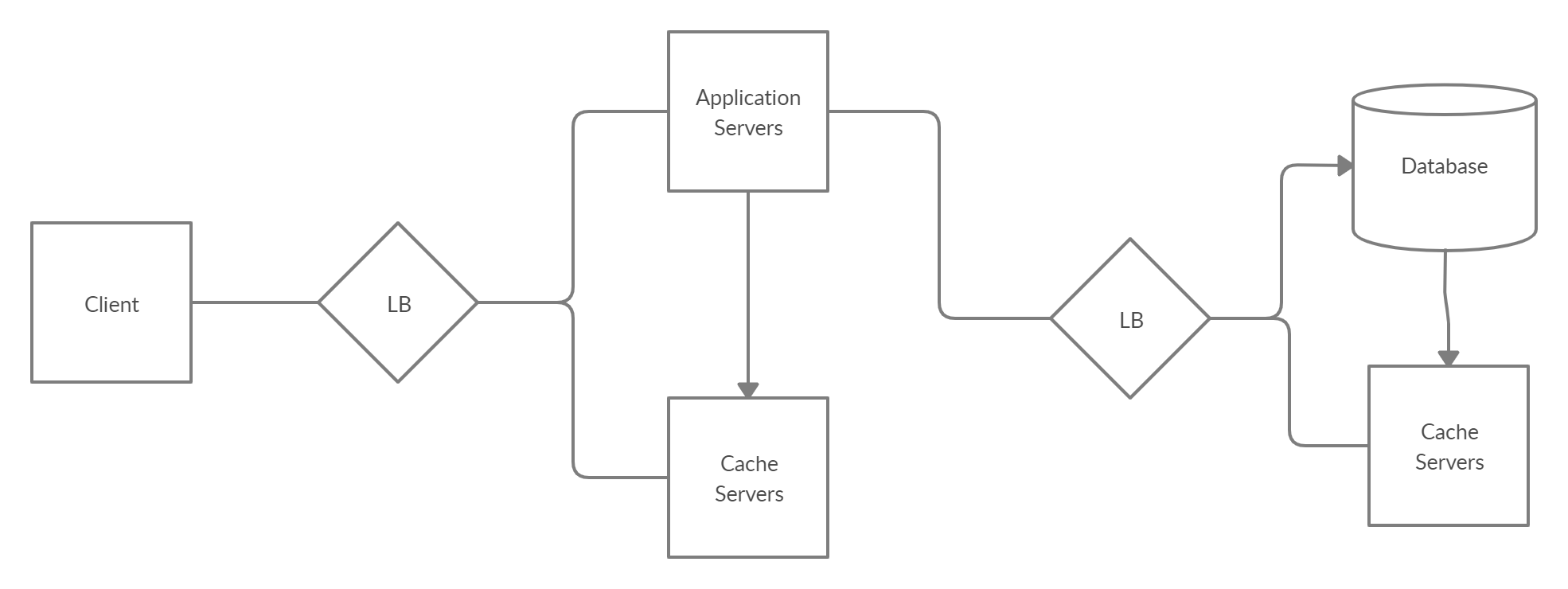
