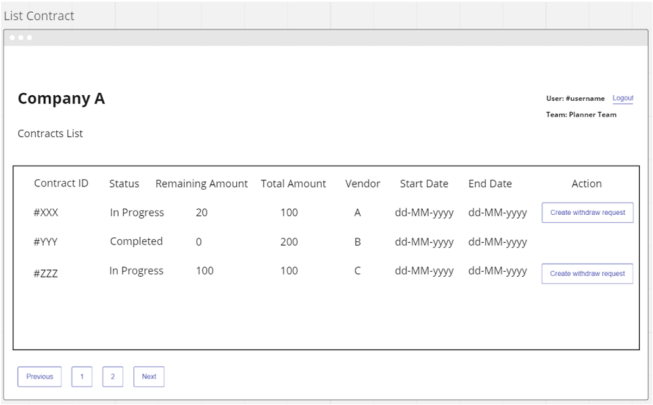
UseCase Cable Drums

Requirements


List Contract

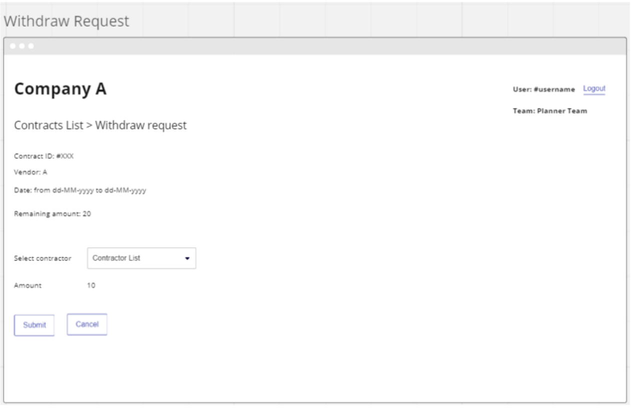
Company A

User: #username

Team: Planner Team

Contracts List

Contract ID	Status	Remaining Amount	Total Amount	Vendor	Start Date	End Date	Action
#XXX	In Progress	20	100	A	dd-MM-yyyy	dd-MM-yyyy	Create withdraw request
#YYY	Completed	0	200	B	dd-MM-yyyy	dd-MM-yyyy	
#ZZZ	In Progress	100	100	C	dd-MM-yyyy	dd-MM-yyyy	Create withdraw request


Withdraw Request

Company A

User: #username

Team: Planner Team

Contracts List > Withdraw request

Contract ID: #XXX

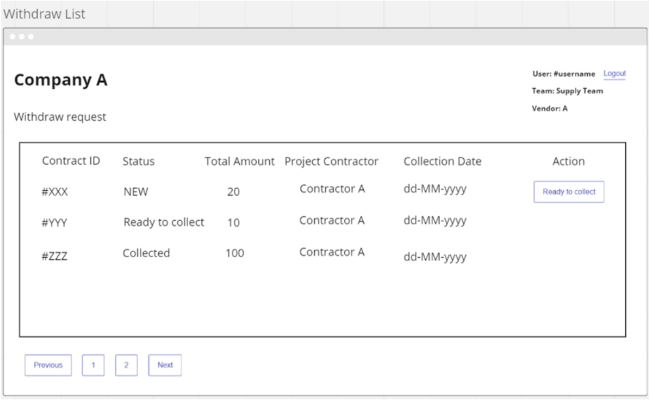
Vendor: A

Date: from dd-MM-yyyy to dd-MM-yyyy

Remaining amount: 20

Select contractor: Contractor List

Amount: 10


Withdraw List

Company A

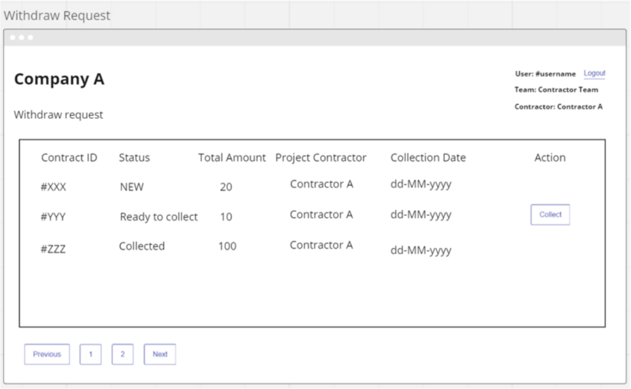
User: #username

Team: Supply Team

Vendor: A

Withdraw request

Contract ID	Status	Total Amount	Project Contractor	Collection Date	Action
#XXX	NEW	20	Contractor A	dd-MM-yyyy	Ready to collect
#YYY	Ready to collect	10	Contractor A	dd-MM-yyyy	
#ZZZ	Collected	100	Contractor A	dd-MM-yyyy	


