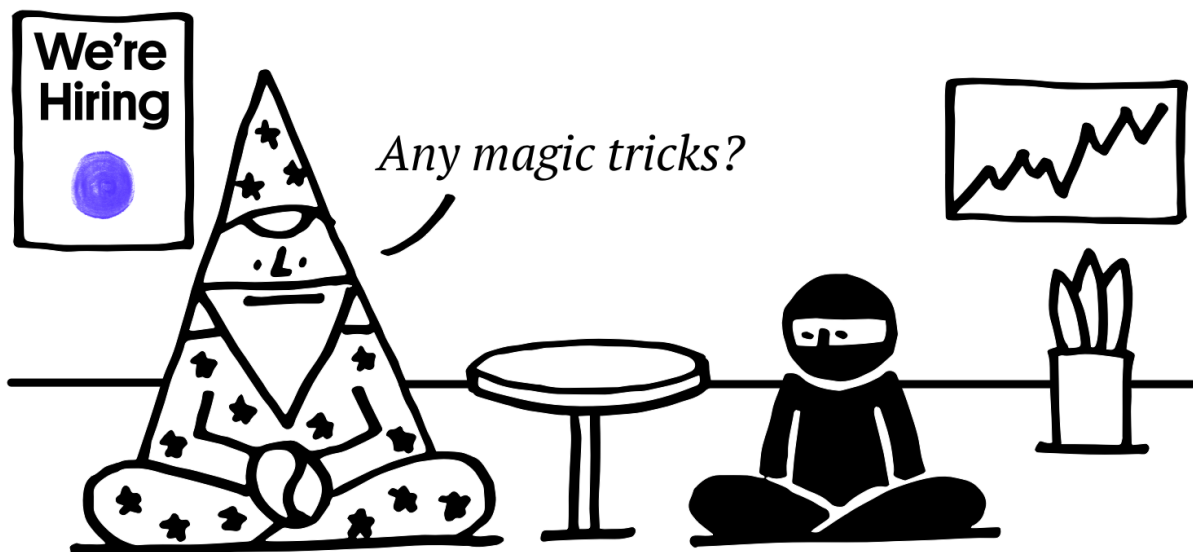




Welcome, Peter.

Backend Developer Test Assignment at d.labs



Dear Peter,

At our first interview, you made quite an impression on us, which is why we want you to solve an assignment with us so we can get to know each other.

We hope you'll enjoy the challenge called **"The Restaurant at the End of the Universe"**!

Think of what is ahead of you less as an assignment and more as **a job simulation** — one that is helpful both for you to see whether the developer role at d.labs is what you're after and for us to see whether you are a good fit.

Your job is to implement an MVP (minimum viable product) application. Since we want to keep your hands open, we have not provided a lot of details. For example, you'll probably notice that we have not provided you with any visual design on how the application should look. Below, you'll more or less only find the basic description of necessary functionalities.

At the same time, we're allowing you to make shortcuts as you see necessary, as long as you can explain them and you have an idea how you'd implement them in a real world scenario. For example, as mentioned below, you're not actually required to make a full user registration and login process. You can take other shortcuts as you see fit, as long as the final application serves its purpose.

At the same time, we'd of course like to see you demonstrate your knowledge of principles and system design. In this light, balancing shortcuts and demonstrating knowledge might prove to be tricky. Try to keep things simple and clean while covering common edge cases.

If anything is keeping you from moving forward, **feel free to nudge us anytime.**



Now, **take a breath**, put on your d.labs developer hat and **have fun!**



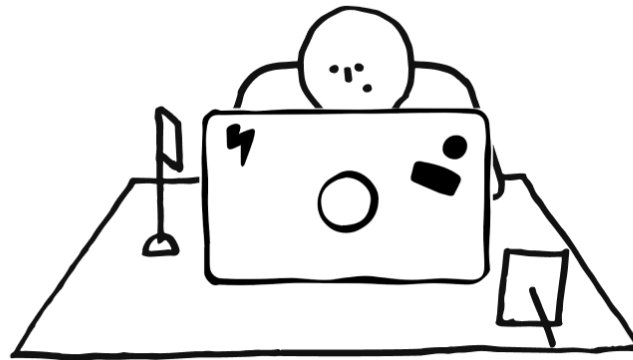
1 Intro: The Restaurant at the End of the Universe

You have just received a Slack message from Pete, our salesperson. An owner of “The Restaurant at the End of the Universe” wants to digitalise his business, and we’ve been chosen for the job. You can find a short brief below:

Located right at the end of the known universe, for the past two centuries, The Restaurant at the End of the Universe has been just another restaurant serving food of questionable quality and mostly occupied by a few groups of locals, who are regular guests.

