Explore   Problems   Contest   **Discuss**   Interview ⌄   Store ⌄   🔔   🔥 0

‹   ...

Flipkart | Machine Coding | Design Online Coding Platform CODING BLOX / Leetcode LLD

Hariom Yadav
👁 35878   📅 Apr 26, 2021   ✎ May 08, 2022

Flipkart   System Design   Low Level Design

Object-Oriented Design   Interview

Design Online Coding Platform CODING BLOX (Design Leetcode LLD)

- Coding Blox is an Online Coding Platform that allows a user to Sign Up, Create Contests and participate in Contests hosted by Others.

- Each contest can have a level (LOW, MEDIUM, HIGH) and will contain a set of questions.

- Each question will have different levels of difficulty(LOW, MEDIUM, HIGH) and score.

- Based on the contest level, the question set is going to be decided. Contest level with LOW difficulty will have questions with LOW difficulty level.

- Final score will be decided based on the difficulty LEVEL chosen for a contest

- Users solve problems and get points based on the difficulty of the problems and after the contests scores of the users are updated.

Functionalities/Requirements:

1. CreateUser <user_name>
2. CreateQuestion <difficulty_level>
3. ListQuestion <difficulty_level>

Explore

4. CreateContest <contest_name> <contest_level>
   <contest_creator_user_name>

5. ListContest <difficulty_level>

6. AttendContest <contest_id> <user_name>

7. RunContest <contest_id>
   <contest_creator_user_name>

8. LeaderBoard <sorting order asc/desc>

9. ContestHistory <contest_id>

10. WithdrawContest <contest_id>

full problem statement & solution (Time given
90min) : http://bit.ly/leetcode-low-level-design

Simply I follow these steps for any LLD.

1. Find out entities
2. Find out the relationship between entities
3. Write entities CRUD (use hashmap or JPA)
4. Start implementing requirements

Step 1,2 and 3 => approx 30min
Step 4 => +1hr

Step 1, 2: Domain layer
Step 3: Database/dao layer
Step 4: Service layer

(***Use this hashmap template to quick start:***
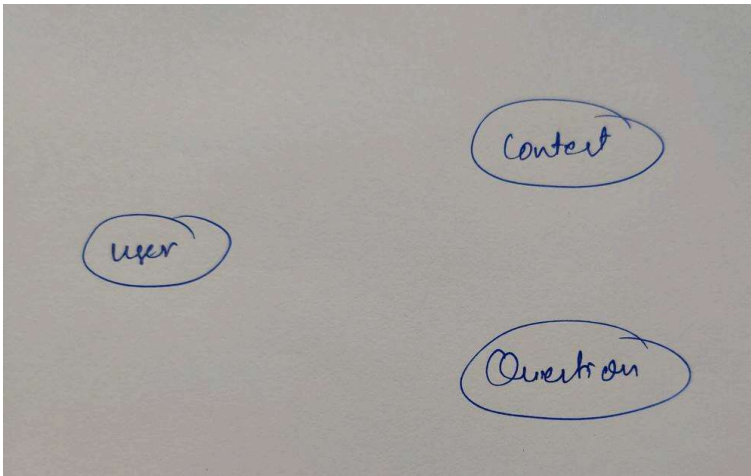*http://bit.ly/lld-hashmap-as-db-template*)

Step 0: *Understand the problem statement and
requirements*

Step 1: *List down important entities/model/domain*

In the above question list of entities are -> **User**,
Contest, and **Question**

Step 2: *List down the relationship between entities.*
e.g.
User can register for multiple contests

```
User:
    List<Contest> contests
```

The contest can have multiple Questions

```
Contest:
    List<Question> questions
```

Now Add other fields of entities

```
User:
    username
    score
    List<Contest> contests -- (A)
```

```
Contest:
    name
    level
    status
    List<Question> questions
```
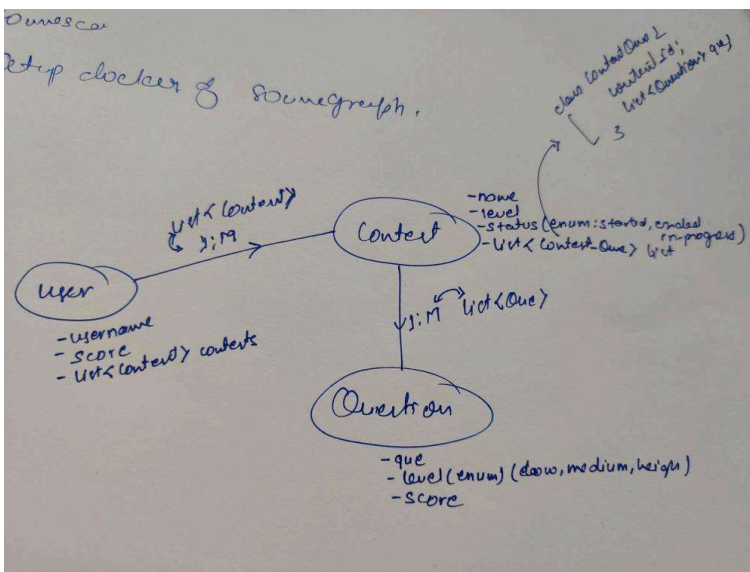
```
Question:
    question
```

```
level
score
```
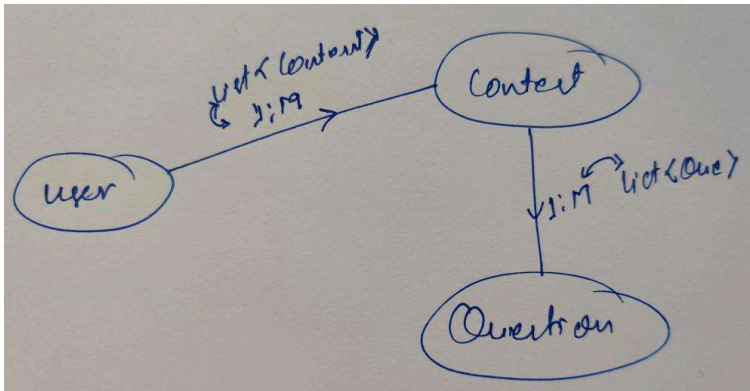

