GeekBand 极客班

互联网人才 + 油站!

GeekBand 极客班

www.geekband.com

硅谷公司及系统设计介绍

大纲

- IT公司大致分类 （地理位置，热门）
- 创业型明星公司 （融资规模，offer长相）
- 创业公司技术类型特点 （云计算）
- 行为面试 （通过技术面试后，需要过这一关）
- 准备过程（情商，演练，知己知彼）
- 实战演练：短URL设计
- 案例：Rate Limit
- 常见问题

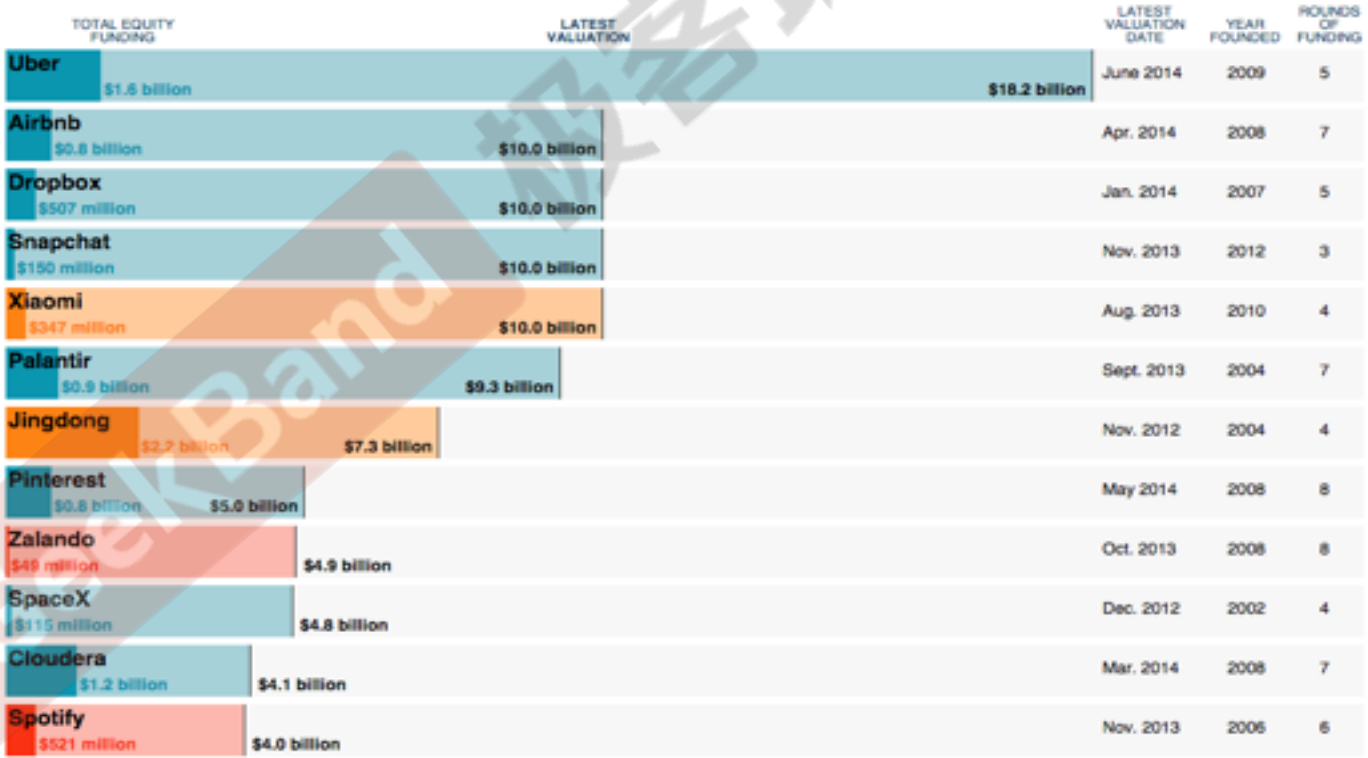
IT Company Category

By Scale

- Mature Company
  - Microsoft, Amazon, Yahoo, Oracle ...
- Public Company
  - Google, Facebook, Linkedin, Twitter ...
- Pre IPO (200 - 1000)
  - Box, Uber, Cloudera, Square ...
- Startup (10 - 100+)
  - Quora, Houzz, Coursera ...

