SAS151830
Siddhanth Shah

Homework - 1

Ans - 1

a) HTTP GET

→ Get method ~~over~~ retrieves data only. It should not have any other effect.
→ Method once called, it request a representation of the specified resource.
→ Example: when we search a website to check ongoing movies on the particular theatre it ~~gives~~ request GET method as bellow:

```
GET martian/cinimax HTTP/1.1
Host: www.fandago.com
```

→ The property of GET is that it will always give same detail ~~to~~ for every request in response,

b) HTTP PUT and c) HTTP POST

→ Both PUT and POST can be used for creating new resource, ~~but~~
→ The PUT method requests that the enclosed entity be stored under the supplied URI.
→ If the URI refers to an already existing

resource, it is modified. If the URI ~~does not~~ points to new resource, then it is created.

Example

Let say we are maintaining question bank for some subject.
Now if I want to add new question following request will be sent:

```
PUT /question/new HTTP/1.1
Host: www.College.edu
```

→ Here se if new was already present then it would get modified.

Now POST method:

→ This method requests the server to accept the entity enclosed in the request as a new subordinate of the web resource.

→ In above example if "new" is not present on server, then request will give error, so that is difference between POST and PUT.

→ So post can update and PUT can

create and modify.

⇒ Also GET is concidered safe, while POST and PUT are not safe, which means that GET is intented only for information retrival and should not change the state of the server.

→ While POST and PUT are not safe for financial transactions and/or transmission of email.

→ Get is cached if required, POST is not.

d) HTTP TRACE

Sometime information regarding server's method and other information is required for development. TRACE method is useful for that.

→ It echoes the received request so that a client can see what changes or additions have been made by intermediate servers.

```
TRACE /HTTP/1.1
Host: www.chase.com
```

This will reply with related data on the path to the server to the chase.com.

e) HTTP OPTIONS:

When in need to find which OPTIONS (HTTP methods) are supported by the server, then HTTP OPTION method is used.

As an example, if we are making an API using facebook Friend search method and adding some feature to it, then we need to know that what all method are supported by API Host server:

OPTION * HTTP/1.1

~~User Agent:~~

Host: facebook.com

→ It will reply with allowed ~~and not allowed~~ method to the server, which in this case would be All.

(2) Shared - Nothing architecture:

* Shared Nothing architecture has following features:
  (i) No shared memory / service
  (ii) No shared disk. (peripheral storage)

→ Shared nothing is the most cost efficient alternative

* Some benifite of shared nothing with comparison of "Shared Disk" and "Shared memory" architectures.

→ Easy to scale to the large number of users.
→ High avialability.
→ Less difficulty for concussency control.
→ Good recovery options.

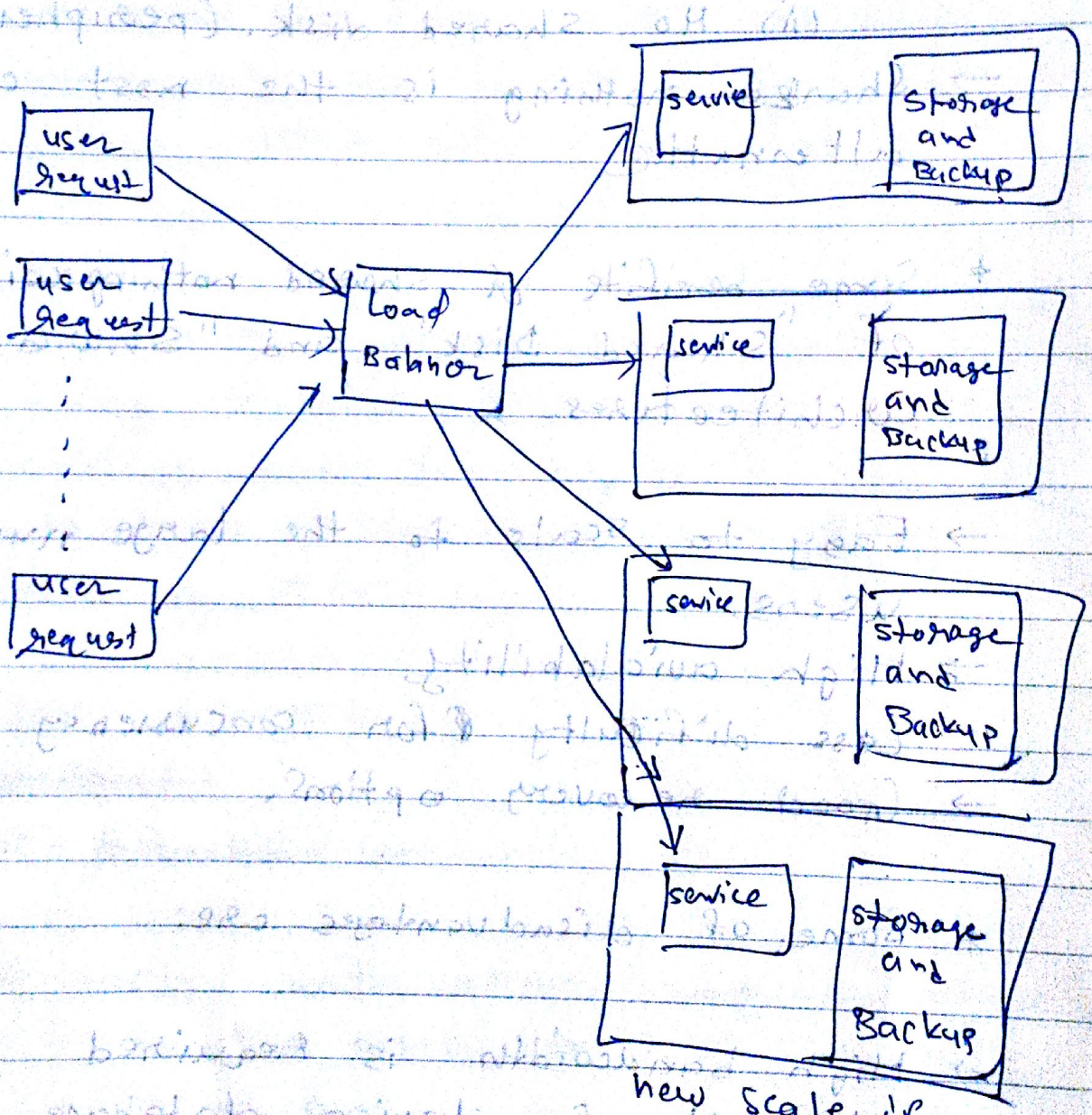
* Some of disadvantages are:

→ High bandcoidth is required
→ Difficult to design database system
→ Number of system replications is high

Example

→ Web application which provides service to find music over internet.

* Let define its architecture as bellow.



→ As shown in the figure, ~~the~~ ~~the~~ users request

for some music will be sent to the load balancer, which will define where to sent the reuist to which of service and server.

→ If one of the server is down, it can be redirected to other server, so it is always available.

~~→ Also if it is redia~~

→ Now if users number increase, then we can add new setup which is indepen to others, as they are no sharing anything