Step 3: *Store entities and write entities CRUD methods **(Use Method 1)***
Method 1: Use hashMap to store entities,

```
Map<Long, User> userDb
Map<Long, Question> questionDb
Map<Long, Contest> contestDb
```

Write CRUD method for each entity (**HashMap sample example**)

Method 2: Use H2 in memory using JPA, (simple to use but need to practice more)
Create 3 DAO class, internally with the help of the

JPA class we can get all CRUD methods so no need
to create any methods (**JPA sample example**)

Step 4: *Create a service class for each entity and start*
implementing the requirement (**service classes**)

```
UserService:
    createUser(username)
    attendContest(contest id or contest object, u
    leaderBoard()
    withdrawContest(contest id or contest object,
```
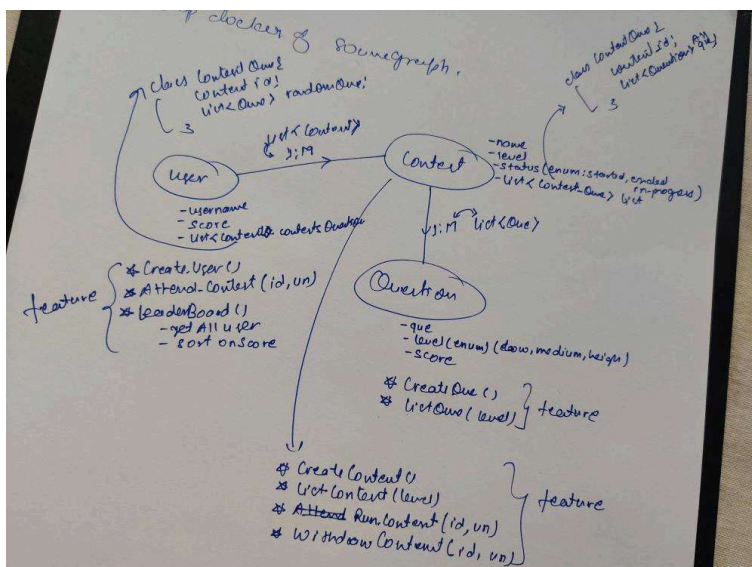
```
QuestionService:
    createQuestion(question)
    listAllQuestion()
```

```
ContestService:
    createContest(contestId or contest object, us
    listAllContest()
    runContest(contestid or contest object, usern
    contestHistory()
```

add logic for above methods plus entity validation

Step 5: *Refactor code : extract class to an Interface,*
Abstract class, add enum, add exceptions, add
constant, read from config

```
 List<Contest> contests --(A) become --> Map<Long,
List<Long>> userContestQuestions = new HashMap<>();
KEY : Contest Id, Value: list of question ids
```

If we do Step-1 and Step-2 correctly then finally we
can see entities tables (these tables are only visible in
case of JPA but this entity relationships are also valid if
we are using HashMap)

Contest entity

SELECT * FROM CONTESTS;

ID	CONTEST_QUESTIONS	LEVEL	NAME	STATUS	USER_ID
1	{"questions":[1,2,7]}	low	contest 1	ended	1
2	{"questions":[3,4]}	medium	contest 2	started	2

Question entity

SELECT * FROM QUESTIONS;

ID	LEVEL	QUESTION	SCORE
1	low	que 1	10.00
2	low	que 2	20.00
3	medium	que 3	20.00
4	medium	que 4	30.00
5	high	que 5	40.00
6	high	que 6	50.00
7	low	que 7	10.00

User entity

```
SELECT * FROM USERS;
```

ID	SCORE	USER_CONTEST_QUESTIONS	USERNAME
1	1480.00	{"userContestQuestions":{"1":[1,2,7]}}	hariom
2	1480.00	{"userContestQuestions":{"1":[1,2,7]}}	chandan
3	1480.00	{"userContestQuestions":{"1":[1,2,7],"2":[3,4]}}	naveen
4	1500.00	{"userContestQuestions":{"2":[3,4]}}	omprakash

Other LLD example:

Problem statement (time given ~6 to 8hr):

https://www.geektrust.com/challenge/my-money

My solution :

https://github.com/hocyadav/geektrust_lld_mymoney

Comments (32)   Sort by: Best ⌄

```
Type comment here...



⟨/⟩  ⊝  @                    Comment
```

f_society 🔶   Apr 26, 2021

Wow I have no idea how to approach these questions!
Where do I practice and how do I start?

⬆ 57 ⬇   ⊙ Show 11 Replies   ⤶ Reply

Shubham6677 🟢   Aug 28, 2021

Hi,
In this problem a user can participate in multiple contests
and a contest can have multiple users participate in it. Then

shouldn't user-contest relationship be many-to-many and shouldn't we create a different UserContestMapping class along with User and Contest class.

⬆ 22 ⬇   ↩ Reply

Hlib Pylypets 🔵   Apr 27, 2021

Were they expecting you to design/implement a server-like system including connections handling, scalability, advanced logging(ELK), errors, or is it purely one of the OOP tasks where the focus is on the local entities' communication and structure?

⬆ 8 ⬇   ◯ Show 2 Replies   ↩ Reply

AlphaRameo ⬡   Apr 29, 2021

I can see tight coupling everywhere. First try to make interfaces or abstract classes and try to implement child classes. For example, User class. You have used it only for one purpose i.e. customer but what about Admin and Other users who works for the website maybe a call center employee. Also try to use design pattern.

⬆ 7 ⬇   ◯ Show 2 Replies   ↩ Reply

Vijay Ram 🔶   Apr 28, 2021

This post is a gem, were you allowed to use ide ?

⬆ 5 ⬇   ◯ Show 7 Replies   ↩ Reply

---

stillanovice   May 21, 2021

Hi All,

Here is the first look of what I have thought at the first instance, I say I am really poor in LLD questions, still picking up stuffs. Please correct me in my following design, really looking forward to master the thinking techniques for these kind of questions.

I am able to find the entities, create a mapping between them but unable to bind each other, like addQuestion

service in the following could be a method in admin class
or user class as well, Let's say I decide it that it should be
with admin, then in the main how can I achieve that, since
in my main I have a list which maintains questions. How
can I make an Object of admin edit questions maintained
by another object of CodingBlox (in our case). How could I
make it more robust?

Thanks in advance for al the help! Please excuse me for any
stupid mistakes I am just learning stuff related to OOPS,
design etc..

```python
import abc


class account:
    def __init__(self, name, phone, email):
        self.name = name
        self.phone = phone
        self.email = email


class person (metaclass=abc.ABCMeta):
    @abc.abstractmethod
    def create (self):
        pass


class admin (person):
    def __init__(self, account, password, admin]
        self.account = account
        self.password = password
        self.adminID = adminID


    def create(self):
        print("Create user of type ADMIN")


    def createContest(self, contestID):
        print("Add contest ID to the list of cor


class user (person):
    def __init__(self, account, password, score)
        self.account = account
        self.password = password
```

```python
        self.score = score

    def create(self):
        print("Create a type of user USER")

    def registerContest(self, contestID):
        pass

    def withDrawContest(self, contestID):
        pass

    def incScore(self):
        pass

    def decScore(self):
        pass


class contest:
    def __init__(self, name, contestID):
        self.name = name
        self.contestID = contestID
        self.status = status
        self.questions = []
        self.users = []

    def registerUser(self, user, contestID):
        pass

    def addQuestion(self, questionID):
        pass

    def setStatus(self):
        pass

    def getStatus(self):
        pass

    def getallQuestions(self):
        pass
```

```python
    def getallUsersforContest(self):
        pass


class question:
    def __init__(self, points, description):
        self.description = description
        self.points = points
        self.level = level


    def editQuestion(self, description, score):
        pass


class CodingBlox:
    def __init__(self):
        self.users = []
        self.questions = []
        self.contests = []
```

⇧ 4 ⇩   ◯ Show 1 Replies   ↰ Reply

XYZ123 🔟   Apr 28, 2021

Would it be better to create abstract classes for contest and question? Then create concrete classes for different levels? Then use Factory method or abstract factory to create objects depending upon hardness level?

⇧ 4 ⇩   ◯ Show 2 Replies   ↰ Reply

---

v21   Apr 27, 2021

Are these questions required to be solved in java only? I wonder if someone has used python in LLD round.

Can someone please share resources for LLD in python?

⇧ 4 ⇩   ◯ Show 1 Replies   ↰ Reply

---

manas sharma 🟩   Apr 27, 2021

how to practise and approach these questions

⇧ 3 ⇩   ◯ Show 3 Replies   ↰ Reply

◯ rjvir   Apr 27, 2021

For how many yoe was it asked?

⇧ 2 ⇩   ◯ Show 1 Replies   ↰ Reply

⟨   **1**   2   3   4   ⟩