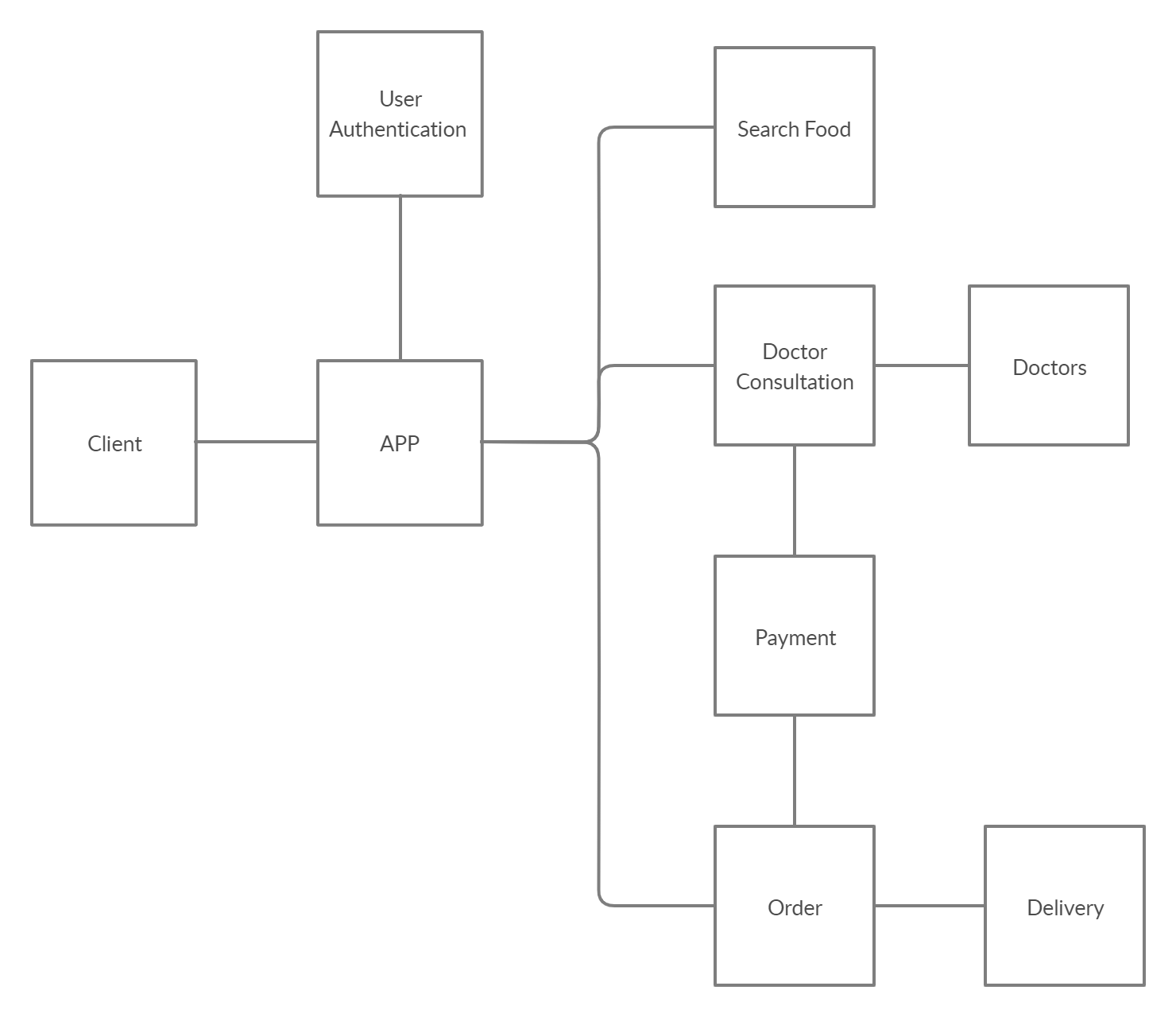
Design for any feature:



Microservice doc:



Design folder in creately: <https://app.creately.com/manage/project/uwuAgaOvuCl>

**User Authentication Service**

User information:

Phone no - primary key

Email

Name

Age

Gender

OTP

AuthToken

Rest APIs:

1. JsonString SignupUser(api\_key, name, email, age, gender, phoneNo)

Signs up user-

Return says whether there is an error or success

Return says why the error like duplicate no

If success invokes GenerateOTP

1. JsonString GenerateOTP(api\_key, phoneNo)

Generates OTP for a user

Return says whether there is an error or success

Return says why the error

Return says whether the client needs to retry

1. JsonString VerifyOTP(api\_key, phoneNo, OTP)

Verifies OTP for a user

Return says whether there is an error or success

Return says why the error

Return says whether the client needs to call GenerateOTP first

If success, return gives authToken

1. JsonString UpdateUser(api\_key, name, email, age, gender, phoneNo, authToken)

Verifies authToken and updates user

Return says whether there is an error or success

Return says why the error

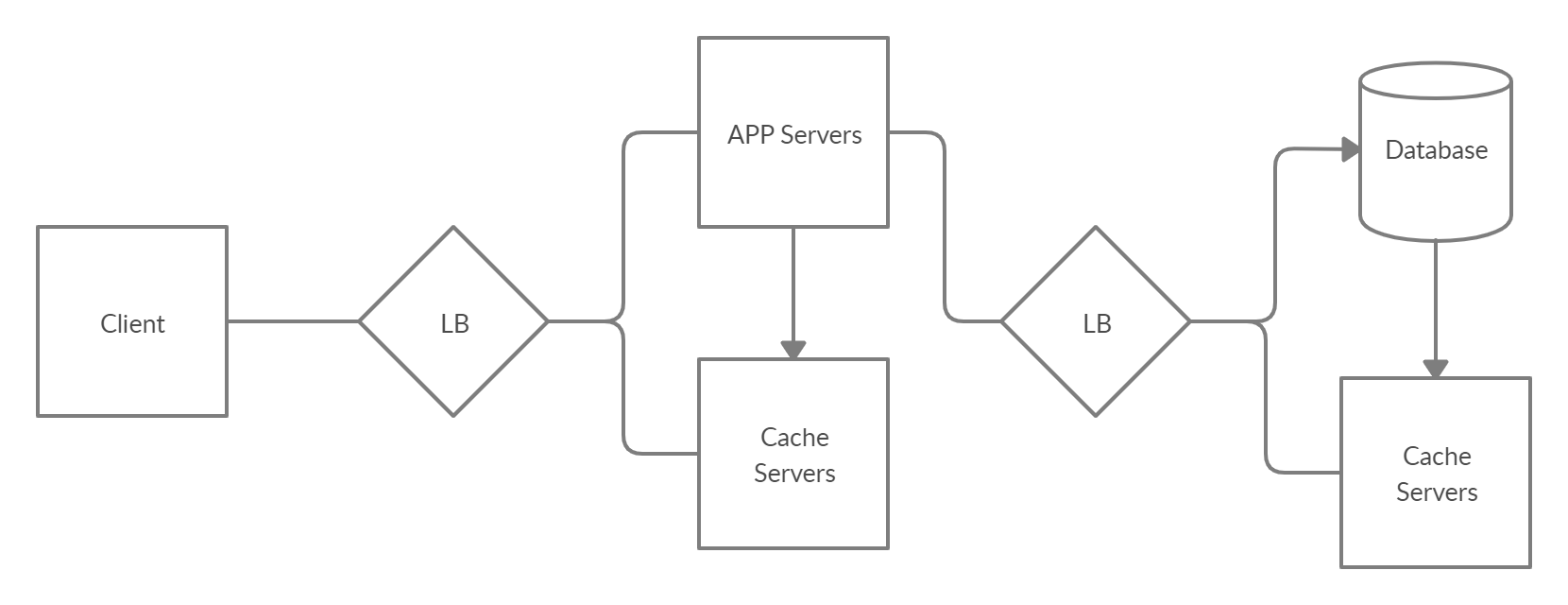
If success, return gives authToken

1. JsonString VerifyToken(api\_key, phoneNo, authToken)

Verifies authToken for a user

Return says whether there is a match or not

HLD:



Logic:

1. SignupUser stores user data into database. It also generates OTP and sends it to the user’s phone no. It also does some more things like checking if the no already exists and so on.
2. GenerateOTP checks if no exists. It generates OTP and sends it to the user’s phone no. It stores the OTP into the database
3. VerifyOTP verifies the given OTP and generates an authToken. It stored the token in DB and returns authToken
4. UpdateUser verifies the authToken and updates the user information and returns new auth token
5. VerifyToken verifies the given AuthToken

Capacity Estimations:

Obviously, the data will be less but we still will store it in multiple nodes

Database:

We will use MYSQL as our database as we need ACID properties here. We do not want inconsistencies with user data

(TODO need to decide)

Sharding:

We will shard based on phone no

LLD:

