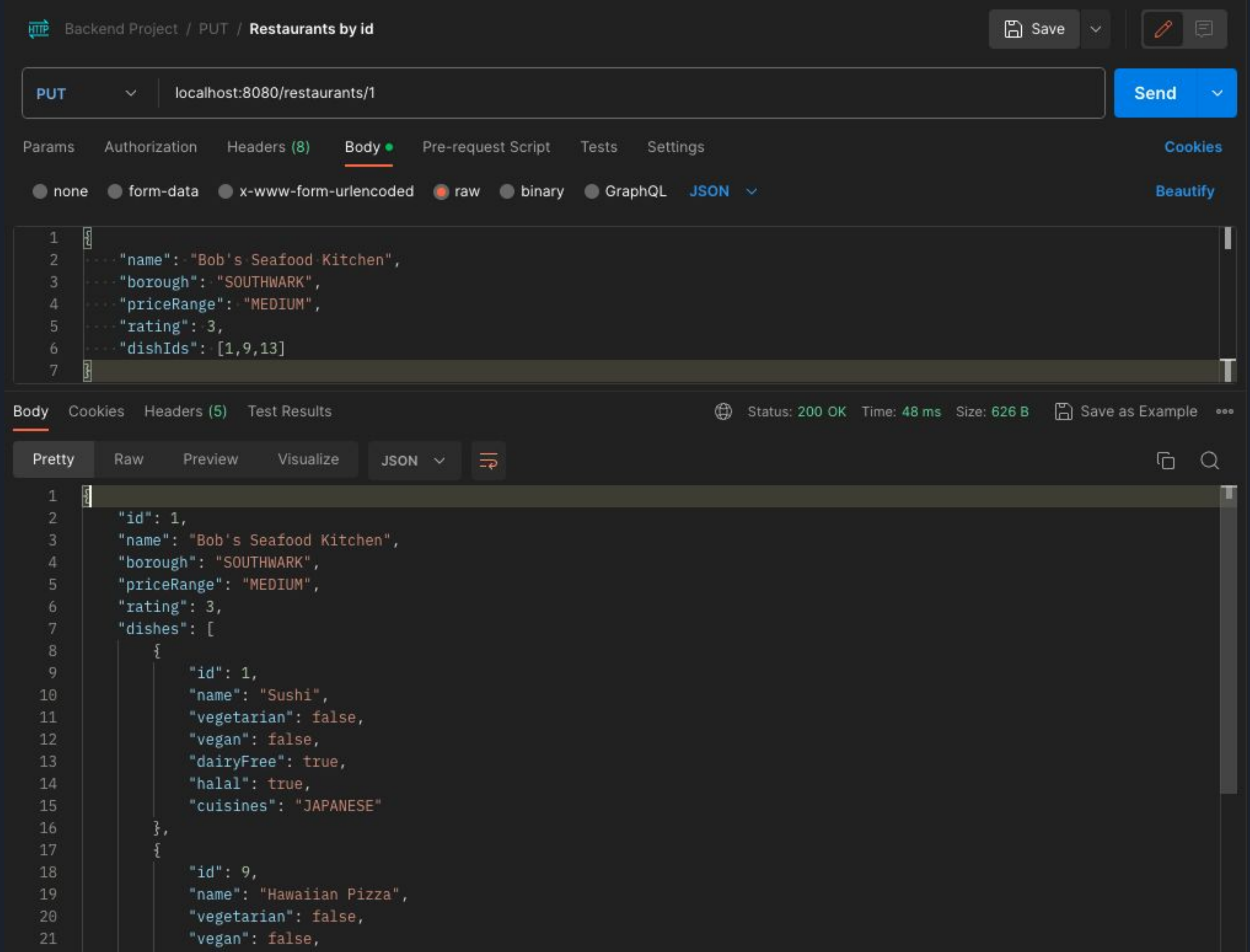
Body: raw [✓ raw] JSON

```
{
    "name": "Spice Hut",
    "borough": "NEWHAM",
    "priceRange": "MEDIUM",
    "rating": 4,
    "dishIds": [12]
}
```

Status: 201 Created Time: 80 ms Size: 387 B

JSON

```
{
    "id": 6,
    "name": "Spice Hut",
    "borough": "NEWHAM",
    "priceRange": "MEDIUM",
    "rating": 4,
    "dishes": [
        {
            "id": 12,
            "name": "Chicken wings and chips",
            "vegetarian": false,
            "vegan": false,
            "dairyFree": false,
            "halal": true,
            "cuisines": "BRITISH"
        }
    ]
}
```