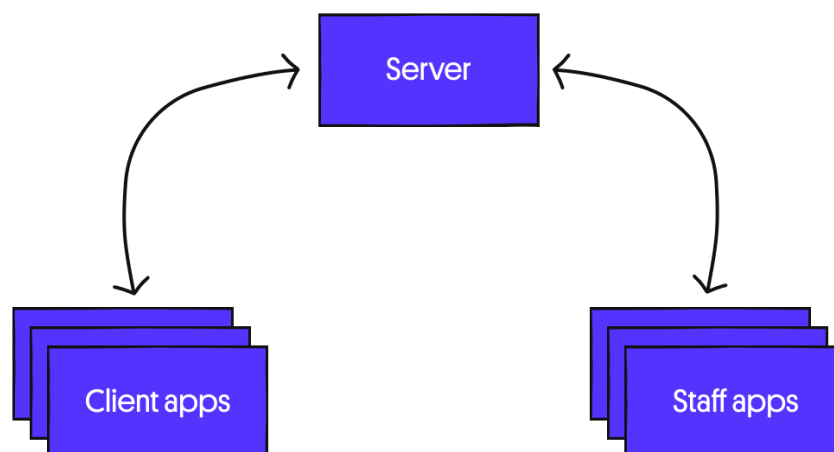
The whole situation changed when Zaphod Beeblebrox, the President of the Galaxy, made an unexpected stop at the planet to refuel his hyper-galactic vessel. As the refueling process took a while, he decided he’d try out the local gastronomy, and that’s how he ended up tasting grilled space-whale steak with spiced algae puree, the most disgusting meal he’s ever had in his life. Since he had to complain to somebody, he posted a picture of his meal to a social network called Galaxygram, where he has a few billion followers. As you might have guessed, now everybody wants to experience that terrible space-whale steak at The Restaurant at the End of the Universe, made by master chefs from the planet of Dentress.

This sudden influx of customers is becoming a great problem for the restaurant. Its waiting staff, people of Jatravartid from the planet called Viltvodle VI, was always praised for being nice and friendly, but with the increased traffic, they can get confused quickly, and they’re not a very organised bunch anyway.



2 Product development: The restaurant application

To make sure the restaurant will keep its clients happy, it wants to introduce an application which would let clients order their food and drinks, as well as assist the staff with preparing and serving the meal. Staff consist of waiters, barmans and chefs. Barmans prepare drinks, chefs prepare food, and waiters pick up food (and drinks) and deliver them to the client's table, as well as handle the bill payment.



A diagram representing the relation between applications

The assignment consists of three separate applications. For the purpose of this assignment, both frontend applications, guest and staff applications, will be provided. Your job is to construct a server which will connect together both frontend applications.

Think about the solution and the steps necessary to accomplish it. Estimate and prioritize your work accordingly. You're not required to implement a perfect solution, but demonstrate your knowledge and understanding of languages, tools, processes and communication.

If you need to visualize or diagram anything, you may use a tool of your choice (pen and paper are still remarkably useful for such tasks). Any technical documentation helps later down the project roadmap, and you'll be able to use it at the end of the day, when you'll be presenting your solution.

For source code versioning, we're asking you to use Git, as it is a standard tool at d.labs. If you want, you can start off with the default JavaScript boilerplate (<https://github.com/dlabs/restaurant-rateofu>) and fork it to your account. As the final solution should be represented as a pull request to the original repository, this might be the easiest way to do it.

2.1 The client application - ordering food and drinks

The process of ordering is the following: a client, or a group of them, get into the restaurant and sit at a table. At the table, there's a tablet, which lets clients pick their food and drinks. When chosen, clients complete the order, and now the staff can take over.

