Design a Social Media Application with Extra Premium Features like Job & Professionals Platform like LinkedIn

This High Level Design will cover up the HLD of Instagram / Facebook along with extra services required to build Linked High Level design. So, this HLD covers HLD of Facebook, Instagram and LinkedIn.

Step-1 :- Requirements Gathering :-

① Functional Requirements :-

(a) For User :-

(i) User Management :-

(a) User can register / login
(b) User can edit their profile like name, skills, bio, education, experience, honours and awards etc.
(c) User can control their profile visibility control. (public / private mode)

(ii) Networking & Connections :-

(a) Users can send / accept / reject connection request
(b) Users can search other users on linkedin based on filters like Location, skills, company, industry etc.
(c) Users can join Groups and communities, create and manage it
(d) User can participate in event creation & management

like Webinars, Hiring events, Career fairs etc.

(iii) Job & Recruitment features :-

(a) Users can search and filter jobs.

(b) Easy apply option is provided so one-click apply feature is provided.

(c) Companies can create and manage jobs posts.

(d) Recruiter Dashboard is provided in which job applicants are filter, shortlist or rejected.

(e) Users are given job recommendations based on their profile.

(f) Job Alerts are given to the User

(iv) Post & Content Management —

(a) Users can create/edit/delete post. Post can be in the form of text, image, videos.

(b) User can like, comment, share, report on posts.

(c) Users can write long-form articles for Article publishing and newletters.

(d) User can use multiple reactions on post like 'Like', 'Celebrate', 'Support', 'Insightful' etc.

(v) Analytics & Insights :-

(a) Users are given analysis based on profile views and post impressions.

(b) Job seekers and Recruiters are given Application status Tracking.

(c) User Activity Dashboard is provided in order to provide Engagement score, active hours, analytics,

(d) Job Market Trends is done in order to provide Industry-wise Hiring trends and skill demands Analysis

(vi) Messaging & Communication :—

(a) Users can send direct message to friends.

(b) InMail Messaging is provided in which user can send messages to non-connection users if they have premium

(c) Email & Real-Time Notifications :— Users should be notified on messages, job updates, connection requests etc.

(vii) Monetization & Premium features
(a) Premium Subscription Plans :- Job seekers and Recruiters are provided special features.
(b) Ad Management System :- Companies can promote targeted ads.
(c) LinkedIn Learning Integration :- Personalized learning course recommendations are also provided.

(viii) Customer Support :-
(a) Users can access help center.
(b) Ticketing system is provided in which user can report issues.
(c) User can provide Live Chat & AI chatbot.

For Recruiter :-
(i) User Management :-
(a) Users (Candidates & Recruiters) register / login
(b) Different types of profile is maintained i.e Candidate, Recruiter, Business etc.
(c) Recruiters can create / manage company profile.
(ii) Networking & Connections :-
(a) Recruiters can search & filter candidates.
(b) Recruiters and Managers can do direct messaging.
(c) Recruiters can expand network (send / accept / reject connections).
(iii) Job & Recruitment Features :-
(a) Recruiter can create / edit / delete post.
(b) Recruiters can provide premium boost on job posting.
(c) Applicant Tracking System :-
Recruiters can shortlist / reject candidates
(d) Recruiters can run bulk hiring campaigns.
(e) Job Referrals System :-
Employees can give job referrals to company.
(f) Recruiters are provided Candidate recommendations based on AI-based filtering.

(iv) Post and Content Management :-
(a) Recruiters can post company updates, articles and hiring announcements
(b) Recruiters can view engagement analytics like views, likes, applications etc.
(c) Companies can share company culture, testimonials and case studies
(v) Analytics & Insights :-
(a) Recruiters are provided job post analytics i.e views, clicks and applications data.
(b) Recruiter Dashboard :- Job Applicants overviewing and application status tracking.
(c) Talent Pool Insights :- Industry-Wise hiring trends and skill-demand analysis.
(d) Candidate Profile Insights :- Candidate profile views, skills and match percentage
(vi) Messaging & Communication :-
(a) Recruiters can do direct contact candidates.
(b) Automated Email & Notifications :- Candidates are provided updates of their interview.
(c) Interview Scheduling System :- Meeting schedules is provided to both Recruiters & Candidates.
(vii) Monetization & Premium Features :-
(a) Premium Job Listings :- Option is provided for paid job ads to recruiters.
(b) Recruiter Premium Subscription :- AI-based candidate suggestions and inmail access.
(c) Employer Branding Solutions :- Customized Company page and marketing tools are provided for companies.
(viii) Customer Support :-
(a) Dedicated support team is provided to the recruiters.

(b) Live Chat & AI ChatBot :- Quick hiring assistance is provided to the recruiters.

(c) Fraud Detection :- Mechanism is provided for Fake job posting and Spam filtering.

(C) Additional Features :-

(i) Event & Webinar System

(a) System for Events & Webinars

(b) Companies & Recruiters can host professional events

(c) Users can register in events and can even participate in it.

(ii) Linked Learning & Skill Verification :-

(a) Linked Learning Integration :- Available online courses & certifications.

(b) Skill Assessments & Badges :- Showcases verified skills profile.

(c) Skill-Based Filtering of Recruiters :- Job recommendations based on skills.

(ii) AI - Smart Recommendations

(a) Smart job recommendations

(b) Smart Candidate Recommendations for Recruiters

(c) Smart content feed (personalized post suggestions for users).

Non-Functional Requirements :-

(a) High Availability & Scalability :-

(i) Distributed System :- Load Balancer & Microservices Architecture provide this high availability & Scalability

(ii) Auto Scaling :- On the basis of User Traffic, System can dynamically scale itself.

(iii) CDN (Content Delivery Network) Integration :- Fast load images & videos.

(b) High Performance & Low Latency :-

(i) Database Optimization :- Uses indexing, partitioning, caching techniques.

(ii) Load Balancing :- Distribute server Workload equally.
(iii) Efficient Api design :- Maintain minimum response time.

(c) High Security & Privacy :-
(i) Data Encryption :- Encryption of User data like passwords, messages etc is provided.
(ii) OAuth & Multi-Factor Authentication (MFA) :- Secure login process.
(iii) Fault Detection & Moderation :- Detection of spam accounts & fake job postings.

(d) Fault Tolerance & Reliability :-
(i) Database Replication :- Master-Slave and Multi-Master replication use is compulsory.
(ii) Disaster Recovery Plan :- Implementation of data-backs & failover mechanism.

(e) Maintainability & Upgradability :-
(i) Modular Microservices Architecture :- Deploy & updation of independent services.
(ii) Automated testing & CI/CD pipelines :- Automated deployment pipeline to prevent bugs.

Step-2 :- Prioritisation :-

① Data Population service :-

Purpose :- Initializes the system with necessary test data, ensuring a smooth environment for service validation.

Reason for Inclusion :- Before any service is operational, there must be foundational data for testing and system validation.

Dependencies on Previous services :- None (First service in the system)

Dependencies on Next services :- ID Generator services requires data for ID allocation.

Basically this service is Database setup which is foundation for any System Design

2. ID Generator Service :-

Purpose :- Generates unique identifiers for users, posts and other entities.

Reason for Inclusion :- A unique ID system is essential for tracking users, posts, and transactions efficiently.

Dependencies on Previous services :- Data Population Service (needs data to generate IDs).

Dependencies on Next Services :- User Account Service, Post Management service (for assigning unique IDs).

3. User Account Service :-

Purpose :- Manage user Registration, profile creation and basic account details.

Reason for Inclusion :- Without a user Account, no one can interact with the system.

Dependencies on Previous Services :- ID Generator Service (assigns unique User IDs)

Dependencies on Next services :- User Authentication Service, User Authorization Service (for login and access control)

4. User Authentication Service :-

Purpose :- Handles User Login, Logout and Session management.

Reason for Inclusion :- Ensures users can securely access their accounts.

Dependencies on Previous Services :- User Account Service (authentication requires an existing user account).

Dependencies on Next Services :- User Authorization Service (determine access rights)

5. User Authorization Service :-

Purpose :- Controls access to system features based on user roles (admin, recruiter, job seeker).

Reason for Inclusion :- Ensures secure role-based access to different services.

Dependency on Previous Services: User Authentication Service (only authenticated users can be authorized).

Dependency on Next Services:- All other services that require restricted access (e.g Post Management Service, Recruiter Dashboard).

⑥ Post Management Service:-

Purpose:- Allows users to create, edit, delete etc. post

Reason for Inclusion:- Enables content sharing, a fundamental feature of the platform.

Dependencies on Previous Services:- User Authorization Service (only authorized users can post).

Dependencies on Next Services:- Post Reaction Service require posts to react to.

⑦ Post Reaction Service:-

Purpose:- Tracks likes, shares, and reactions to posts.

Reason for Inclusion:- Facilitates User Engagement and interaction with content.

Dependencies on Previous Services:- Post Management Service requires posts to react to.

Dependencies on Next Services:- Recommendation Service uses engagement data for recommendations.

⑧ Recommendation Service:-

Purpose:- Suggest posts, job listings, and connections based on user activity.

Reason for Inclusion:- Enhances user experience by providing personalized content.

Dependencies on Previous Services:- Post reaction service (relies on engagement data).

Dependencies on Next Services:- Search Service can work in combination with search for better suggestions.

⑨ Spam Detection & Blocking Service:-

Purpose:- Identifies and prevents spammy or abusive content.

Reason for Inclusion:- Ensures Platform safety by filtering out spam.

Dependencies on Previous Service:- Post Management Service (analyzes posts for spam)

Dependency on Next Services:- Fraud Detection service (works alongside fraud detection to ensure security).

(10) Fraud Detection Service:-

Purpose:- Identifies fake profiles, fraudulent activities, and spams.

Reason for Inclusion:- Maintain User trust & Platform integrity.

Dependency on Previous services:- Spam detection & Blocking service can share flagged user data.

Dependency on Next services:- Report service (flagged content can be reported by users).

(11) Report Service:-

Purpose:- Allows users to report inappropriate posts, profiles or behaviours.

Reason for Inclusion:- Enables community moderation by allowing users to report issues.

Dependencies on Previous services:- Fraud Detection service (User can report the fraudulent activities)

Dependencies on Next services:- None directly, but integrates with moderation teams.

(12.) Skill Verification Service:-

Purpose:- Verifies user skills and qualifications through assessments or certifications.

Reason for Inclusion:- Helps recruiters and companies validate candidate skills before hiring.

Dependencies on Previous Services:- User Account service (skills are linked to user profiles)

Dependencies on Next services:- User profile service (verified skills appear on profiles)

(13) User Profile Service

Purpose :- Manages User profile data, including Bio, Experience, and skills.

Reason for Inclusion :- Essential for showcasing a user's professional background.

Dependencies on Previous Services :- User Account Service (profiles belong to registered users).

Dependencies on Next Services :- Search Service (profiles are searchable).

(14.) Search Service :-

Purpose :- Provides a search feature for users, companies, jobs and content.

Reason for Inclusion :- Allows users to find relevant connections and opportunities

Dependencies on Previous service :- User profile service (profiles needs to be searchable)

Dependencies on Next services :- Job Application Page service (searching for jobs).

(15.) Job Application Page Service :-

Purpose :- Displays available job listings based on search criteria.

Reason for Inclusion :- Enables users to find job opportunities

Dependencies on Previous services :- Search service (search helps find job listings).

Dependencies on Next services :- Application Tracking system (tracks applications).

(16.) Application Tracking System :-

Reason for Inclusion :- Helps users keep track of their applications

Purpose :- Track job Applications and updates status (applied, shortlisted and rejected.

Dependencies on Previous services :- Job Application Page Service (requires job applications)

Dependencies on Next services :- Job referral System (recruiters might refer applications)

(17.) Job Referral System :-

Purpose :- Enables users to refer job opportunities to connections.

Reason for Inclusion :- Encourages networking & Job referrals

Dependencies on Previous services :- Application Tracking system (to refer tracked Applications)

Dependencies on next Services :- Recruiter Dashboard service (recruiter manage referrals)

(18) Recruiter Dashboard Service :-

Purpose :- Provides Recruiters with tools to manage job postings and applicants.

Reason for Inclusion :- Allows recruiters to manage talent acquisition efficiently.

Dependencies on Previous Services :- Job Referral system can help recruiters track referrals.

Dependencies on Next Services :- Talent Pool Insights service (analyzes recruiter data).

(19) Talent Pool Insights Service :-

Purpose :- Analysis of Talent pool is provided to recruiters and companies so that insights of hiring trends and available talent is provided to both recruiters and companies.

Reason for Inclusion :- Companies can see & analyze that what kind of candidates they have in their company.

Dependencies on Previous Service :- Recruitment Dashboard service (fetching data from Recruitment Dashboard to generate reports) and Job application page Service and Application Tracking System, Candidates applic-ations and hiring applications give real-time data to this service.

Dependency on Next Services :- Candidate Profile Insights Service (for individual candidates insights, data is processed from talent pool).

One of the main reason for inclusion is that recruiters require analytics for taking hiring decisions. This service will give overview to market trends and candidate availability.

(20) Candidate Profile Insights Service :-

Purpose :- Generate Profile Engagement and interest-based analytics. By this recruiter can see what candidates are active, for what job roles, candidates are usually interested and what is the skill set of candidate.

Reason for Inclusion :- Important in order to provide personalized candidate insights to recruiters and companies.

Dependency over Previous Services :- Talent pool Insights Service (by taking breakdown of the

aggregate data, it gives insight at individual level), Uses profile service (access candidate profile details and updates).

Dependency over next services:- Connection services (these insights are very helpful when recruiters and candidates try to connect each other).

21) Connection Service :-

Purpose:- Manages the features like to connect the users, follow/unfollow the user and accept/reject a connection. Core feature of Professional Networking

Reason for inclusion:- Networking is the crucial feature for every professional platform. It is important so that user can interact with their connections.

Dependencies over previous services:- Candidate Profile insight service (by using insights, connections are suggested)

Dependencies over Next services:- Follow-up service (Notifications and updates are provided when a new connection is created and Messaging service (enable messaging between the connections).

22) Follow-up Service :-

Purpose:- Based on User interactions and activities, notifications and reminders are managed.

Reason for inclusion:- Important to increase engagement on platform so that users are updated and this is possible only with follow-up notifications

Dependences over Previous Services:- When someone made new connections, then for that notification is generated. (connection service), Post Action Service (for any activity on post updates are fetched).

Dependencies over next services:- Messaging service (if any message comes, it will notify).

23) Messaging Service :-

Purpose:- Enables direct messaging between users, recruiters and connections.
Reason for inclusion:- Private communication is a fundamental feature of proffessional platform
Dependencies over Previous services:- Connection service (allow communication between connected users), Follow-up service (message notifications are generated).
Dependencies over Next services:- Customer support Service (for user queries, messaging can be used).

(24) Post Action Service:-

Purpose:- If a reply is given in form of like, comment etc. then notification is generated.
Reason for inclusion:- Notifications is important to engage the users
Dependencies over previous services:- Post Management Service (Post and interactions are tracked), Follow-up service (Notifications are sent).
Dependencies over Next services:- This service do work of notification delivery.

(25) Customer Support Service:-

Purpose:- User complaints, complaints tic tickets are managed by this service.
Reason for inclusion:- Support must be available for users if they face any issue.
Dependencies over previous services:- Messaging service (Support queries are handled through direct message)
Dependencies over Next services:- Automatic Email service (for complaint resolution and ticket updates, email notifications are sent).

(26) Subscription & Billing Service:-

Purpose:- For premium features and billing management.
Reason for inclusion:- Subscription model is necessary for monetization and premium services.
Dependencies over previous services:- Monetization services (tracking of subscription payments are done).

Dependencies over next Services:- Event and Webinar Management service (integrate billing for paid events and webinars).

(27) Monetization & Premium Services :-

Purpose:- Manage programs of learnings and premium job listings, recruiter subscriptions i.e platform paid services.

Reason for inclusion:- Essential for Revenue generation.

Dependencies over Previous Services:- Subscription and Billing service (premium services payments and access are managed)

Dependencies over Next Services:- Ad Management service (Premium Advertisement Services are linked)

(28) Event & Webinar Management Service

Purpose:- management of Online Events and Webinars.

Reason for inclusion:- Important for Learning & Professional growth

Dependencies over Previous services:- Monetization and Premium services (Paid webinars and premium events are integrated).

Dependencies over Next services:- Notification service (reminders and updates of event)

(29) Ad Management Service:-

Purpose:- For serving and tracking advertisements.

Reason for inclusion:- Revenue model is important.

Dependencies over Previous Services:- Monetization Service (Paid ads are managed here).

Dependencies over Next services:- AI/Model service (AI-based targeted ads are enabled).

(30) Notification Service:-

Purpose:- Centralized Management of Platform-wide notifications.

Reason for inclusion:- Essential for increasing platform engagement.

Dependencies over previous services:- Post Action Service, follow-up service

Dependencies over next services:- Automatic Email service

(31) Request Handling Service:-

Purpose:- This service is reponsible for classifying and validating all incoming user requests, ensuring that they are appropriately routed to the right microservices. It performs initial checks, including authentication, authorizations and request format validation.

Reason for inclusion:- This service is critical for maintaining request integrity, ensuring proper routing, and preventing unauthorized or malformed requests from reaching the system.

Dependencies over previous services:- Relies on User Management service for Authentication

Dependency over Next services:- All services except User Account service, User Authorization service, Load Balancer, User Authentication service and zookeeper relies on this service as other service will not be activated until this request Handling service is enabled.

Designing this service at last because we wont on idea that how many types of services will be connected through it. On the basis of that we design this service.

(32) AI/Model service:-

Purpose:- Provides AI-driven capabilities such as predictive analytics, fraud detection, intelligent job recommendations, and automated resume screening. It enhances the platform's intelligence using machine learning models.

Reason for inclusion:- This service enables automation, personalization, and fraud prevention making the platform smarter and improving user experience.

Dependency over Previous service:- Requires data from Talent pool insights service for analyzing recruitment trends. User candidate profile insights service to refine job recommendations. Leverages request handling service to process AI

Dependencies over Next Services :- Hooks with automated Email service to send AI-driven recommendations.
Collaborates with Notification Service to provide AI-based alerts.
Supports Ad Management service by optimizing and targeting using predictive models.

33. Automatic Email Service :-

Purpose :- Handles automated email notifications and alerts for user interactions, such as job updates, interview invites and transactional emails.
Reason for inclusion :- Automates communication, improves engagement and ensures that users receive timely and relevant information.

34. Zookeeper Design :-

Purpose :- Manage distributed system coordination by ensuring microservices register, communicate efficiently and handle failovers. It maintains service availability and health checks.
Reason for inclusion :- Essential for managing a scalable and fault-tolerant microservices architecture, preventing downtime and failures.

35. Load Balancer Design :-

Purpose :- Distributes incoming requests across multiple servers to optimize resource utilisation, ensure high availability, and prevent overload.
Reason for inclusion :- Ensures platform scalability, fault tolerance and optimal resource management, improving performance under heavy traffic.
Dependency over Previous Services :- Works with Zookeeper Design to monitor and manage traffic distribution dynamically.
Relies on the Request Handling service to ensure balanced request routing.

Dependency over Next services:- Works with Notification service to maintain real-time performance for updates.
Supports Ad Management service to optimize and delivery under high traffic.

Zookeeper Dependency over previous services and next services :-

Previous services:- Relies on Request Handling service for distributing requests to the right microservices.
Works with AI/ Model service to manage distributed model execution.
Support Automatic email service to ensure smooth email dispatch.
Next services:- Works with Load Balancer design to distribute load across microservices efficiently and supports notification service to ensure real-time updates in case of failures.

Automatic Email Service Dependency over previous services and next services :-

Previous services:- Works with messaging service and Post action service to send notifications for direct messages and post engagements. User Subscription and Billing service to notify users about subscription renewals. Leverages AI/model service to send personalized recommendations.
Next services:- Works with Notification Service to maintain consistency across email and in-app notifications. Supports load balancer design to ensure efficient email delivery under high traffic

Step-3:- Infrastructure Estimation

(a) User Estimation :-

Let, total users registered on LinkedIn = 1 Billion
20% growth expected upto next year. So
Total Users registered on LinkedIn upto Next year = 1.2 Billion
Let 70% of the Users are Daily active users
Then Daily Active Users upto next year = 0.84 Billion

(b) Data Estimation :-

(1) User Profile Data Estimation :-

Let 100KB be the required data to store basic profile details of a user

User Basic Profile Data Breakdown :-

Component	Estimated Size
Name, Email, Phone	5KB
Profile Photo (compressed)	30KB
Work Experience (3-5 Entries)	20KB
Education Details	15KB
Skills (10-20)	5KB
Connections (Basic Meta data, not Full Profile	10KB
Endorsements & Recommendations	10KB
Awards & Honours	5KB

User Basic Profile Data Storage = 100KB

User Analytics

Let, Profile views on LinkedIn for each user per day = 50

Search Appearences on LinkedIn for each user per day = 20

Post Impressions on LinkedIn for each user per day = 10

Let Each view, appearence and impression take 2 KB of Data, 1 KB of data and 2 KB of data respectively

Then Overall Estimated Storage for an year required :-

Profile view = $50 \times 30\,KB \times 12 = 18\,MB \approx 20\,MB$

Search Appearences = $20 \times 30\,KB \times 12 = 7200\,KB \approx 7.5\,MB$

Post Impressions = $10 \times 30\,KB \times 12 = 3.6\,MB \approx 4\,MB$

Overall 32MB required

So, User Analytics Storage Required = 32 MB

User Activity :-

Let, Each user per year give 500 likes, 50 comments, 20 shares and 5 reposts

Let, 1 Like take 4 Bytes of data, comment take 200 byte, shares take 100 Byte and Reposts take 100 byte

Unique ID take 16 Bytes of Data. Now, Unique ID consist of data first Post ID, Reaction ID of User itself and Link will also be given for the post.

ID data = 32 Bytes (Post-ID & User-Reaction ID)

Link = 150 Bytes (on an average)

Also, User ID that has hosted the post :- 16 Bytes

Like Data :- 32 + 150 + 4 + 16 = 200 Bytes

Comments Data :- 48 + 150 + 200 = 398 Bytes ≈ 400 Bytes

Shares Data :- 48 + 150 + 100 = 298 Bytes ≈ 300 Bytes

Reposts Data :- 48 + 150 + 100 = 298 Bytes ≈ 200 Bytes

So, User Activity Data = (500×200 + 50×400 + 20×300 + 5×300) Bytes

= (100000 + 20000 + 6000 + 1500) Bytes

= 127500 Bytes

≈ 130KB

User Post Data :-

Let each user yearly posts 30 in total

Out of 30 posts, all posts include some text, 80%. post consist of images and 50% of post consist of videos.

Let, avg size of video be 20MB (avg video length 1-2 minutes assumed and 1080 pixel quality is assumed)

Let, image size be 2MB (High resolution PNG/JPEG) assumed.

For text, assuming 500KB of Data (Long messages)

Now,

Storage Required :- 15×20MB + 2MB×24 (S3 storage / file storage)

= 300MB + 48MB

≈ 350MB per user per year

Post data = (8×15 Bytes + 4×200 Bytes + 4×400 Bytes

(8×15 Bytes: Hashtag data, let each user do 8 Hashtags in his post)

(4×200 Bytes: Let user provide 4 links in his post like github link etc.)

(4×400 Bytes: Tag Data in Post Let each user do 4 tags in post)

+ 100 Bytes (for Post ID, User ID & Time Stamp) +

$+500\,KB$ (Text Data) $+ 0.5 \times 30CB + 0.8 \times 300\,B) \times 30$
(Link of Video) (Link of Image)

$= (0.12\,KB + 0.8\,KB + 1.6\,KB + 0.1\,KB + 500\,KB + 150\,KB \times 10^{-3}$
$+ 0.24\,KB) \times 30$

$= 15090.3\,KB$

$\approx 15.1\,MB \approx 16\,MB$

So, Each User post data per year = 16MB

Post Reaction / Post Engagement Data :-
We consider likes, comments, shares & Repost
Let, Each Post on an average receive 100 likes, 20 comments, 5 shares and 2 reposts.

Estimated Data storage :-

Field	Size per Entry	Total Entries (Avg)	Total size
Like/ Reaction	50 Bytes (UserID + Timestamp)	100	5KB
Comment	200 Bytes (Text + User ID + timestamp)	20	4KB
Share	100 Bytes (User ID + timestamp)	5	0.5KB
Repost	150 Bytes (User ID + Timestamp + Post ID)	2	0.3KB
Total Estimate Size (per Post)	-	-	~10 KB

On average, size of each post's engagement = 10 KB
So, Each user post engagement data per year = 30×10KB
= 300KB

Search Data Estimation :-
Let Each user do 10 searches per day
And Each Query take 1KB of data this search query include data like timestamp, user ID, search filters, locations, and device information
So, yearly Data Estimation for each user in search = 10×30×12×1KB
= 3600 KB

Spam, Report and Block Data Estimation:-

Assumption: On average 1% of users report spam atleast once a year

Total Users:- 840 Million

Spam Report Per Year:- 8.4 Million

Data stored per spam report:- 500 Bytes

Reported ID (100B), Reported content ID (100B), Reason (100B), Timestamp (100B), Additional Metadata (100B)

Total Yearly Data for Spam reports = 4.2 GB

Assumption:- 0.5% of users report some content as inappropriate per year.

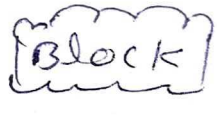
Reports per year:- 4.2 Million

Data stored per Report:- 600 Bytes per report

Reporter ID (100 Byte), Reported User / Post ID (100B), Reason (200B - Multiple categories), Timestamp (100B), Meta Data (100B - Device, Location, etc.)

Data Stored for User Reports:- 2.52 GB

Assumption:- On Average, 5% of users block atleast one user per year.



Blocks per year:- 42 Million

Data Stored Per Block:- 200 Bytes per Block entry

Blocker ID (100B)

Blocked User ID (100B)

Total Yearly Data for Block Dist:- 8.4 GB

Final Storage Estimation = (8.4 + 2.52 + 4.2) GB

= 15.12 GB

≃ 16 GB

Notification Estimation:-

Let Size of Each Notification be 1KB and each user receives 10 Notifications per day

Yearly Data Estimation for each user for notification

= 10 × 365 × 1KB

= 3.65 KB ≃ 3.7 MB

Company Profile Page Data estimation:-

Let, there are 80 million companies upto next year
Each Company Profile Data take 10MB
Then data estimation upto next year = 80M x 10MB
$= 80 \times 10^6 \times 10 \times 10^6$
$= 800TB$

Job Page Data Estimation:-

Let, Each Company announcies vacancies in 10 different Job roles.
And Each Role take 500KB of Data.
Then Job page data Estimation
$= 80M \times 500KB \times 10$
$= 80M \times 5MB$
$= 400TB$

Connection Data Estimation

Let each user 1000 connection and for storing each connection detail, storage required is 10KB
So, Connection Data Estimation = $1.2B \times 1000 \times 10$ KB
$= 12 \times 10^{15}$ Bytes
$= 12PB$

Job Application Page Data Estimation:-

Let, Job Application take 50KB of data and 20% of total users on linkedin apply for Jobs
So, 0.24 Billion users will Apply yearly
Average Job Applications per user per year = 50
(includes resume, cover letter, job details, timestamps etc.)
Hence, Job Application Page Data Estimation over year
$= 50KB \times 50 \times 0.24 \times 10^9$
$= 25 \times 10^3 \times 24 \times 10^9$ Bytes
$= 600TB$

Application tracking system Data Estimation:-

Let 10KB be the size of ATS for each Job Application
So, Estimation = $50 \times 0.24 \times 10^9 \times 10 \times 10^3$ Bytes
$= 120 \times 10^{12}$ Bytes
$= 120TB$

(This includes all Application detail like JobID, Company ID, User ID, Application Status (Hired, Rejected, Review), Timestamps (submitted, updates last modified), recruiter notes, attachments like resume, cover etc.

Event & Webinars, Groups & Communities Data Estimation:-
Total Event & Webinars on LinkedIn per year = 10 million
Average Data stored per event / webinar = 5MB
Total Storage Estimation upto next year = 50TB
Total Groups & communities on LinkedIn = 3 Million
Average data per group = 100MB (Group details, posts, discussions and engagement, member list, permissions & roles)
Total storage estimation for group & communities upto next year = 300TB
Event Data include Event details (title, description, date, time, organizer), attendee list and engagements like comments, reactions and multimedia content like banners, promotional videos, recordings.

Ad Management Service Data Estimation:-
Let, Total Ads shown on Linken (including video, image text) be 20000 per year.
Each Ad Size = 10 MB
Then Data Estimation = 10 MB × 20000
= 200 GB per year

Subscription & Monetization Service:-
Assumption for Estimation :-

Component	Assumption	Estimated Data Size / User or Unit
Total Users	1 Billion (as Before)	
Premium Users	~5% of total = 50 Million users (industry average for premium service)	
Subscription Meta Data	Plan Name, duration, features	2KB per user
Billing Info	Invoices, Transaction logs, method, history	20KB per user annually
Monetization Events (creators)	~10 Million users participate	10 events per user per year = 10KB

Sponsored Earnings/ Creator Mode	Earnings metadata, tracking logs	15KB per monetizing user annually
Total Subscription logs	Timestamped logs of renewals, cancellations, upgrades	5KB per log, 3 logs/year = 15KB

Tabular Summary:-

Feature	Users / Units	Per Unit Size	Total Annual Size (approx)
Subscription metadata	50 Million	2KB	100 GB
Billing & invoices	50 Million	20KB	1 TB
Monetization Events	10 Million	10KB	100 GB
Sponsored Earnings Metadata	10 Million	15KB	150 GB
Subscription logs	50 Million x 3 logs	5KB/log	750 GB
Total			2.1 to 2.2 TB ≈ 2.5 TB

Worst case assuming, we need to store 3TB of Data upto next year

Customer Support service Data Estimation :-
Let 50% of total users on LinkedIn do a customer support request once in an year.
Then, Data Estimation = 0.6 Billion x Size of request
Taking size request as 50KB which covers all request ID, user ID, Timestamp, Subject/Title of issue, message content (detailed query), System Metadata, status tracking (open, closed, etc.), Response logs, screenshots (small attachments if any)

$$= 0.6 \times 10^9 \times 50 \times 10^3 \text{ KB}$$
$$= 30 \text{ TB}$$

Analytics & Insights Service Data Estimation:-

Metric	Estimated Value	Justification
Total Recruiters	~10 Million	only 1% users are recruiters on LinkedIn
Avg Jobs per Recruiter per year	50	Mid-Size + High Activity Recruiters
Average Applications per Job	100	Typical Medium-Sized Job Posts
Average Analytics Records per Application	5	Includes views, status changes, recruiter actions, skill-match, engagement scores
Average Size per Analytics Record	1KB	Optimized storage for numeric / enum fields

① Application Insights

→ Total Records :- 10M recruiters × 50 Jobs × 100 Applications
= 50 Billion Records

Size = 50 Billion × 5 records × 1 KB = 250TB

② Candidate Profile Engagements :-

Assume :- 20% Application profiles are actively viewed.

So, 50 Billion × 20% = 10 Billion

Average 3 engagement entries per user = 30B entries

Size = 30B × 1KB = 30TB

③ Skill Trends / Resume Analysis :-

Assume :- Each Job aggregates top 20 skills

10M recruiters × 50 Jobs × 20 = 10 Billion records

Size = 10B × 0.5KB = 5TB (compressed skill strings)
= 10TB (uncompressed)

④ Recruiter Activity Logs :-

Assume :- Recruiters perform 100 actions / month = 1200/year

10M × 1200 = 12 Billion

Size = 12B × 1 KB ⇒ Size = 12TB

⑤ Summary & Reported Tables (Aggregated)

Pre-computed Analytics for dashboards, reports, trends

Max :- 5TB (aggregated stats, indexes, rollups).

Total Storage Estimation ≈ 312 TB/year
≈ 315TB/year

Talent insights service Data Estimation:-
Provide Recruiters with comprehensive data & Analytics to help understand the talent pool, company insights and market Trends.
Data components & Estimation:-
Total User profiles:- Approximately 1 Billion
Avg. Data per Profile:- 2KB (includes metadata, skills, experience etc.)
Total Storage Required:- 950M x 2KB = 1.9TB
≈ 2TB (for 1 Billion)
≈ 2.4TB (for 1.2 Billion)
= 2.5TB (finalized)

Other components:-
Company data:- information about organizations, size, industry, etc.
Educational Intitution Data:- Details about schools, universities, courses.
Skill and Job Data:- Trending skills job market analysis.
Estimated Total Storage ~ 3.5 TB
Candidate Profile insights Service:-
Provides deep insights into candidate profiles, such as skill matching experience and suitability analysis.
Data Components & Estimations
Per Candidate Profile Data:- 5KB (includes detailed experience, projects, certifications).
Total Storage Required:- 1.2B x 5KB = 6 TB
Other components:- Total Profile View logs: Track which recruiters/companies viewed which profiles.
Engagements data:- Candidate Activities such as posts, likes, comments
Estimated Total Storage:- 8 TB

Fraud Detection Service Data Estimation :—

Per User Daily Activity Logs = 1 KB

Daily Storage Required = 1.2 Billion × 1 KB = 1.2 TB

Yearly Storage Required = 1.2 TB × 365 ≈ 438 TB = 446 TB

Other Components :—

Detailed Logs of Suspicious Activity : Will Require Additional storage.

Machine Learning & Model Data :— includes Training & test datasets

Estimated Total Storage = 450 TB

This service relies on detailed Activity logs from Service.