Withdraw Request

Company A

User: #username

Team: Contractor Team

Contractor: Contractor A

Withdraw request

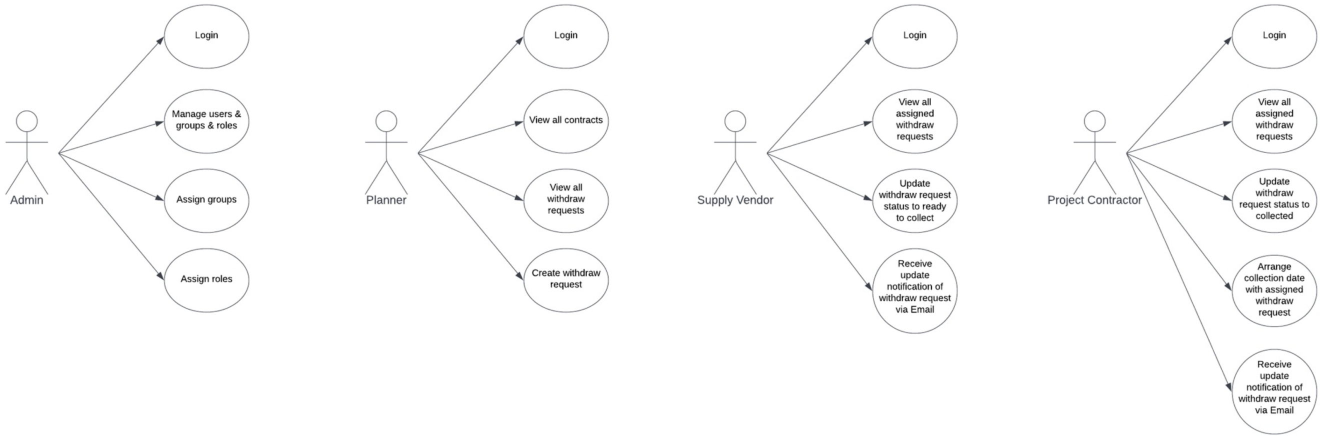
Contract ID	Status	Total Amount	Project Contractor	Collection Date	Action
#XXX	NEW	20	Contractor A	dd-MM-yyyy	
#YYY	Ready to collect	10	Contractor A	dd-MM-yyyy	Collect
#ZZZ	Collected	100	Contractor A	dd-MM-yyyy	

Company A is a business that need to manage the reliability of energy and gas supply in VN. For various projects related to build and maintain the utilities stations, cable drums is one of the most essential parts need to be used. The current processes to procure and consume cable drum are manual with lots of emails and offline coordinations. The involving parties are:

- Planner Team : staff from Company A, to do the capacity planning, procure the cable drum via Contract with Supply Vendors and manage the consumption via coordinating with Sypply Vendors and Project Contractors
- Supply Vendor Team : vendors who have the contract awarded by Company A to supply cable drums, to manufacture and stock cable drums at their own warehouse according to the contracting amount
- Project Contractor Team : vendors who have the contract awarded by Company A to build & maintain the utilities stations, to collect the cable drums from Supply Vendor warehouse and carry out the neccessary project works

There is an initiative to build the web portal which will be able to help various parties to handle the processes end-to-end digitally. The following Use Cases are critical to satisfy:

Use cases


Admin
Login
Manage users & groups & roles
Assign groups
Assign roles
Planner
Login
View all contracts
View all withdraw requests
Create withdraw request
Supply Vendor
Login
View all assigned withdraw requests
Update withdraw request status to ready to collect
Receive update notification of withdraw request via Email
Project Contractor
Login
View all assigned withdraw requests
Update withdraw request status to collected
Arrange collection date with assigned withdraw request
Receive update notification of withdraw request via Email

1. As the System Admin, I want to manage the account for users with various roles (Planner Team, Supply Vendor Team and Project Contractor Team) So that the users can login and carry out necessary actions according to their role (Assumption: System Admin account is pre-created in DB and cannot be modified, there is no need to do verification during account creation for other users)

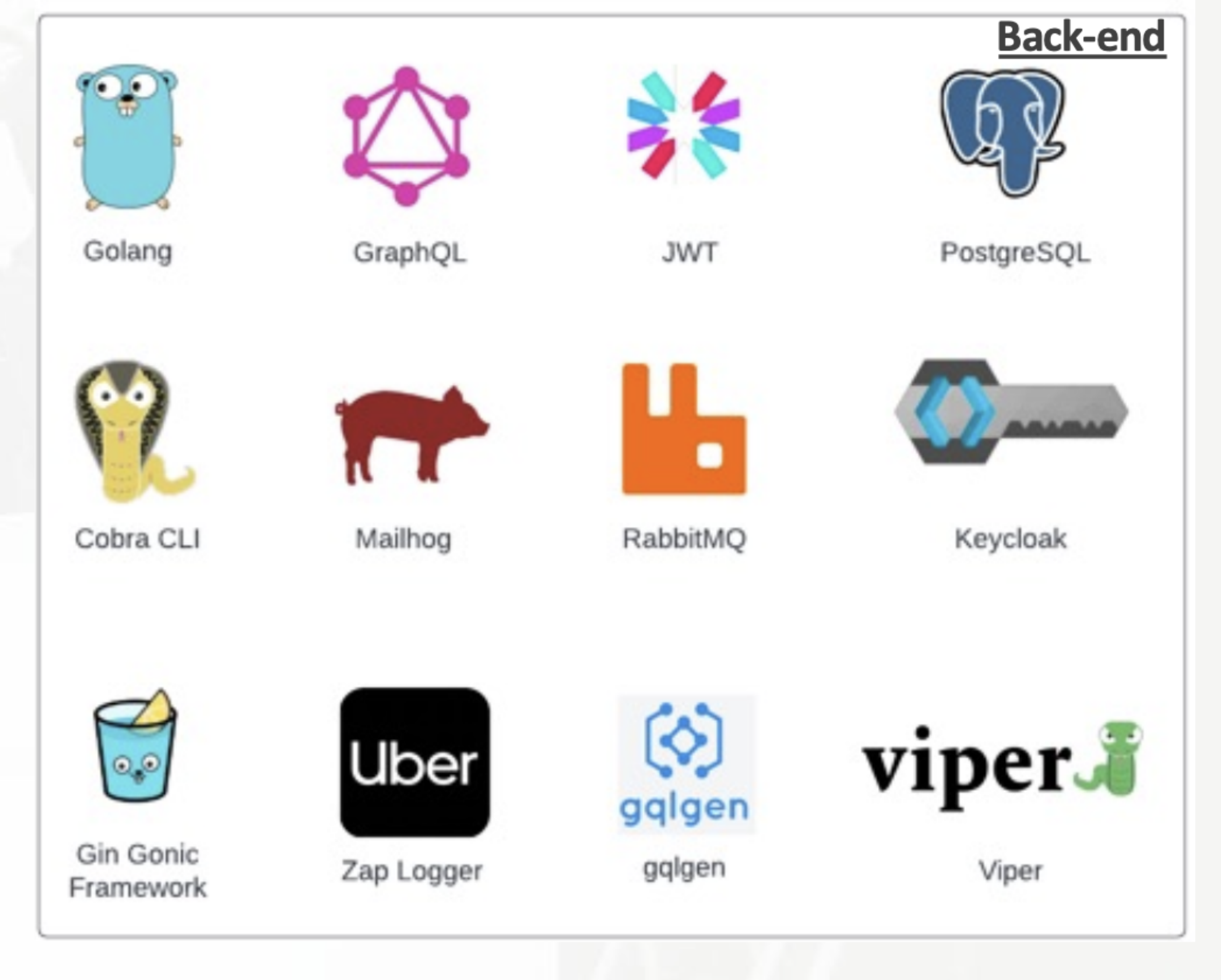
2. As the Planner Team user, I want to view all the contracts of cable drums supply in the system from all Supply Vendors, each contract will include minimally the start date, end date and contracting amount of cable drums to be supplied by specific Supply Vendor So that I can plan and manage the usage of cable drums (Assumption: procurement processes involving calling the tenders, selecting vendors/contractors, awarding contracts are not in scope of the system, we only managed the confirmed Contracts, Supply Vendors and Project Contractors. Contracts can be pre-created in DB)