The client application is relatively simple. Clients can list food and drinks, add them to cart, set the quantity of specific items, and submit the order. While they wait for the order to be served, they can see the order summary on the screen. The list of available drinks and food the restaurant offers is the following:

| Name | Type | Price |
|---|-------|---------|
| Grilled space-whale steak with algae puree | Food | £ 66.50 |
| Tea substitute | Drink | £ 1.50 |
| Hagro biscuit | Food | £ 32.00 |
| Ameglian Major Cow casserole | Food | £ 55.75 |
| Pan Galactic Gargle Blaster | Drink | £ 5.50 |
| Janx Spirit | Drink | £ 7.00 |
| Tzjin-anthony-ks the Gagrakackan version of the gin and tonic. | Drink | £ 11.50 |

The guest application is already written, and will communicate with the backend server through a series of API calls, which are described in more detail in the Appendix.

2.2 The staff application

The staff has a different application, connected to the same backend services. First, a member of staff needs to be identified within the system by his name and role (waiter, barman, chef). After identifying with the system, they are available to work.

When they see a new order was created, a chef (or barman) starts preparing the food (or drinks). As soon as they start the task, they mark the order item as being processed. After they've finished preparing the item, an available waiter (chosen by the system) is notified about the item being ready to be served. The waiter serves the ordered item and marks it as served in the application.

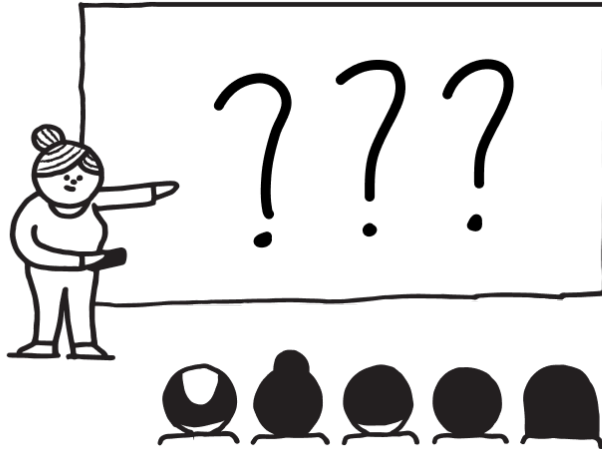
For the identification process, no proper authentication is necessary. The staff application's login screen consists of only two inputs - username and a role.


The staff application will communicate with the server with a series of API calls, which are described in more detail in the Appendix.

2.3 The server

The server is responsible for orchestrating the communication between client and staff applications, as well as responsible for assigning orders and its items to appropriate staff to be prepared and later served.

The choice of server technology is not limited, however for the sake of demonstrating knowledge, we'll "forbid" backend-as-a-service types of platforms, such as Firebase and AWS Amplify. Otherwise, the choice of technology, the framework and other necessary tooling is up to you. The clients will consume and send data to/from REST endpoints as explained in the Appendix.





DURATION
60 min

TYPE
Interactive
assignment

3 Presentation + Q&A

As a final task of the assignment, you are tasked with presenting the implemented solution. Besides describing what was implemented, this is also the time and place to talk about various assumptions, caveats and tradeoffs done in order to get the MVP done in a short period of time.

You may also talk about what you'd do differently if you had more available time. Don't have to limit yourself to the codebase alone; you may suggest UI, process and product improvements as well.

Agenda:

- ☐ Tech stack and toolset discussion (15 min)
- ☐ Product demo (30 min)
- ☐ "Disaster scenarios" (details to be revealed - 15min)



Hint: **Keep it brief** and don't worry about being formal. For the Q&A, it's completely ok if you don't have all the answers.



Next Step: A quick wrap-up with Urban - **first impressions and feedback from both sides** and next steps.

Appendix

Guest application API calls

Below is a list of RESTful endpoints the guest application will call, along with examples of request/response structures.