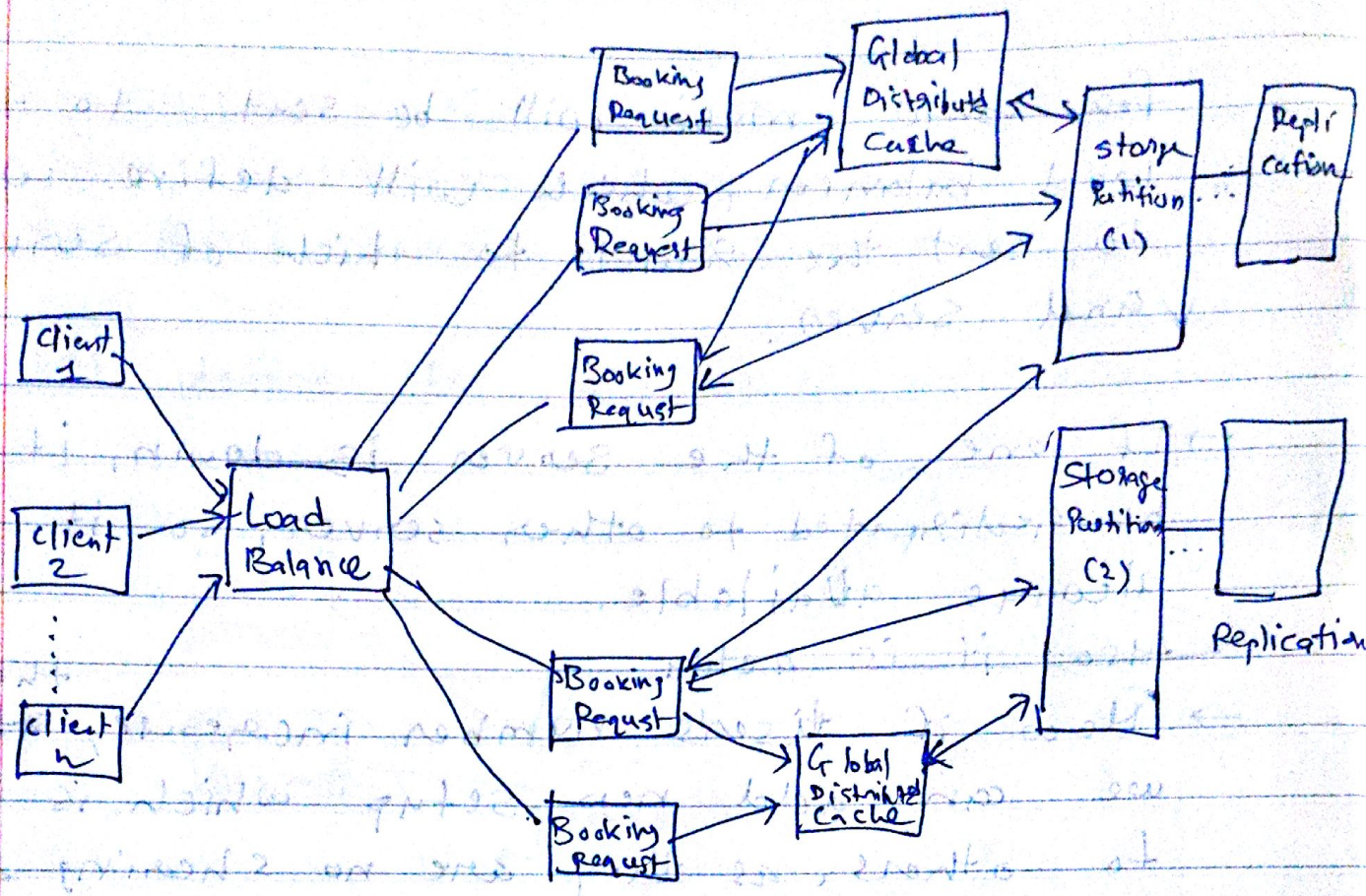
→ So it is highly scalable,

→ Also all node can contain same detail over time with communication, so recovery can be made easily.

* But the design part of it is very complex and highly difficult. Also, data reliability is also something which need to apply carefully.

→ As each infrastructure have their own requirement, ~~it is hard to get~~ it will consume lot of bandwidth and storage itself also.

(3)



* Here client's request goost to load balancer which define which request node to select for provide one of two service:
  - flight search
  - travel booking
* Once request is received, it will check in designated global cache for flight search.
→ If not found, node can access any of storage point based on assumption, that it has some efficient algorithm to locate the appropriate server.

→ Distributed caches will be updated from database to keep fresh data.

→ Also all storage partition will be working as shared-nothing architecture, so they do not need to worry about synchronization and as said before a good efficient algorithm will define which data will be stored where,
Ex: A-P named flight in partition one and P-Z named flight in partition two.

→ To get reliability of service, we have many services. Also we store all data backups for each partition.

* So following requirement will be satisfied as below:

(a) Very fast flight search:
→ we have global distributed Cache, so that it will boost frequent searches.
→ Also storage's are Indexed and flight information updation is relatively less to other systems (like retail), so direct query to database (in case of global cache failure) will give fast

~~result~~ results.

(b) Highly available travel booking

→ We have multiple instances of services

→ So in case of failure of one service node, other service node will take over.

→ Also periodic database recovery to recovery databases will make sure to provide fresh data, if in case of database failure.

(C) Reliable travel itinerary access:

→ This is shared nothing architecture where each partition is designed to be responsible for data contained by it only. Example: if partition I is have flight itinerary of A-L flights, ~~its data will~~ it doesn't need to worry about keeping consistency with flight named M-Z.

41

→ Global cache keep updating with updates in the database. So all data present in cache is fresh. So it gives high reliability to the Clients.