3. As the Planner Team user, I want to plan and manage the usage of cable drums by: - creating the Request For Withdraw for cable drums from any available contract with Supply Vendor of choice as long as the quantity not exceeding remaining contracting amount - and assigning the newly created Request For Withdraw to the Project Contractor of choice so that the appointed Project Contractor can contact the Supply Vendor to collect the cable drums accordingly And then I can see the Request For Withdraw in the system with status "New" together with any existing Request For Withdraw with up-to-date status So that I can manage the operations with neccessary actions and solving the issues if needed

4. As the Supply Vendor Team user, I want to see the Request For Withdraw from the contract of my team So that I can prepare the requesting cable drums in warehouse for collection And then I can update the status of Request For Withdraw in the system become "Ready to Collect" when the preparation is finished So that the Project Contractor Team can arrange the collection

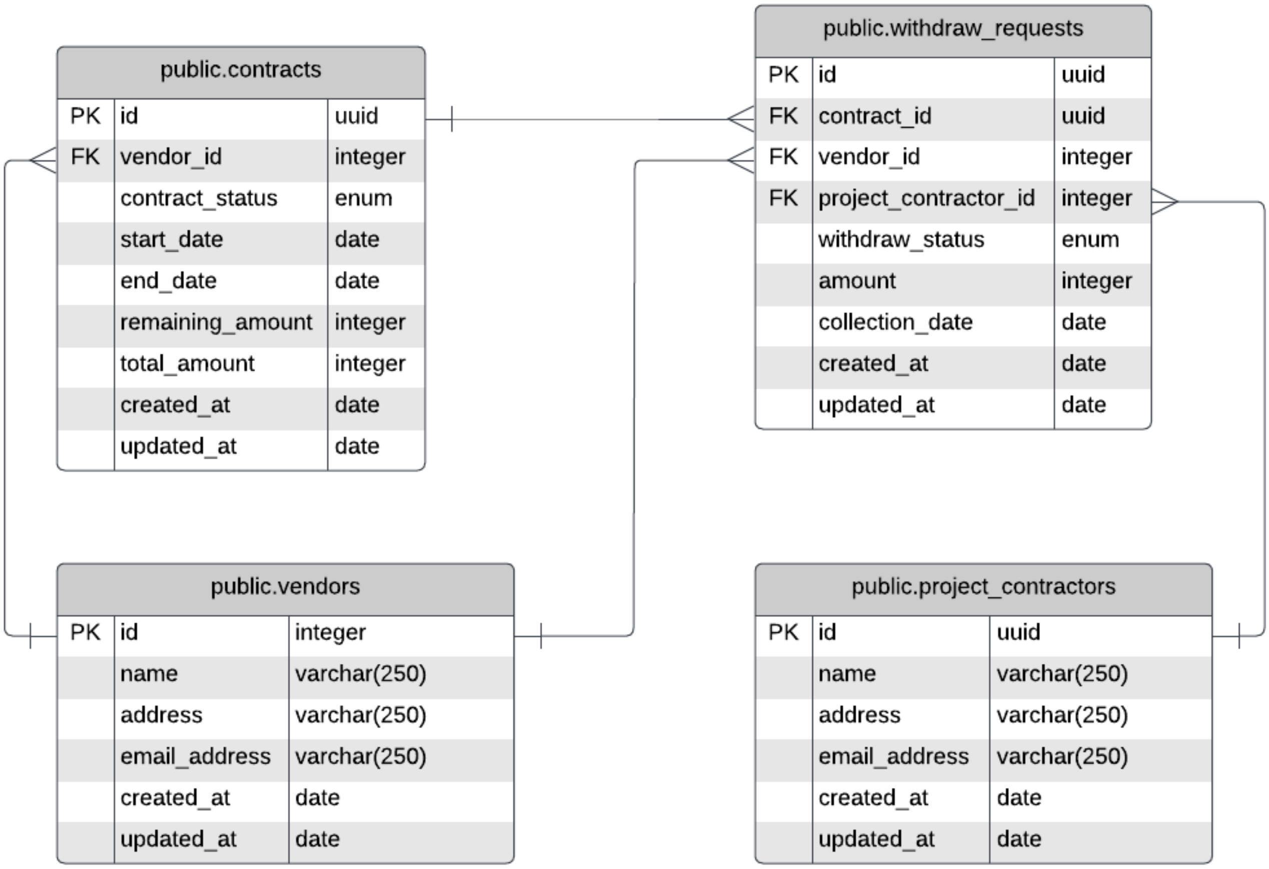
5. As the Project Contractor Team user, I want to receive notification via email when there is new Request For Withdraw assign to my team So that I can arrange the collection for cable drum to do neccesary project works And then I can log in to the System and update the status of Request For Withdraw become "Collected" So that the system can update the remaining cable drums amount of relevant contract for future planning and managing by Planner Team