Overall Data Estimation Completed, now computing the overall data estimate of LinkedIn

Complete Storage Estimation (in PB) = 0.45 + 0.008 + 0.0035 + 0.315 + 0.25 + 0.03 + 0.01 + 0.012 + 0.03 + 0.003 + 0.0002 + 0.3 + 0.12 + 0.6 + 12 + 0.4 + 0.8 + 4.5 + 0.000016 + 0.00000252 + 0.0000042 + 4.4 + 0.36 + 20 + 0.16 + 0.12 + 40 = 84.871723 ≈ 85 PB

Now, Doing Compression (Worst case compression Assumed). compression done in that manner no inconsistency is reacted

Compression Algorithm (for Lossless decomposition) by default taken :— LZ4 (20% compression assumed)

So, Data Estimate = 0.8 × 85 PB = 68 PB

Now, Assuming Replication factor to make system highly available = 3

Total Data Estimation = 204 PB

(c) Cache Estimation :-

Assuming 30% of Total Data is placed in cache Deciding in which service, we require caching

Caching Data Estimation chosen services & their data storage required for caching given below :-

User Account service → PostgreSQL → No Caching

User Analytics → Cassandra → Yes, caching (12 PB of Data)

User Activity Service → PostgreSQL → Yes, caching (0.05PB of Data)

User Post Data → Cassandra → Yes, caching (6 PB of Data)

Post Reaction → MongoDB → Yes, caching (0.2PB of Data)

Search Data → Elastic Search → Yes, caching (1.3PB of Data)

Spam Detection & Block Data → PostgreSQL → Yes caching (21 GB, 100% data taken as data is very small) ← (Used for AI/model service training.

Repost service → PostgreSQL → Yes caching (2.52 GB, 100% data, take to train AI/Model service, this data is very small)

Notification service → Messaging queue (KAFKA) & Cassandra → No caching

Company Profile Page → PostgreSQL → Yes caching (240TB)

Job page data :- PostgreSQL → Yes caching (40TB)

Connection Data :- Neo 4J → Yes, caching (3.6PB)

Job Application Page :- PostgreSQL → Yes, caching (180TB)

Application Tracking system :- PostgreSQL → Yes, caching (40TB)

Groups & Communities :- Mongo DB :- No, caching

Events :- Mongo DB :- Yes, caching (15TB)

Ad Management Service :- No, caching :- Mongo DB

Subscription & Billing service :- PostgreSQL → Yes, caching (1TB)

Customer Support service :- PostgreSQL → Yes, caching (9TB of data) ← (Used by AI/Model service to identify where our system lags and give automated response to increase user interaction/engagement

Analytics Service :- MongoDB → caching yes (9TB + 3TB + 3.6TB)

Insights :- PostgreSQL → No, caching

Talent Insights service :- PostgreSQL → Yes, caching (1TB)

Candidate Profile Insight service :- PostgreSQL → caching, No (2.4TB)

Fraud Detection Service :- PostgreSQL → yes, caching

used by AI/Model service ← (132TB of data)

Taking Replication factor for caching = 2 to increase Availability of our system as well as performance

Total caching data required Storage = 23.8660325 PB
≈ 24 PB

Taking RF (Replication Factor) = 2

Total Cache storage Required = 2 × 24 PB
= 48 PB

So, Total 48 PB of Data required for cache storage without compression. So, Taking LZ4 compression in worst case it compress only 20% of data.

So, Total Cache Storage required = 0.8 × 48 PB

Total Cache Storage Required = 38.4 PB

Total Memory Required / Disk Storage = 204 PB

→ Replication factor taken 3 here

→ Replication factor taken 2 here

Note!- Whenever we need to store image, videos, Gifs etc., we use Distributed File storage like Amazon S3

(c) Traffic Estimation :-

i) Daily Active User Estimation :-

Already Done, DAU = 0.84 Billion (Upto next year)

(ii) API request per second (RPS) :—

Assumed 10% of the users are active at the same time.

Peak concurrent user = 10% of DAU
= 0.1 × 0.84Billion
= 0.084Billion
= 84 Million

Let on an average each user do 10 requests per second

Then, Peak Load = $\frac{10 \times 84 \times 10^6}{60}$

= 14M Requests per second

Average Load = $\frac{60 \times 840 \times 10^6}{24 \times 60 \times 60}$

(Assumed, each day user give total 10 requests per day on an average)

= 583.333K
≈ 584K Requests per second.

(iii) Read vs Write Heavy

Read Heavy Application

Read : 80% and Write : 20%

(d) Compute Power Estimation :—

(i) Backend servers estimation (Application Layer)

Assume 500 Request per second is easily handled by a server

for peak load, Required Number of servers

= $\frac{14M}{500}$

= 28K servers i.e 28000 servers

(ii) Database scaling (Read & Write) Request Handling

Read Request = 0.9 × 14M ≈ 12.6M (assumed worst 90% reads)

Write Request = 0.2 × 14M = 0.28M (Assumed worst 20% writes)

Due to heavy Read system, we need database replica

Assumed 1 DB instance handles 50 K reads easily

Required Read Replica = $\frac{12.6M}{50K}$

= 252 servers

Hence, Read Replica servers = 252

For writes, strong consistency is needed hence we use horizontal scaling

1 strong Write Replica can handle 10 K writes easily, it is assumed

Required Write Instances = $\frac{2.8M}{10K}$

$= \frac{28 \times 10^5}{10K}$

= 280

So, Required DB write instances = 280 servers.

(iii) Caching Layer (Redis / Memcached)

To reduce Database Load, we serve 80% read requests by cache alone.

Cache hit ratio = 80%

Total cacheable requests = 0.8 × 14M × 0.9

= 10.08M requests

Each Redis / Memcached Node can handle 100K RPS assumed.

So, Total Cache Nodes Required = $\frac{10.08M}{100K}$

= 108

Total Cache Nodes required = 108

(iv) Throughput calculation :-

Already computed in DAU estimation i.e Traffic Estimation :- Peak load & Average Load.

(v) Bandwidth Estimation :-

Assume average request size = 50KB

Total Bandwidth Estimation = 14M × 50KB/s

= 700 GB/s

If used compressed algorithm / Techniques, we can reduce it by 50%. i.e approx 350 GB/s

CDN and caching will reduce the load on main servers and we can achieve bandwidth estimation upto 175 GB/s.

(8) Replication Factor :-

Already chosen

For Normal Database :-

Replication factor :- 3

For caching :-

Replication factor :- 2

Step-4 :- Component Identification :-

Services involved in this System :-

- ID generator service
- Load Balancer
- zookeeper
- User Account service
- User Authentication service
- User Authorization service
- Request Handling service
- Search Page service
- Post Management service
- Post Reaction service
- Post Action service
- Recommendation Service
- Spam Detection & Blocking service
- Fraud Detection service
- Repost service
- Skill verification service
- User Profile checking service
- User Analytics Service
- Job Application Page service
- Company Profile Page service
- Application Tracking System
- Job Referral System
- Recruiter Dashboard Service
- Talent Pool Insights Service
- Candidate Profile Insights Service
- Connection Service
- Follow-up Service