GET /api/menu-items

Example response payload

```
[
  {
    "item_id": "e95321ac-f0d5-4009-8420-da8fc17c7731",
    "item_title": "Example item",
    "item_price": 29.25,
    "item_description": "Example description",
    "item_type": "food" | "drink",
    "item_image": "https://miro.medium.com/max/699/1*UfV5sxZUkgyUQIyZf8Wzjg.png"
  },
  ...
]
```

POST /api/orders

Example request payload

```
{
  "table_id": "a589d607-2843-4a35-9e49-753cabef71df",
  "items": [
    {
      "item_id": "e95321ac-f0d5-4009-8420-da8fc17c7731",
      "quantity": 1
    },
    {
      "item_id": "87678655-7f4e-4339-b1eb-848837cd683f",
      "quantity": 2
    }
  ]
}
```


Example response payload:

```
{
  "order_id": "3e1f8d4c-f9c6-4ba5-92be-ae1467e4ff1f",
  "table_id": "a589d607-2843-4a35-9e49-753cabef71df",
  "order_items": [
    {
      "order_item_id": "3ea42592-c9a6-408d-8eb0-4e6d5c1d8dcc",
      "item_id": "e95321ac-f0d5-4009-8420-da8fc17c7731",
      "status": "ordered"
    },
    {
      "order_item_id": "61801cac-2254-42d5-8118-316ce997d84c",
      "item_id": "87678655-7f4e-4339-b1eb-848837cd683f",
      "status": "preparing"
    },
    {
      "order_item_id": "15800df0-de39-49a5-bbb7-8f4cb684cda1",
      "item_id": "87678655-7f4e-4339-b1eb-848837cd683f",
      "status": "ready_to_serve"
    },
    {
      "order_item_id": "8b5882b2-0c0b-425b-aca2-fe155996c425",
      "item_id": "87678655-7f4e-4339-b1eb-848837cd683f",
      "status": "delivered"
    }
  ]
}
```

GET /api/orders/<order-id>

Example response payload

```
{
  "order_id": "3e1f8d4c-f9c6-4ba5-92be-ae1467e4ff1f",
  "table_id": "a589d607-2843-4a35-9e49-753cabef71df",
  "order_items": [
    {
      "order_item_id": "3ea42592-c9a6-408d-8eb0-4e6d5c1d8dcc",
      "item_id": "e95321ac-f0d5-4009-8420-da8fc17c7731",
      "status": "ordered"
    },
    {
      "order_item_id": "61801cac-2254-42d5-8118-316ce997d84c",
      "item_id": "87678655-7f4e-4339-b1eb-848837cd683f",
      "status": "preparing"
    },
    {
      "order_item_id": "15800df0-de39-49a5-bbb7-8f4cb684cda1",
      "item_id": "87678655-7f4e-4339-b1eb-848837cd683f",
      "status": "ready_to_serve"
    }
  ],
  "order_total": 105.35
}
```

Staff application API calls

Before accessing the staff application, the user will need to enter their username and role, without a password. The server should issue a bearer token, which will be sent by staff application with every request as a standardized HTTP Authorization header.

POST /api/login

Example request payload

```
{
  "username": "John Doe",
  "role": "chef" | "barman" | "waiter"
}
```

Example response payload

```
{
  "bearer_token": "<some form of bearer token>"
}
```

PUT /api/order-items/<order-item-id>

Example request payload

```
{
  "order_item_id": "15800df0-de39-49a5-bbb7-8f4cb684cda1",
  "item_id": "87678655-7f4e-4339-b1eb-848837cd683f",
  "status": "ready_to_serve"
}
```

GET /api/orders?has_unfinished_items=true

Example response payload

```
{
  "order_id": "3e1f8d4c-f9c6-4ba5-92be-ae1467e4ff1f",
  "table_id": "a589d607-2843-4a35-9e49-753cabef71df",
  "order_items": [
    {
      "order_item_id": "3ea42592-c9a6-408d-8eb0-4e6d5c1d8dcc",
      "item_id": "e95321ac-f0d5-4009-8420-da8fc17c7731",
      "status": "ordered"
    },
    {
      "order_item_id": "61801cac-2254-42d5-8118-316ce997d84c",
      "item_id": "87678655-7f4e-4339-b1eb-848837cd683f",
      "status": "ordered"
    },
    {
      "order_item_id": "15800df0-de39-49a5-bbb7-8f4cb684cda1",
      "item_id": "87678655-7f4e-4339-b1eb-848837cd683f",
      "status": "preparing"
    }
  ]
}
```