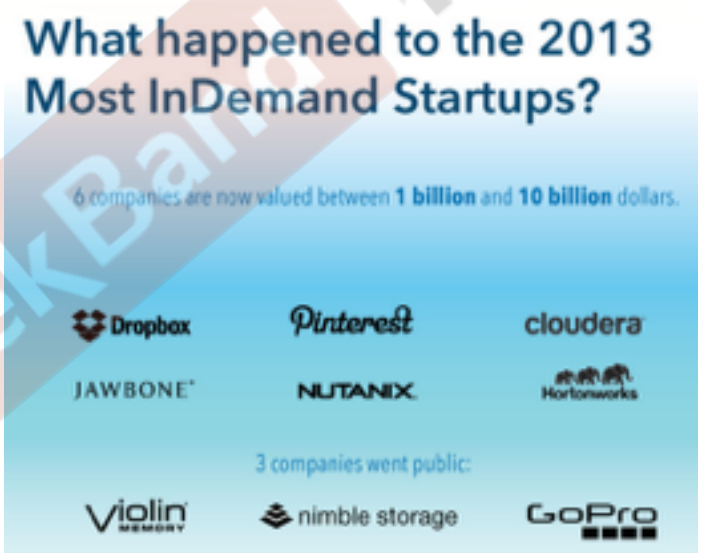
热门创业公司

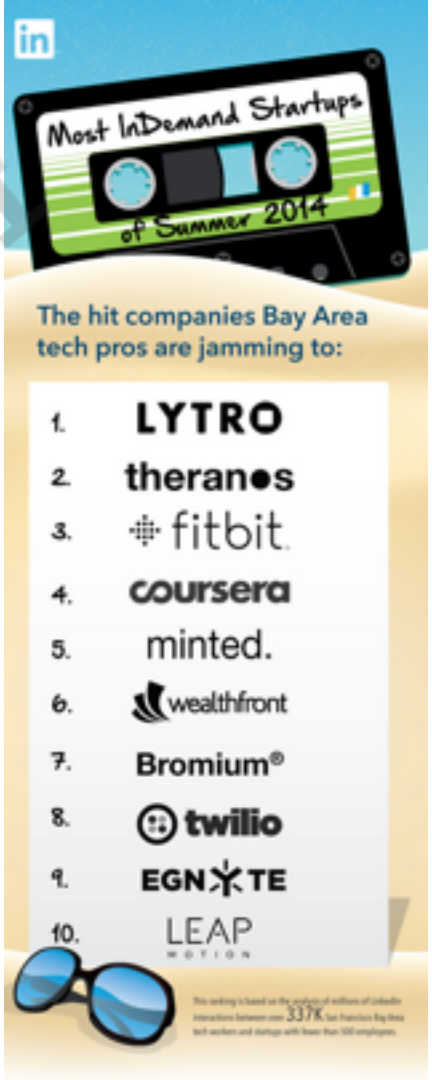
- Mobile
- Payment
- Search
- Wearable
- Security
- Cloud
- Big Data
- Ads
- Consumer Internet



Star startups

LinkedIn Indemand Startups over years





Tech Pick

- Language
  - Java, Scala, Javascript, Go, Python
- Framework
  - Spring, Play, Node.js
- Data flow
  - Hadoop MapReduce, Kafka, Spark
- Storage
  - HDFS, MySQL, NoSQL, Memcached

Tech Pick

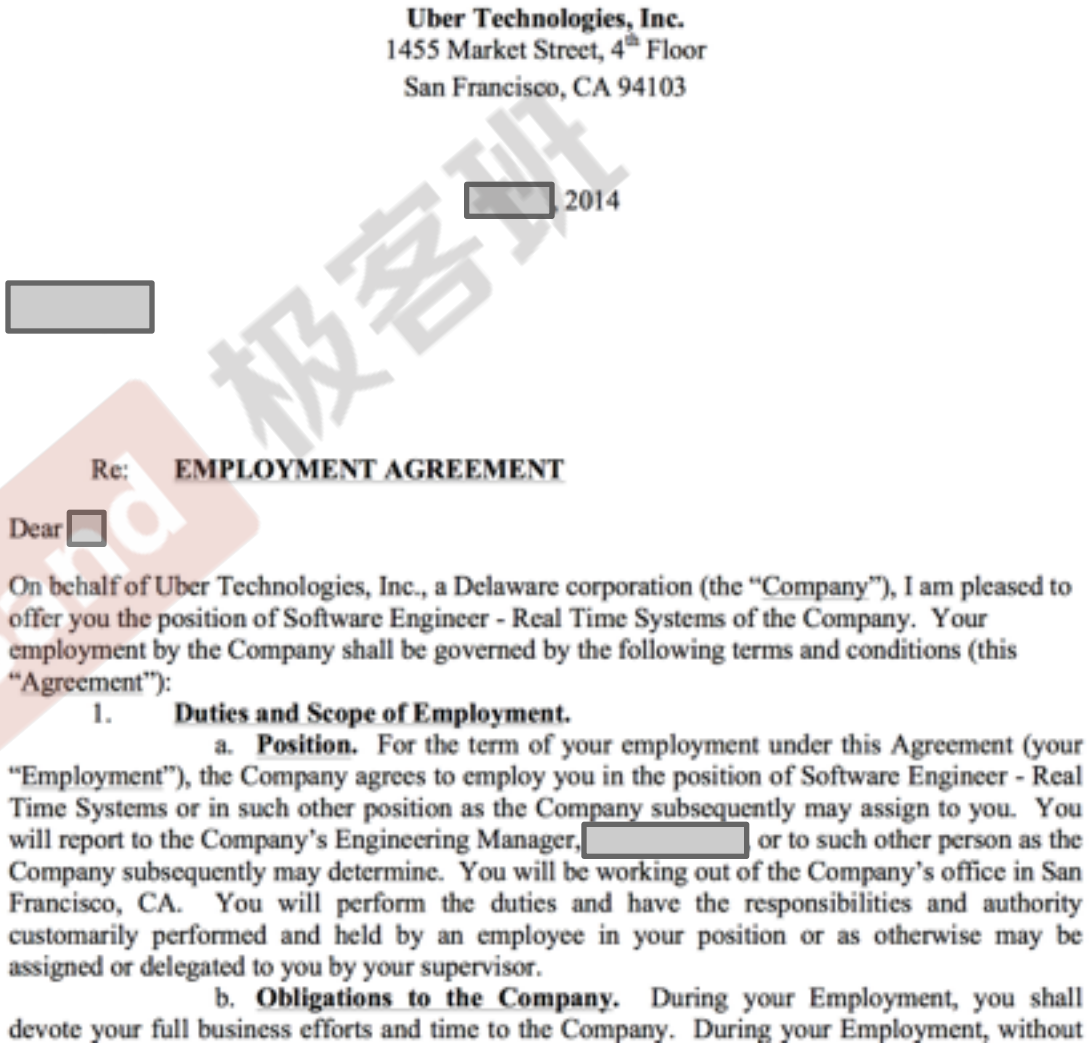
amazon web services

VS.

Google Cloud Platform

Offer什么样

- Position
- Salary
- Stock Options
- Vacation
- Employment at will



Uber Technologies, Inc.
1455 Market Street, 4th Floor
San Francisco, CA 94103

[redacted], 2014

[redacted]

Re: **EMPLOYMENT AGREEMENT**

Dear [redacted]

On behalf of Uber Technologies, Inc., a Delaware corporation (the "Company"), I am pleased to offer you the position of Software Engineer - Real Time Systems of the Company. Your employment by the Company shall be governed by the following terms and conditions (this "Agreement"):

1. **Duties and Scope of Employment.**

a. **Position.** For the term of your employment under this Agreement (your "Employment"), the Company agrees to employ you in the position of Software Engineer - Real Time Systems or in such other position as the Company subsequently may assign to you. You will report to the Company's Engineering Manager, [redacted] or to such other person as the Company subsequently may determine. You will be working out of the Company's office in San Francisco, CA. You will perform the duties and have the responsibilities and authority customarily performed and held by an employee in your position or as otherwise may be assigned or delegated to you by your supervisor.

b. **Obligations to the Company.** During your Employment, you shall devote your full business efforts and time to the Company. During your Employment, without

System Design的面试的一般流程

1. 题目描述
   a. 往往非常简单，如：设计一个XX系统。 或者：你有没有用过XXX，你来设计一个。
2. Clarification
   a. 面试者需向面试官询问系统的具体要求。如，需要什么功能，需要承受的流量大小，是否需要考虑可靠性，容错性等等。
3. 面试者提供一个初步的系统设计
4. 面试官这对初步的系统中提出一些Follow-Up的问题：如果要加某个功能怎么办，如果流量大了怎么办，如何考虑Consistent怎么办，如果机器挂了怎么办。
5. 面试者根据面试官的Follow Up逐个解决问题
6. 完成面试

特点：
不需要写程序
纯粹聊天扯淡

Design Evaluation

- Guidelines:
  - Adapt to the changing requirements
  - Produce a system that is clean, elegant, well thought
  - Explain why you choose this implementation
  - Be familiar with your experience level to make decisions
  - Answer in high level of scale and complexity

Tips

Know your stuff
Understand the problem
Make assumption
Describe your implementation
Do not mention a naive approach to fill time
Ask questions about the interviewer

Best Practice

Have Passion
Find Vision
Use Product
Culture Fit
Open Source
Prepare Questions

常见的系统设计类问题

1. Tiny Url
2. RateLimit
3. Messenger / Chatroom
4. Stats Server
5. News Feed
6. Web Crawler
7. Location Based Service
8. Web Application
9. Word Count

Design TinyURL

- Design a system to take user-provided URLs and transform them to a shortened URLs that redirect back to original.

Evaluation

How to Evaluate System Design Interview?

Scoring	*Candidate*	*Criteria*
1.0	*Bad*	*No sense of requirement, no scoping*
2.0	*Pool*	*Limited knowledge, common sense*
3.0	*Good*	*Reasonable Solution, explain clearly*
4.0	*Great*	*Out of expectation, well thoughtful, tradeoff*

Level 1.0

- Devise an encoding/compression scheme from full URL to 5 character code
- Use a single database to store mapping from short URL to original URL
- Retrieve short URL. If exist, return stored URL, otherwise, return null.

ID	URL
1	http://www.google.com
2	http://www.yahoo.com
3	...
4	...

Level 1.5

1. URL Encoding:

t.cn/000001
t.cn/000002
t.cn/000003

URL--> DB ID --> Short URL
Short Url --> ID --> URL

1.0 --> 0~99999
1.5 --> 64^5 (a..z A..Z 0..9 _ -)

2. Cache
    1. Read/Write rate?
    2. LRU / LFU

ID	URL
1	http://www.google.com
2	http://www.yahoo.com
3	...
4	...
...	...
99999	...
100000	...
100001	...

Why Bad?

1. Performance: what if 10000 qps (queries per second)?
2. Scalability and reliability: Not consider extension and scalability. Single DB? Which kind of DB? What if some DB fails?

Level 2.0 Performance

1. Key-Value DB
    1. MySQL is slow (e.g. 10,000 qps)
    2. Key-Value (e.g. 100,000 qps)

2. Encode URL --> KEY
    2. URL --> md5 (128 bits)
    3. Base64 --> 6 bits, 128 / 6 ~ 21 chars > 5 chars
    4. Truncate(md5(URL), 5)
        1. How to resolve collision?

Level 3.0 Scalability & Reliability

1. Multiple Servers (Memcache/DB)
    1. Sharding: hash(URL) % N = Server ID
    2. Standby*

2. Reliability
    2. Replica (cross region, master slave)
    3. Recovery (master: checkpoints, slave: recreate)
    4. Consistency

Level 4.0* Super Star!

1. Utilize a cluster of id generators that reserve chunks of the id space from a central coordinator (e.g. ZooKeeper) and allocate IDs from their chunk
2. How to prevent urls scanned/crawled?
3. How to limit single user RPS, strategy? (Ratelimit, MITBBS example)
4. How to implement the redirect servers?

Ratelimit

Question: Block user when requests more than 10/min or 100/hour or 1000/day ...

Ratelimit

Memcache:

- key: IP (Registration)
  - Username/Email (Login)
  - UserID (Logged In)
- Real key in memcache: key+timestamp(in minute)
- Value: counter
- Expiration: /m /h /d

Memcache介绍

	存储	数据持久性	***key-value***	***expire***	访问速度
memcache	内存	临时，机器重启就没了	是	可以	最快
redis	内存+磁盘	持久，定时从内存***flush***	是	可以	快
nosql database	磁盘	持久	是	可以	慢
sql database	磁盘	持久	不是	不可以	最慢

RateLimit L0

已输入密码登陆为例，要求用户不能在1小时之内登陆出错超过5次。

内存里放一个HashMap，存谁在哪个时间做了某件被限制的事情。

然后……然后……解不了……

无法查询某个用户在10分钟之内到底是否

RateLimit L1

使用Cache, 带上expire信息，设置1小时之后自动销毁。key=event_name +user_id，如”login100”，代表记录了login100在最近的登陆出错总次数。

一旦用户登陆出错，在cache中找到这个计数器，然后+1，并延长销毁时间为一小时之后。
如果发现计数器超过了5，就出错。

缺点：算法有一定正确性的问题。不能完全保证是一小时以内超过5次，会block掉一些正常的访问：如: 12:00登陆出错(计数器=1)，12:59登陆出错(计数器=2)，13:58登陆出错(计数器=3)，14:57登陆出错(计数器=4)，15:56登陆出错(计数器=5)，然后被block。。。但按照之前的规定，1小时之内其实没有超过5次。

RateLimit L2

每分钟为一个bucket（因为一般不会出现以秒为单位的block，如果要以秒为单位的话，就每秒钟一个bucket）

使用Cache，key=timestamp+event+user_id/username

如：”1400_login_username”

value是count。
expire设为2小时。

在用户登陆出错之后。记录这个key到memcache里。并以当前时间为key往前数1-60分钟，在memcache中查询对应的时间上该用户是否有登陆失败的记录。如果有，统计一共登陆失败多少次，超过5次，就告诉系统这个用户需要block。

System Tips+

Cache
Lazy Computing
Read ahead
Asynchronized
Polling
Static memory pool

FAQ

How to find “靠谱” Company?

FAQ

How do I know ready for Interview?

FAQ

How to Get Interview Opportunity ?

FAQ

How to work out Onsite?

FAQ

How to Answer Behavioral Questions?

FAQ

How to Negotiate Offer and Make Decision?

Reference

- http://graphics.wsj.com/billion-dollar-club/
- 硅谷中型Startup有哪些?
- https://cloud.google.com/developers/startups/
- http://aws.amazon.com/solutions/case-studies/start-ups/
- As a software developer candidate at a technology startup, what kind of questions can I ask the interviewer?
- How can I quickly improve my programming skills?