Customer Support service

Subscription and Billing service

Monetization and premium service

Event and Webinar Management service

Ad Management service

Notification Service

AI / Model service

Kafka Messaging Queue

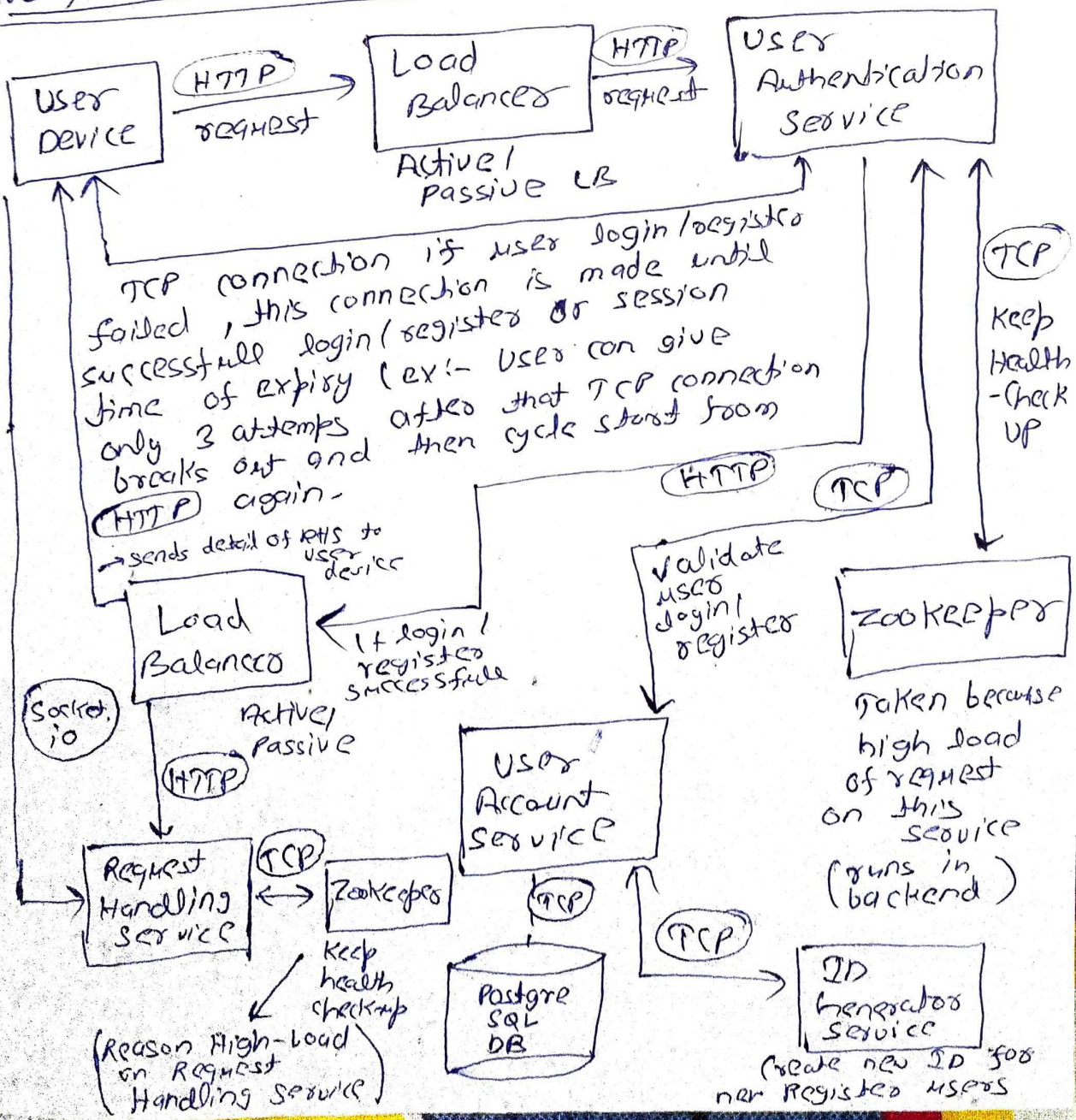
User Activity log service

Automatic email service

Groups & communities service

Analytics & Insights service

Now, How the services interact with each other



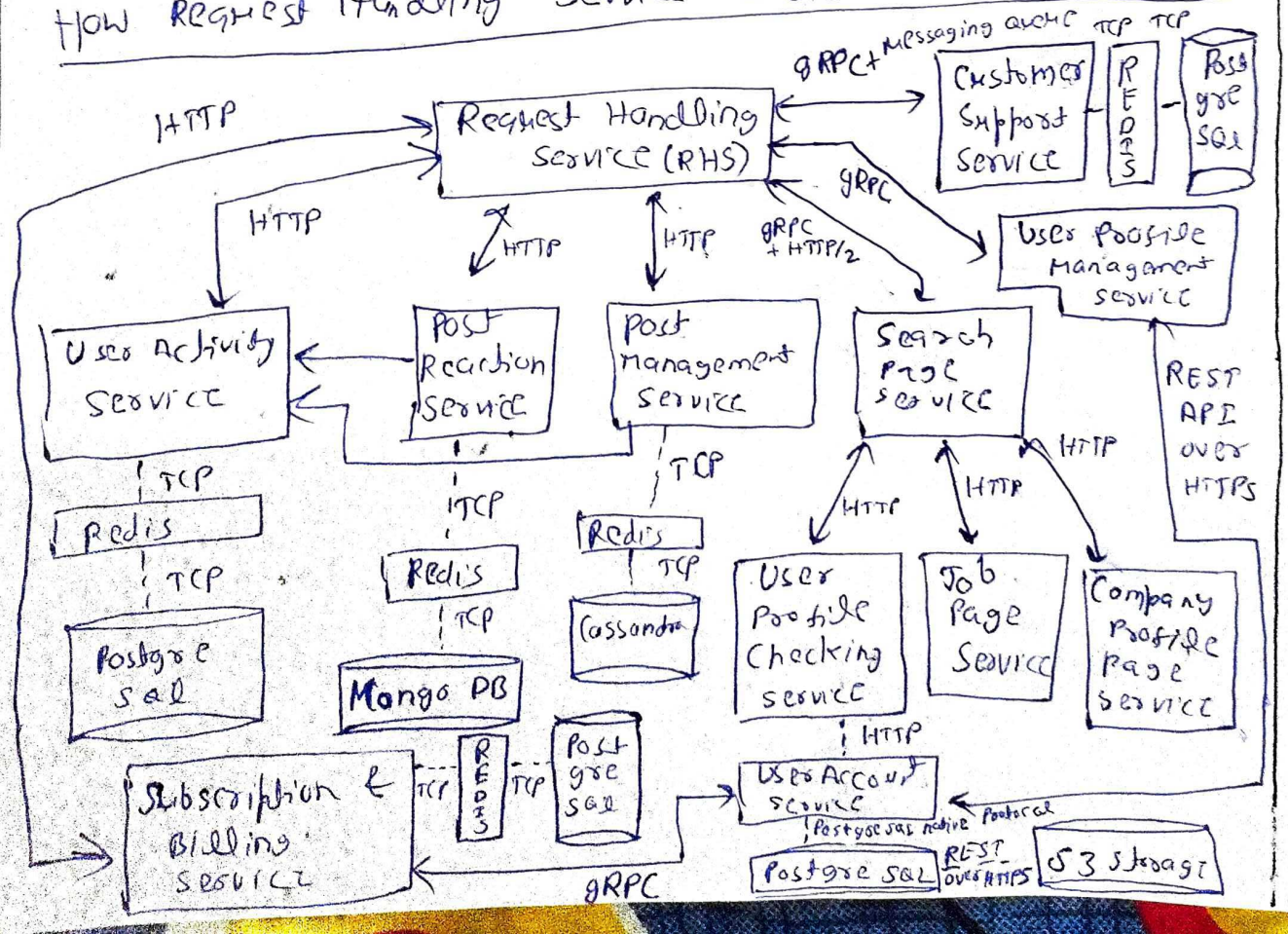
Work flow Explanation :-

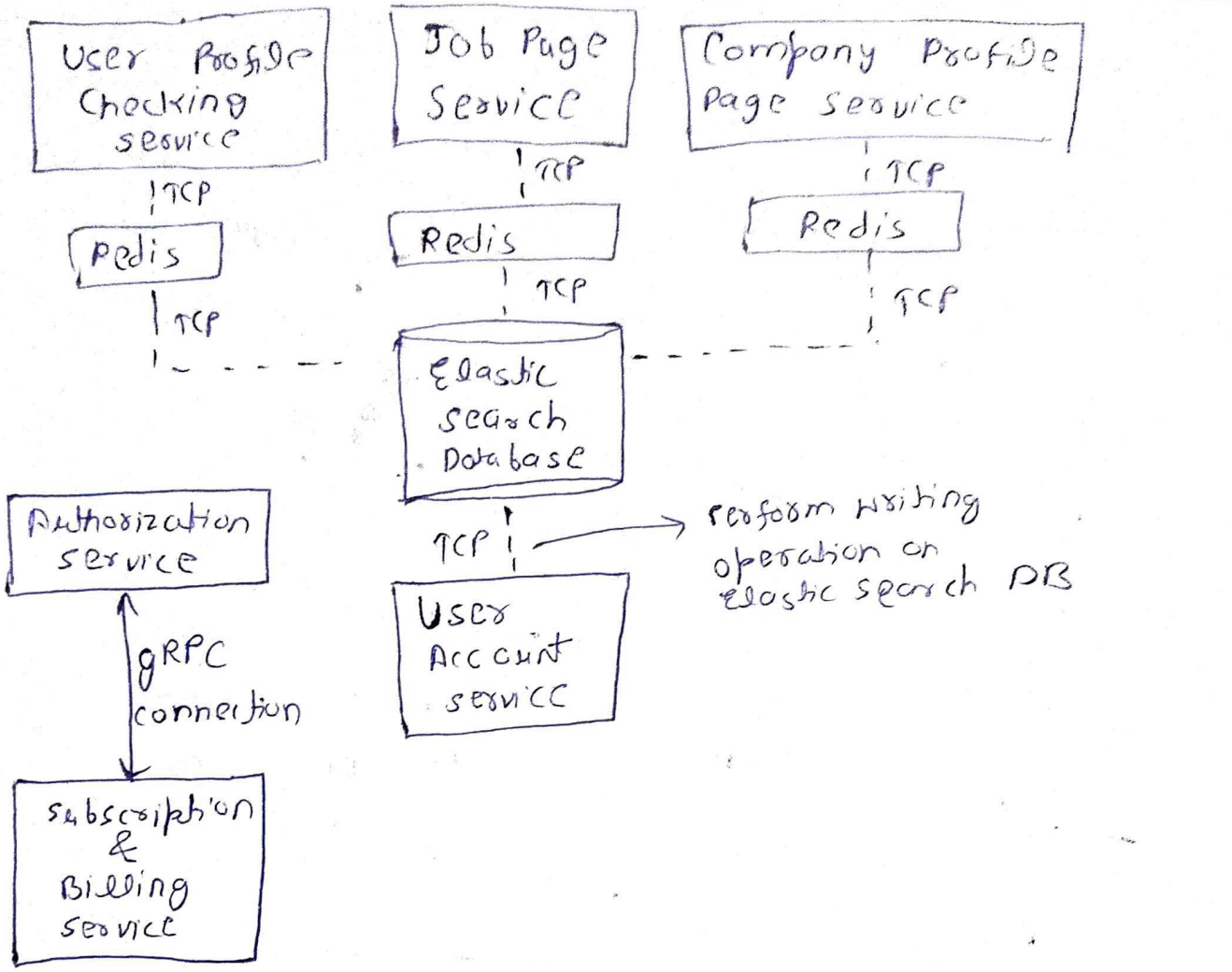
① User device make http call to Load Balancer. This Load Balancer is Hardware Load Balancer

All services are distributed in nature. If Load Balancer not explicitly mentioned by-default we are using software Load Balancer at that time.

② Now, Hardware Load Balancer make HTTP call to the Authentication Service. Authentication service will be having high load so it is conneded with zookeeper in Backend (keeping health check-up and replacing it with another sever if server goes down) and hence Authentication Service make call to the User Account Service to do login/register. If they are successfull, Authentication service make call to Hardware Load balancer that connect to request handling page of LinkedIn and establish socket.io connection and goes out, otherwise if login/register fail a TCP connection is made b/w user device & Authentication service to decrease latency time upto certain period/attemp to login/register or session time.

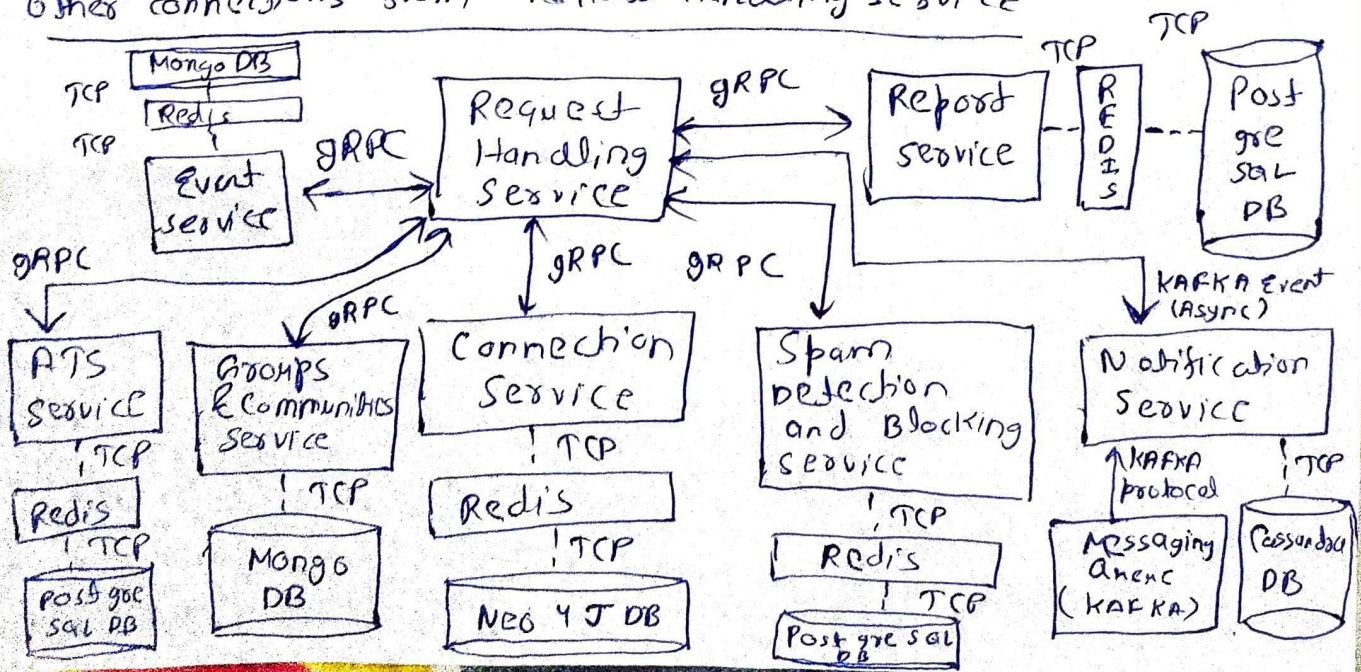
How Request Handling Service is distributing the requests