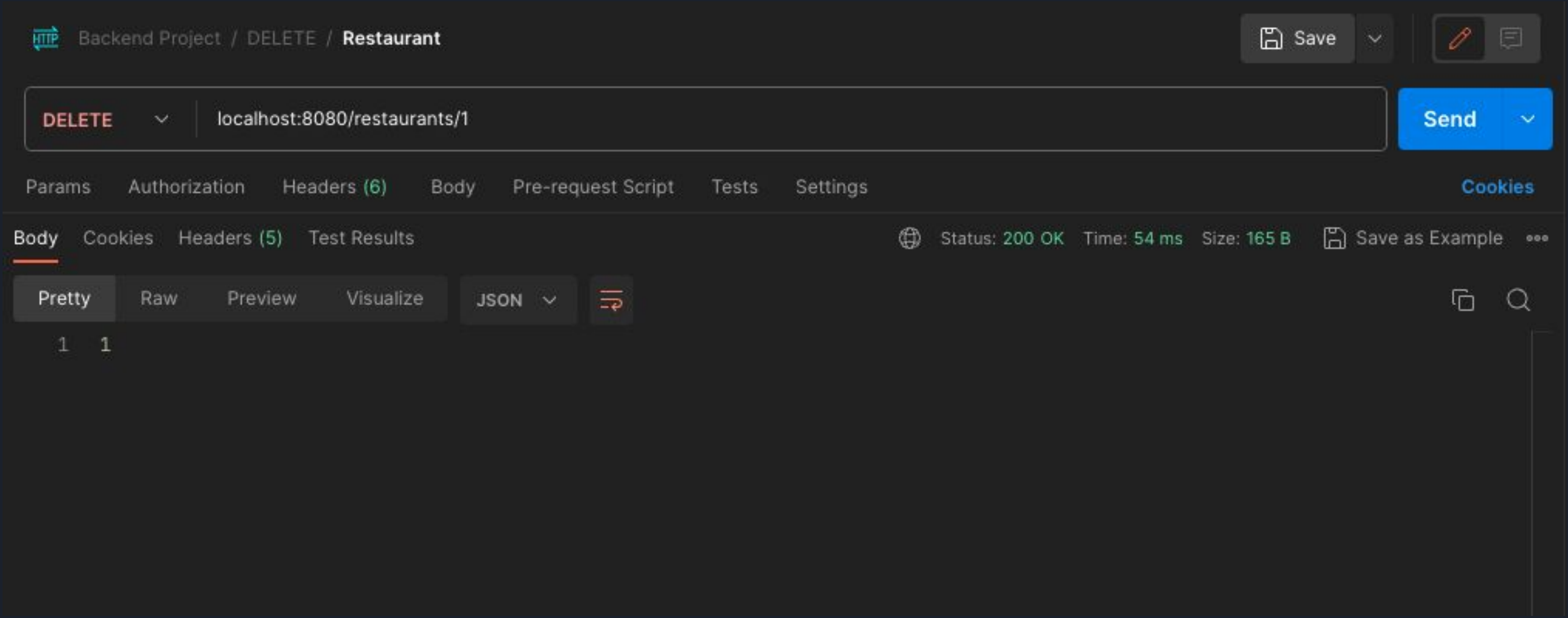
PUT localhost:8080/restaurants/1

Body: [✓ raw] JSON

```
{
    "name": "Bob's Seafood Kitchen",
    "borough": "SOUTHWARK",
    "priceRange": "MEDIUM",
    "rating": 3,
    "dishIds": [1,9,13]
}
```

Status: 200 OK Time: 48 ms Size: 626 B

```
{
    "id": 1,
    "name": "Bob's Seafood Kitchen",
    "borough": "SOUTHWARK",
    "priceRange": "MEDIUM",
    "rating": 3,
    "dishes": [
        {
            "id": 1,
            "name": "Sushi",
            "vegetarian": false,
            "vegan": false,
            "dairyFree": true,
            "halal": true,
            "cuisines": "JAPANESE"
        },
        {
            "id": 9,
            "name": "Hawaiian Pizza",
            "vegetarian": false,
            "vegan": false,
```


Backend Project / DELETE / **Restaurant**

Save

DELETE localhost:8080/restaurants/1

Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (5) Test Results Status: 200 OK Time: 54 ms Size: 165 B Save as Example