Technical Requirements



- *Backend*: use Golang (version 1.20 and above). There is no limitation on the libraries you can use to build the functionalities. API must be GraphQL.
- *Authentication*: use JWT, must has role to validate permission
- *Database*: use PostgreSQL with migrations by SQL scripts
- *Tests*: please ensure that you have included sufficient unit tests in your code, coverage at least 80%. You can write test only for business code, no need for boilerplate or auto-generated code.
- *Commits*: please avoid only having 1 commit into the repository. We would like to see how the Engineer gradually built the software and the thought process applied
- *Documentations*: provide sufficient documentation so reader can understand your code structure and your design considerations, clean architecture and clean code is recommended
- *Deployment*: please use Docker and Docker Compose to compose up all deployments

Entity Relationship Diagram Design



public.contracts		
PK	id	uuid
FK	vendor_id	integer
	contract_status	enum
	start_date	date
	end_date	date
	remaining_amount	integer
	total_amount	integer
	created_at	date
	updated_at	date

public.withdraw_requests		
PK	id	uuid
FK	contract_id	uuid
FK	vendor_id	integer
FK	project_contractor_id	integer
	withdraw_status	enum
	amount	integer
	collection_date	date
	created_at	date
	updated_at	date

public.vendors		
PK	id	integer
	name	varchar(250)
	address	varchar(250)
	email_address	varchar(250)
	created_at	date
	updated_at	date

public.project_contractors		
PK	id	uuid
	name	varchar(250)
	address	varchar(250)
	email_address	varchar(250)
	created_at	date
	updated_at	date

API Design

Kind	Resource	API	Planner	Supply Vendor	Project Contractor	Unauthorized
mutation	token	getToken				X
mutation	token	exchangeToken				X
mutation	token	logout				X
query	contracts	contracts	X			
query	projectContractors	projectContractors	X			
query	projectContractors	projectContractor			X	
query	vendors	vendors	X			
query	vendors	vendor		X		
mutation	withdrawRequests	create	X			
mutation	withdrawRequests	readyToCollect		X		
mutation	withdrawRequests	collected			X	
mutation	withdrawRequests	arrangeCollectionDate			X	

Kind	Resource	API	Planner	Supply Vendor	Project Contractor	Unauthorized
query	withdrawRequests	withdrawRequests	X			

API Schema Design

- Get Token

```
mutation {
  token{
    getToken(input: {username: "planner_user_1", password: "demo"}){
      accessToken
      refreshToken
      user {
        firstName
        lastName
        username
        group
        role
        groupDetail {
          id
          name
          address
          emailAddress
        }
      }
    }
  }
}
```

- Exchange Token

```
mutation {
  token{
    exchangeToken(refreshToken: "your-refresh-token"){
      accessToken
      refreshToken
      user {
        firstName
        lastName
        username
        group
        role
        groupDetail {
          id
          name
          address
          emailAddress
        }
        }
      }
    }
  }
```

- Get All Contracts

```
query {
  contracts {
    id
    vendor {
      id
      name
    }
    contractStatus
    startDate
    endDate
    remainingAmount
    totalAmount
  }
}
```

- Get Vendors

```
query {
    vendors {
        id
        name
        address
    }
}
```

- Get Project Contractors

```
 query {
    projectContractors {
        id
        name
        address
    }
}
```

- Get Withdraw Requests

```
query {
    withdrawRequests {
        id
        contract {
            id
            vendor {
                id
                name
            }
            contractStatus
            startDate
            endDate
            remainingAmount
            totalAmount
        }
        vendor {
            id
            name
            address
        }
        projectContractor {
            id
            name
            address
        }
        withdrawRequestStatus
        amount
        collectionDate
    }
}
```