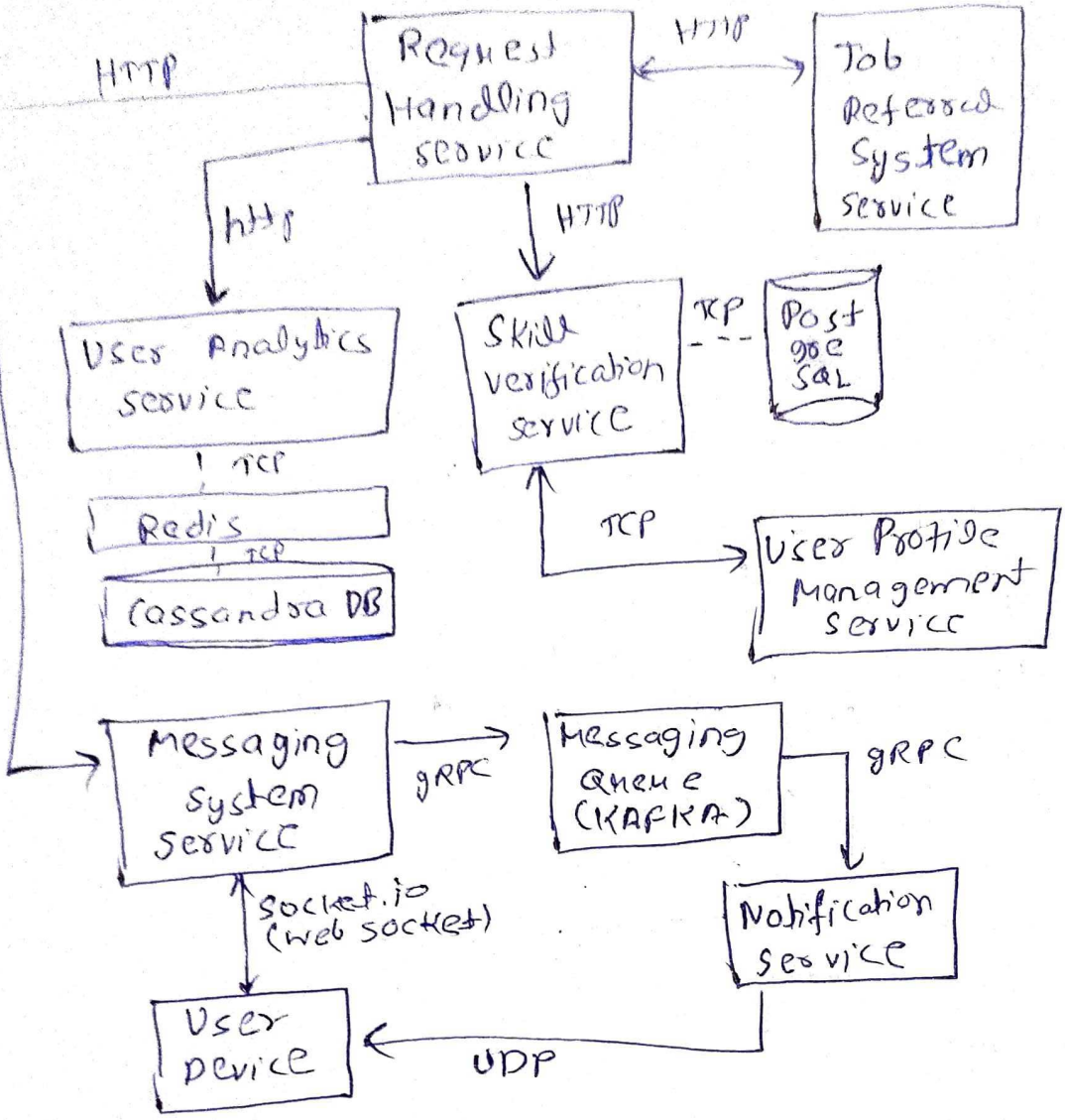


Breakdown of communication b/w Request Handling service & Customer Support service:-

Other connections from Request Handling service



Other Connection from Request Handling service :-

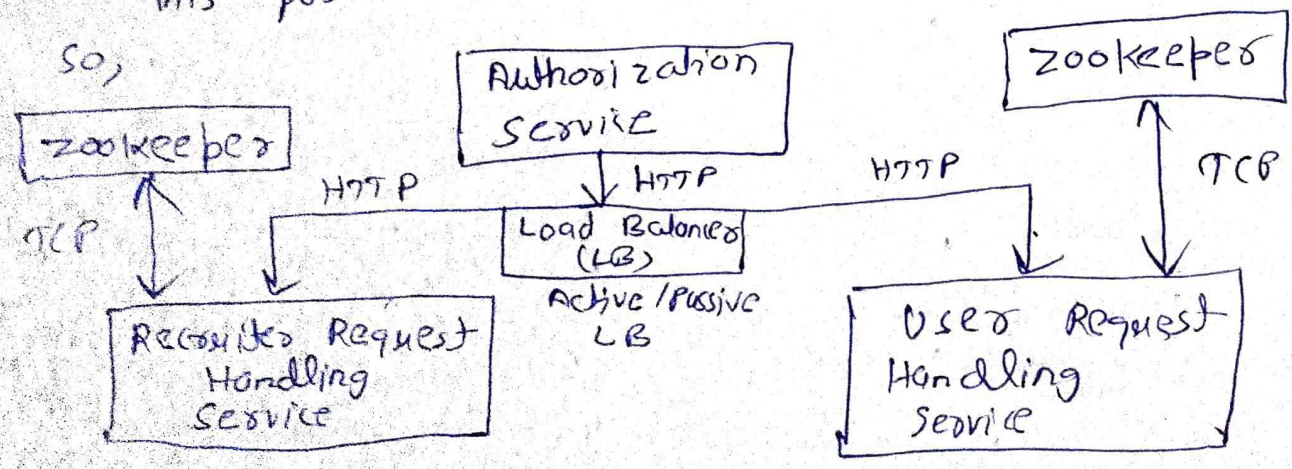


Important Planning

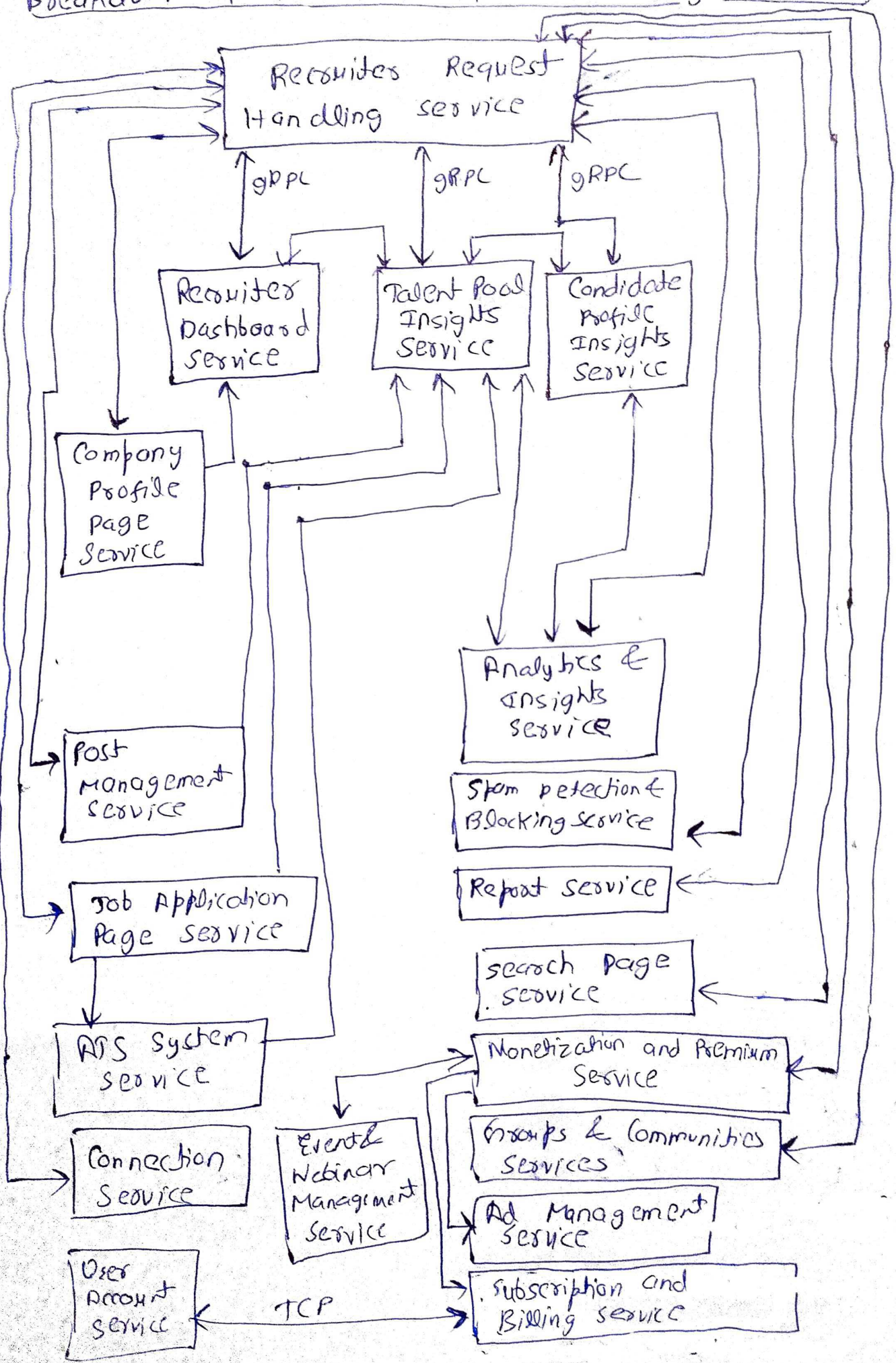
Here, we see that already a Bulk load on Request Handling service is coming. So we are seperating the Request Handling service into two parts :- (i) Recruiter Request Handling service
(ii) User Request Handling service.