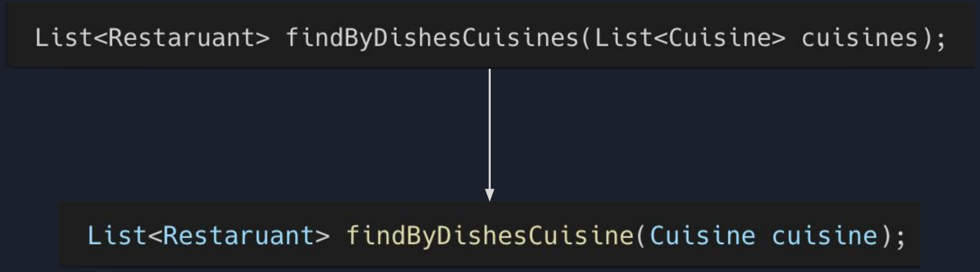
Pretty Raw Preview Visualize JSON

```
1
```

BUGS AND ISSUES

Querying from repository scalar argument error

- When starting the application ran into scalar argument error with our findByCuisine query
- Scalar argument requires a single value to be passed into the function
- Changed the definition of the dish to only have one cuisine instead of a list
- Refactored code and fixed the error



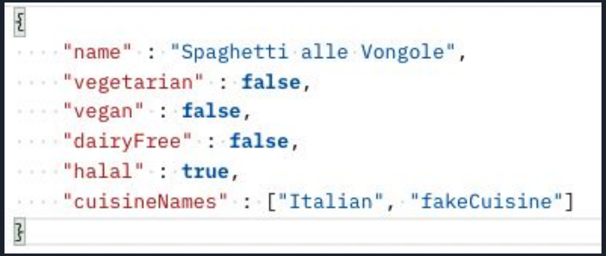
```
List<Restaruant> findByDishesCuisines(List<Cuisine> cuisines);
```

```
List<Restaruant> findByDishesCuisine(Cuisine cuisine);
```

Creating dish with null value for cuisine

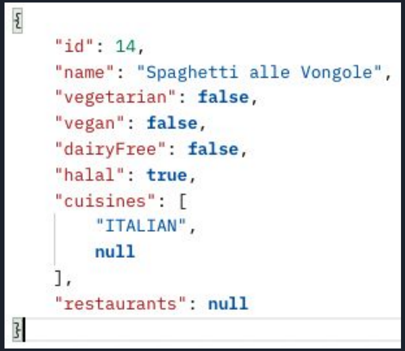
Creating dish object if cuisine did not exist in enum

- POST localhost:8080/dishes
- Request Body



```
{
    "name" : "Spaghetti alle Vongole",
    "vegetarian" : false,
    "vegan" : false,
    "dairyFree" : false,
    "halal" : true,
    "cuisineNames" : ["Italian", "fakeCuisine"]
}
```

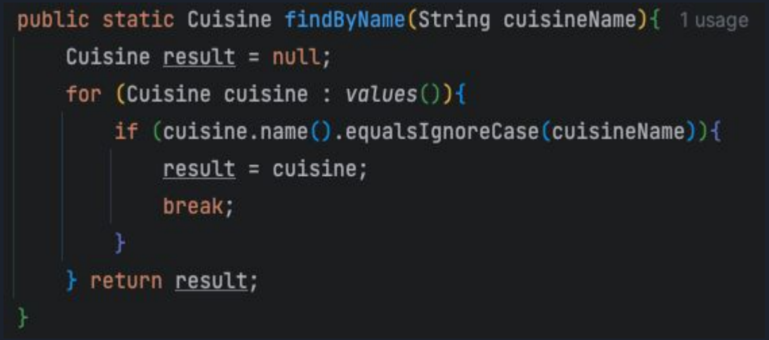
- Response Body



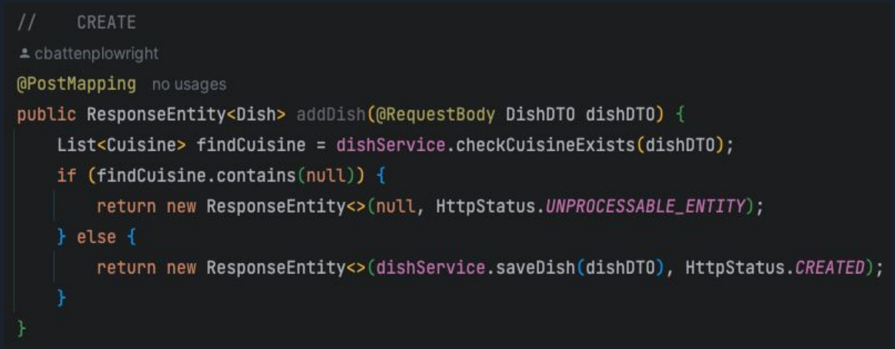
```
{
    "id": 14,
    "name": "Spaghetti alle Vongole",
    "vegetarian": false,
    "vegan": false,
    "dairyFree": false,
    "halal": true,
    "cuisines": [
        "ITALIAN",
        null
    ],
    "restaurants": null
}
```

Creating dish with null value for cuisine

- Added a validation method called findByName in the Cuisine Enum
- Added condition checking the returned value then returns the correct ResponseEntity



```java
public static Cuisine findByName(String cuisineName){  1 usage
    Cuisine result = null;
    for (Cuisine cuisine : values()){
        if (cuisine.name().equalsIgnoreCase(cuisineName)){
            result = cuisine;
            break;
        }
    } return result;
}
```



```java
//    CREATE
cbattenplowright
@PostMapping  no usages
public ResponseEntity<Dish> addDish(@RequestBody DishDTO dishDTO) {
    List<Cuisine> findCuisine = dishService.checkCuisineExists(dishDTO);
    if (findCuisine.contains(null)) {
        return new ResponseEntity<>(null, HttpStatus.UNPROCESSABLE_ENTITY);
    } else {
        return new ResponseEntity<>(dishService.saveDish(dishDTO), HttpStatus.CREATED);
    }
}
```

GROUP ACHIEVEMENTS

What we are particularly proud of?

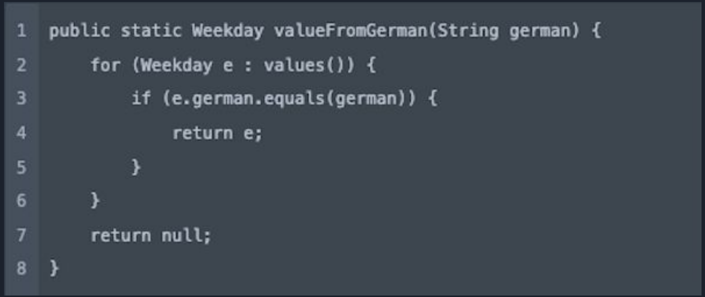
git push origin branch-name

Team Achievements

- Agenda establishment
- Commitment
- Team Support
- MVP achieved
- Functional model

Overcoming Issues

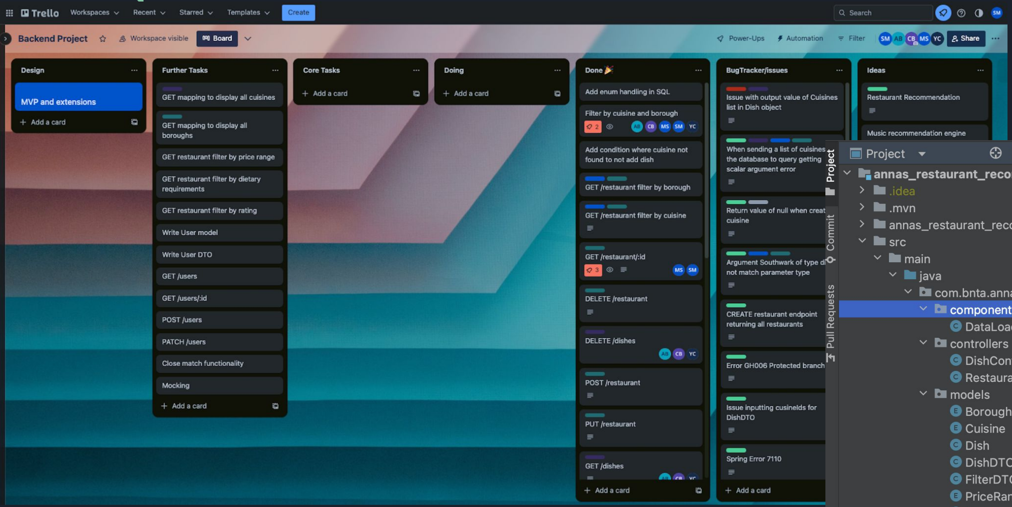
- Branch conflicts
- Finding items in the ENUM files by their values implementing a static method
- Filter restaurants by more than one parameter using a DTO



```java
public static Weekday valueFromGerman(String german) {
    for (Weekday e : values()) {
        if (e.german.equals(german)) {
            return e;
        }
    }
    return null;
}
```

JUNIOR DEVELOPER IN THE MORNING
WAITING FOR THE TEAM TO TELL THEM "I COMMITTED ALL MY WORK TO MAIN"

Collaborative Management



Design
- MVP and extensions
- + Add a card

Further Tasks
- GET mapping to display all cuisines
- GET mapping to display all boroughs
- GET restaurant filter by price range
- GET restaurant filter by dietary requirements
- GET restaurant filter by rating
- Write User model
- Write User DTO
- GET /users
- GET /users/:id
- POST /users
- PATCH /users
- Close match functionality
- Mocking
- + Add a card

Core Tasks
- + Add a card

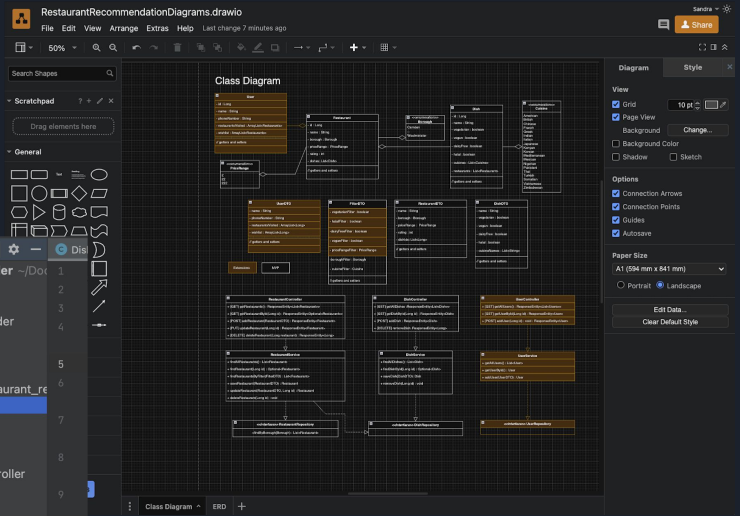
Doing
- + Add a card

Done 🎉
- Add enum handling in SQL
- Filter by cuisine and borough
- Add condition where cuisine not found to not add dish
- GET /restaurant filter by borough
- GET /restaurant filter by cuisine
- GET /restaurant/:id
- DELETE /restaurant
- DELETE /dishes
- POST /restaurant
- PUT /restaurant
- GET /dishes
- + Add a card

BugTracker/Issues
- Issue with output value of Cuisines list in Dish object
- When sending a list of cuisines to the database to query getting scalar argument error
- Return value of null when creat cuisine
- Argument Southwark of type d not match parameter type
- CREATE restaurant endpoint returning all restaurants
- Error GH006 Protected branch
- Issue inputting cuisineIds for DishDTO
- Spring Error 7110
- + Add a card

Ideas
- Restaurant Recommendation
- Music recommendation engine

Trello

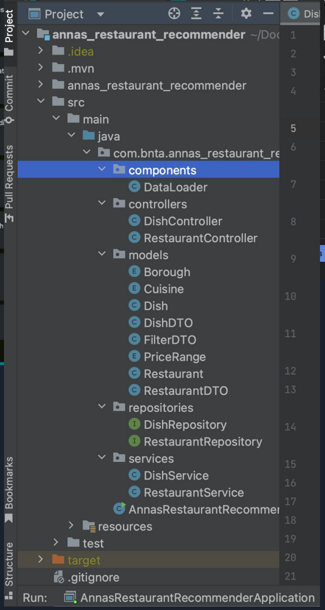


View
- [✓] Grid 10 pt
- [✓] Page View
- Background Change...
- [ ] Background Color
- [ ] Shadow
- [ ] Sketch

Options
- [✓] Connection Arrows
- [✓] Connection Points
- [✓] Guides
- [✓] Autosave

Paper Size
- A1 (594 mm x 841 mm)
- Portrait [✓ Landscape]
- Edit Data...
- Clear Default Style

Google drawio



```
Project
annas_restaurant_recommender ~/Doc
  .idea
  .mvn
  annas_restaurant_recommender
  src
    main
      java
        com.bnta.annas_restaurant_re
          components
            DataLoader
          controllers
            DishController
            RestaurantController
          models
            Borough
            Cuisine
            Dish
            DishDTO
            FilterDTO
            PriceRange
            Restaurant
            RestaurantDTO
          repositories
            DishRepository
            RestaurantRepository
          services
            DishService
            RestaurantService
          AnnasRestaurantRecommen
      resources
    test
  target
  .gitignore
Run: AnnasRestaurantRecommenderApplication
```

IntelliJ

Reflections

Five heads are better than one

Coding alone is not sustainable nor efficient

A basic functional model is better than a non-running model with fancy features

Get better WiFi. WiFi is essential

"It's not about the results, it's about the friends we make along the way"