- Get Vendor By ID

```
{
  vendor(id: 2){
    id
    name
    address
    contracts {
      id
      startDate
      endDate
    }
    withdrawRequests {
      id
      collectionDate
    }
  }
}
```

- Get Project Contractor By ID

```
{
  projectContractor(id: 1){
    id
    name
    address
    withdrawRequests {
      id
      contract {
        id
        startDate
        endDate
        remainingAmount
        totalAmount
      }
      vendor {
        id
        name
        address
      }
    }
  }
}
```

- Query Withdraw Request

```
query {
  withdrawRequests {
    id,
      contract {
        id,
        startDate
        endDate
        remainingAmount
        totalAmount
      }
      projectContractor{
        id
        name
      }
      vendor {
        id
        name
      }
      withdrawRequestStatus
      amount
      collectionDate
  }
}
```

- Create Withdraw Request

```
mutation{
  withdrawRequest{
    create(input: {contractID: "f75da947-d5a0-466a-8973-ed3e21b9aa2e",
vendorID: 1, projectContractorID: 1, amount: 20}){
      id,
      contract {
        id,
        startDate
        endDate
        remainingAmount
        totalAmount
      }
      projectContractor{
        id
        name
      }
      vendor {
        id
        name
      }
      withdrawRequestStatus
      amount
      collectionDate
    }
  }
}
```

- Change Withdraw Request To Ready To Collect

```
mutation {
  withdrawRequest{
    readyToCollect(id: "2466a050-05cb-402e-a3d3-15b321f81ff9"){
      id,
      contract {
        id,
        startDate
        endDate
        remainingAmount
        totalAmount
        contractStatus
      }
      projectContractor{
        id
        name
      }
      vendor {
        id
        name
      }
      withdrawRequestStatus
      amount
      collectionDate
    }
  }
}
```

- Change Withdraw Request To Collected

```
mutation {
  withdrawRequest{
    collected(id: "2466a050-05cb-402e-a3d3-15b321f81ff9"){
      id,
      contract {
        id,
        startDate
        endDate
        remainingAmount
        totalAmount
        contractStatus
      }
      projectContractor{
        id
        name
      }
      vendor {
        id
        name
      }
      withdrawRequestStatus
      amount
      collectionDate
    }
  }
}
```

- Arrange To Collect Withdraw Request

```
mutation{
  withdrawRequest{
    arrangeCollectionDate(input: {id: "2a2dd4c9-b041-431e-bcf1-f0ca0fcc5d99",
collectionDate: "2022-08-26T00:00:00Z"}){
      id
      withdrawRequestStatus
      collectionDate
    }
  }
}
```

- Logout

```
mutation{
  token{
    logout(refreshToken:
"eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICI2ODA5YjgwNC02MjNiLTQzMDMt
OWQwYy0wMjFlNzgxNjViOGMifQ.eyJpYXQiOjE2NjE0MzAwNzcsImp0aSI6ImMyMDY4MGM3LTE5Y2
EtNDJjMy1iYzEzLWJiNTA5MjAzNTUzYiIsImlzcyI6Imh0dHA6Ly9sb2NhbGhvc3Q6ODA4MC9hdXR
oL3JlYWxtcy9zcGRpZ2l0YWwiLCJhdWQiOiJodHRwOi8vbG9jYWxob3N0OjgwODAvYXV0aC9yZWFs
bXMvc3BkaWdpdGFsIiwic3ViIjoiODdhMWZjMTktZGI2Mi00ZjdhLTkxNmYtYmY2NjkzYTA4ODg5I
iwidHlwIjoiT2ZmbGluZSIsImF6cCI6InNwZGlnaXRhbC1jbGllbnQiLCJzZXNzaW9uX3N0YXRlIj
oiNWUzZDA0MzYtYmZiYy00MTljLTk5NjUtOTBlMjY4MmZkZWJlIiwic2NvcGUiOiJvZmZsaW5lX2F
jY2VzcyIsInNpZCI6IjVlM2QwNDM2LWJmYmMtNDE5Yy05OTY1LTkwZTI2ODJmZGViZSJ9.9LxNWUA
Zukue5makV78VAQ-3LzaCqF8IRIYr7ZCbgP8")
  }
}
```