This position is done by Authorization Service

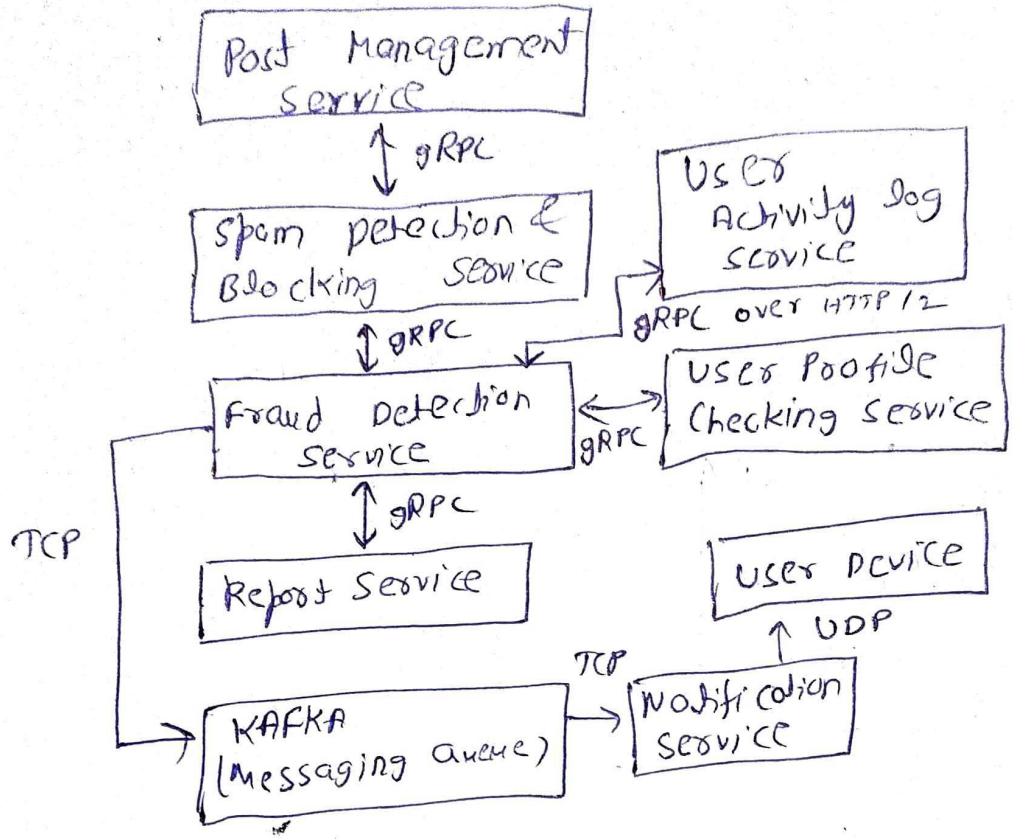
So,



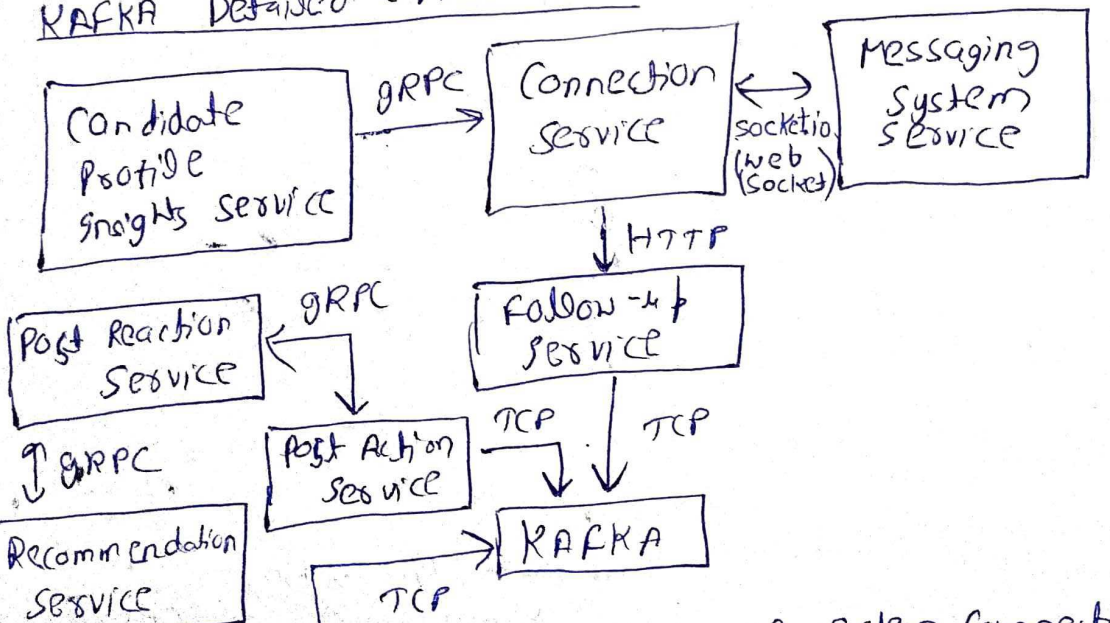
Breakdown of Recruiter Request Handling Service



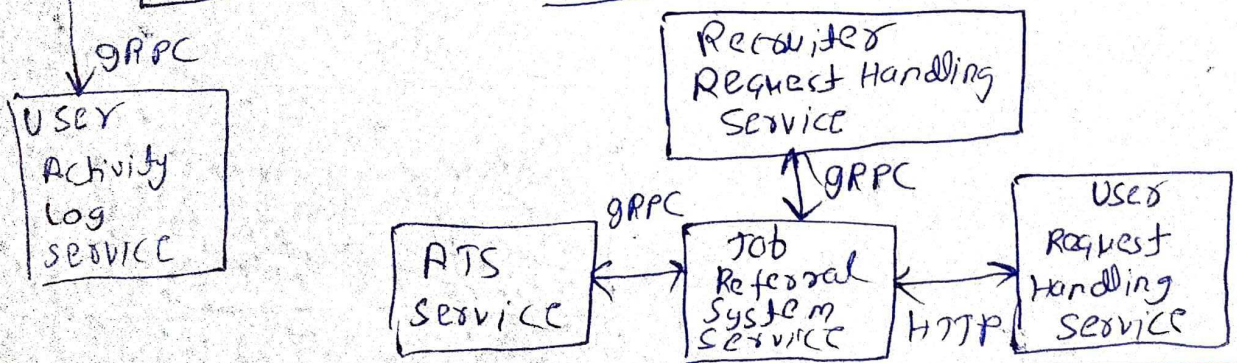
More Detailed Connection of Fraud Detection service



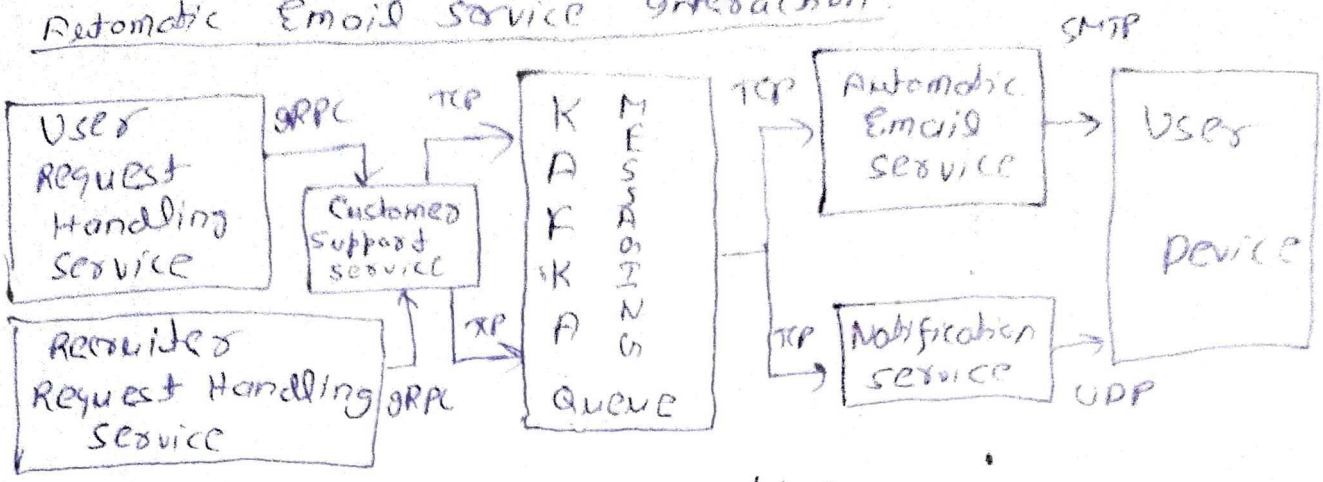
KAFKA Detailed Connections :—



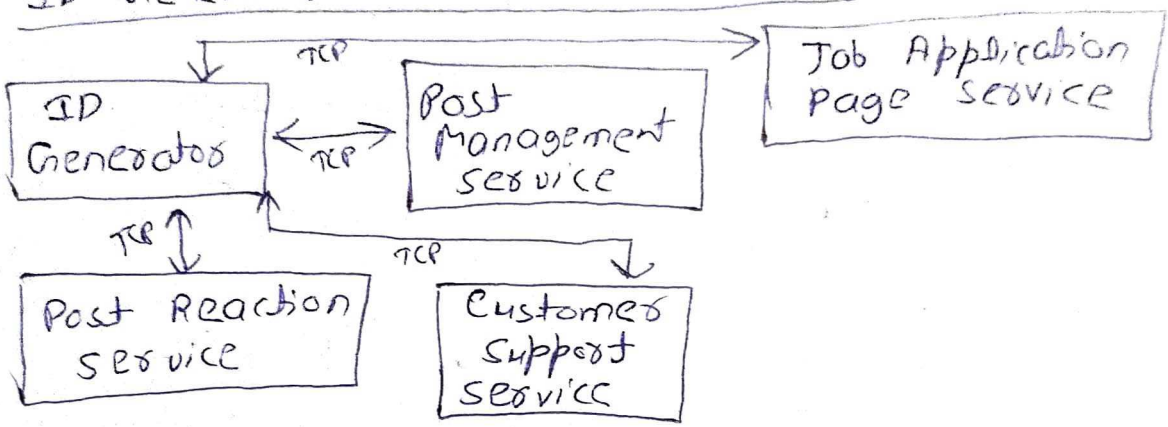
Job Referral System Connection



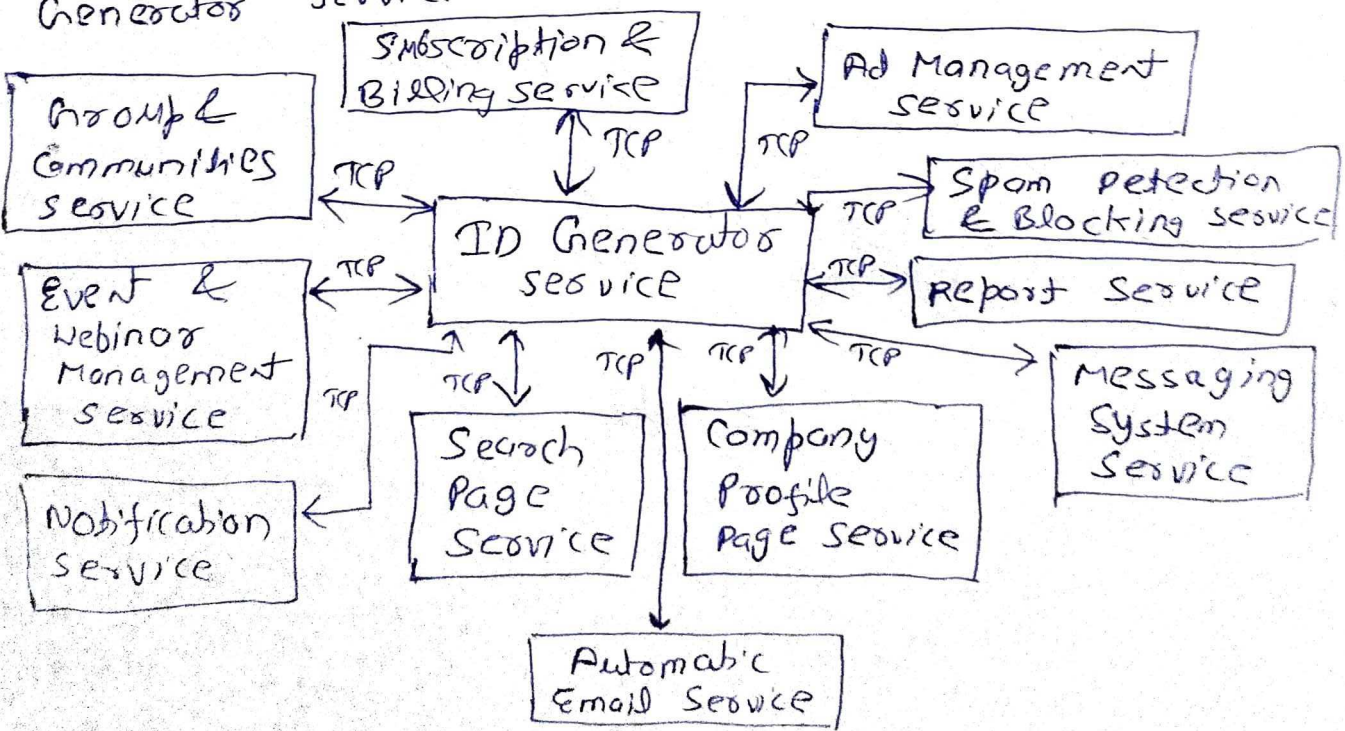
Automatic Email service Interaction :-



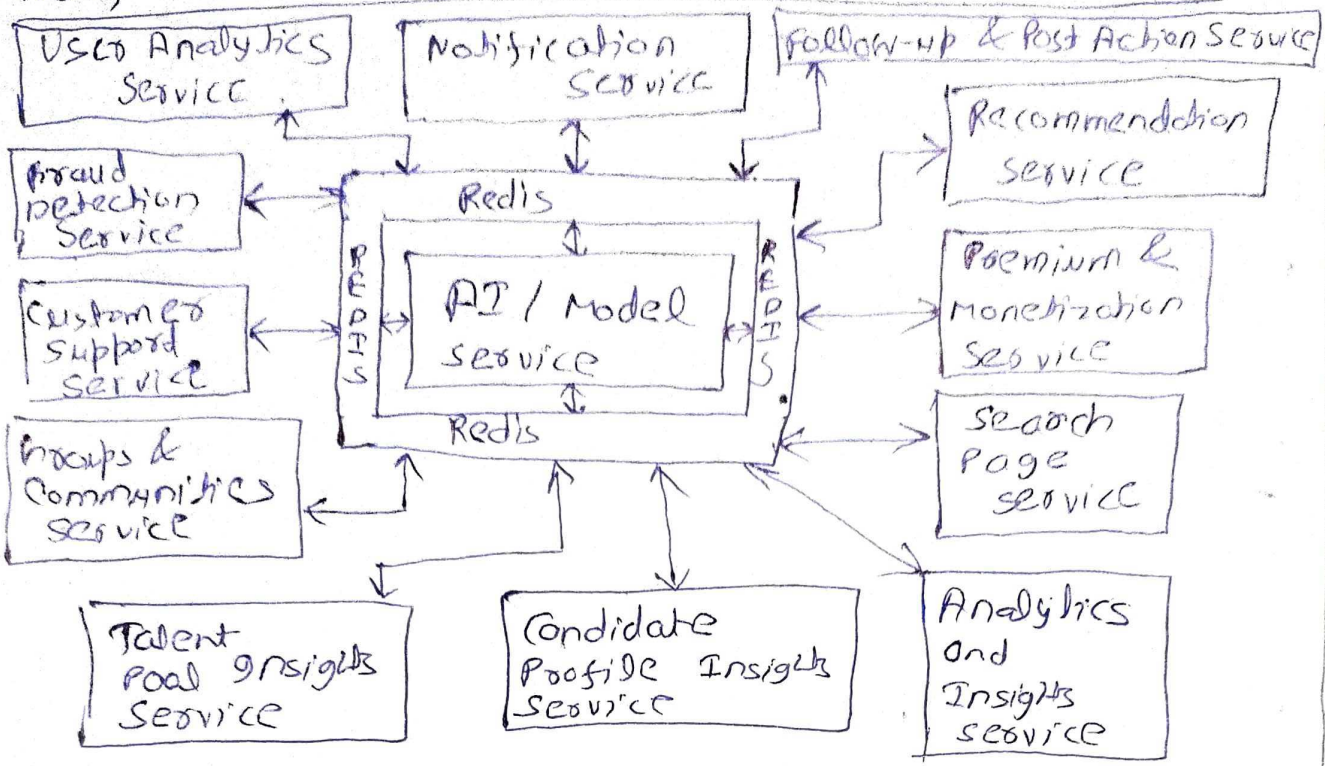
ID Generator Service Interaction



Note :- Here you are thinking that ID Generator service is connected to different service. But it is not, we maintain Different ID Generator service for Difference services. So that, each service interact with it's unique ID Generator service.



Now, AI / Model service Integration into System:-



All connections here are TCP connections

Noted Point :- For diagram simplicity, we have not shown databases. Databases selection is already done in cache estimation part of infrastructure estimation. Note :- All Redis to Database & service to Redis & service to Database, we used TCP communication protocol in order to avoid Data inconsistency. Now, overall discussion of All connection & interact of different services of LinkedIn is completed.

Distributed Database strategy

Master-Master :- Notification Service, Group & Communities Service, (Chat service (Messaging system service), Automatic Email Service, Analytics & Insights service, KAFKA Messaging Queue, Post Reaction Service, (Write-Heavy Application services uses this strategy) -ies

Post Action Service, Fraud Detection Service, Talent Pool Insights Service, User Analytics Service, Zookeeper, User Activity log Service, Recommendation Service, AI/Model Service, Request Handling Service Write Replica,

Master-slave:- (Used for Read-Heavy services) User Profile Checking Service, User Activity Log Service, Post Management Service, Search Page Service, Job Application Page Service, Spam Detection and Blocking Service, Report Service, User Profile Management Service, User Account Service, User Authentication Service, User Authorization Service, Candidate Profile Insights Service, 2D generator service, Event and Webinar Management service, Subscription and Billing Service, Job Reference System, Application Tracking system Service, Company Profile Page Service, Recruiter Dashboard Service, Follow-up Service, Ad Management service, Read Replica Database Servers, Cache Nodes, etc.

Partitioning or Sharding selection

Sharding is used along with range-based partitioning.

Stateful vs Stateless System:-

Stateless Chosen because of:-

(a) Scalability (b) Microservices Architecture (95% of Architecture is based on microservices)

(c) Load Balancing

(d) Failure Handling.

Noted Point:-

95% Architecture is Microservices only 5% architecture is service oriented Architecture. SOA (service oriented Architecture) is used in AI/model service where AI/model service do not maintain its own database it fetches data from

redis layer connected to different database of different services. Based on that most frequent data, it trains it's model and return invaluable result/response for different services.

Also, Elastic search DB is shared with User Account service and User Profile checking service. So, this is also an example of Service Oriented Architecture

Sequence of Load Balancing Algorithm

Round Robin
↓
Least connection
↓
Weighted Round Robin

RPS (Request per second) increases, then we switch our load Balancer algorithm in this manner.

Noted Point:- In Prioritization, we have discussed already how the service depend over previous and next services in detail so please revisit those points in order to understand Overall final Design

Communication Protocol & L4/L7 Load Balancing :-

Service 1	Service 2	Communication Protocol	Load Balanced L4/L7
User Device	Load Balancer	HTTP	L7
Load Balancer	User Authentication Service	HTTP	L7
User device	Authentication service	TCP	L4
User authentication service	User Account Service	TCP	L4

Service 1	Service 2	Communication Protocol	Load Balancer L4 / L7
User Authentication Service	zookeeper	TCP	L4
Request Handling service	User Device	socket io	L7
Request Handling service	Load Balancer	HTTP	L7
Any Service	ID Generator Service	TCP	L4
User Account Service	PostgreSQL	TCP	L4
Request Handling Service (User or Recruiter)	zookeeper	TCP	L4
RHS (User or Recruiter) (Request Handling service)	Subscription & Billing Service	HTTP	L7
Request Handling Service (User or Recruiter)	User Activity Service	HTTP	L7
Request Handling service (User or Recruiter)	Post Management Service	HTTP	L7
Request Handling Service (User or Recruiter)	Post Reaction Service	HTTP	L7
RHS (Request Handling Service) for both User or recruiter	Search Page Service	gRPC over HTTP/2	L7
RHS (User or Recruiter)	User Profile Management Service	gRPC	L7
RHS (User or Recruiter)	Customer Support Service	gRPC	L7
Search Page Service	User Profile Checking service	HTTP	L7
Search Page Service	Job page Service (Job page Application Service)	HTTP	L7
search page service	Company Profile Page service	HTTP	L7
User Account Service	Subscription & Billing service	gRPC	L7
User Profile Management Service	User Account Service	REST API over HTTP	L7
PostgreSQL	S3 Storage	REST over HTTPS	L7
RHS (User or recruiter)	Report Service	gRPC	L7
Customer Support service	Message QUEUE (KAFKA	TCP	L4

Service 1	Service 2	Communication Protocol	Load Balancer L4 / L7
Messaging Queue (KAFKA)	Automatic Email Service	TCP	L4
KAFKA (Messaging Queue)	Notification Service	TCP	L4
Automatic Email Service	User Device	SMTP	L4
Notification Service	User Device	UDP	L4
Post Management Service	Spam Detection and Blocking Service	gRPC	L7
Spam Detection and Blocking Service	Fraud Detection Service	gRPC	L7
Fraud Detection Service	User profile Checking Service	gRPC	L7
Fraud Detection Service	Report service	gRPC	L7
Fraud Detection Service	User Activity log service	gRPC over HTTP/2	L7
Fraud Detection Service	KAFKA (messaging queue)	TCP	L4
Messaging System Service	User Device	Socket.io (web-sockets)	L7
KAFKA (Messaging Queue)	Any Service	TCP	L4
Post Reaction Service	Recommendation Service	gRPC	L7
Recommendation Service	User Activity log Service	gRPC	L7
Candidate Profile Insight Service	Post Reaction Service	gRPC	L7
Connection Service	Follow-Up Service	HTTP	L7
Connection Service	Messaging system Service	socket.io	L7
RHS (Recruiter)	Job Referral System Service	gRPC	L7
RHS (User)	Job Referral system Service	HTTP	L7
Job Referral System	ATS Service	gRPC	L7

Service 1	Service 2	Communication Protocol	Load Balancer (L4 / L7)
RHS (Recruiter)	Recruiter Dashboard Service	gRPC	L7
RHS (Recruiter)	Talent Pool Insights Service	gRPC	L7
RHS (Recruiter)	Candidate Profile Insights Service	gRPC	L7
RHS (Recruiter)	Analytics & Insights Service	gRPC	L7
RHS (Recruiter)	Monetization & Premium Service	gRPC	L7
RHS (Recruiter)	Groups & Community Service	HTTP REST	L7
RHS (Users)	Groups & Community Service	HTTP	L7
Premium & Monetization Service	Subscription & Billing Service	TCP	L4
Premium & Monetization Service	Ad Management Service	TCP	L4
Premium & Monetization Service	Event & Webinar Management Service	TCP	L4
RHS (Recruiter)	Company Profile Page Service	HTTP REST	L7
RHS (Recruiter)	Job Application Page Service	HTTP / REST	L7
Job Application Page Service	ATS System Service	HTTP	L7
ATS Service	Talent Pool Insights Service	HTTP	L7
Job Application Page Service	Talent Pool Insights Service	HTTP	L7
Post Management Service	Talent Pools Insight Service	HTTP	L7

Service 1	Service 2	Communication Protocol	Load Balancer L4/L7
Recruiter Dashboard service	Company Profile Page service	HTTP	L7
Recruiter Dashboard Service	Talent Pools Insights service	HTTP	L7
Talent Pools Insight service	Candidate Profile Insights service	HTTP	L7
User & Analytics Service	Candidate profile Insights Service	HTTP	L7
User & Analytics service	Talent Pools Insight service	HTTP	L7
AI/model Service	Redis	TCP	L4
Notification Service	KAFKA (Messaging Queue)	KAFKA Protocol	L4
RHS (Recruiter or User)	Job Referral System Service	HTTP	L7
RHS (Recruiter or User)	Skill Verification service	HTTP	L7
Skill Verification Service	User Profile Management service	TCP	L4
Messaging system Service	Messaging Queue (KAFKA)	gRPC	L7
RHS (User or Recruiter)	Messaging System service	HTTP	L7
Authorization Service	Subscription & Billing service	gRPC	L7
RHS (User)	ATS Service	gRPC	L7
RHS (User or Recruiter)	Notification Service	Kafka event (async)	L4
RHS (User or Recruiter)	Connection Service	gRPC	L7
RHS (User)	Event & webinars Management Service	gRPC	L7
RHS (User)	Spam Detection & Blocking service	gRPC	L7
RHS (User)	User Analytics Service	HTTP	L7
Rest services	Redis	TCP	L4
Rest services	Database (Any)	TCP	L4
Redis (Caching)	Any Database	TCP	L4

Note:- We have not detailed out the components of messaging system service. Please note Design a Messaging system service like Whatsapp, Messenger etc. is itself a High Level Design Problem and it is already done/solved in detail, that's why this service detailing is skipped you can get this Design from High Level Design Repository of Yatharth Kumar Saxena