1\_USER SERVICE

our **User Service** is quite basic → only CRUD and role-based filtering. But we can add many **realistic, resume-boosting features** to make it more powerful and aligned with your Dairy Farm app.

**🛠️ Possible Features for User Service**

**📊 3. User Earnings & Reports**

* Add endpoints:
  + GET /users/{id}/earning → shows earnings for a single user.
  + GET /users/company/{companyId}/earnings → show total salaries paid.
* Owners can use this to check **profit vs worker salaries**.

**📅 4. User Activity Tracking**

* Add an activityLog table linked to User.
* Store logs like:
  + “User logged in”
  + “User marked attendance”
  + “User added cow health record”
* GET /users/{id}/activity → shows activity history.

**🌍 6. Location-based Filtering**

* Since each user has a location:
  + GET /users/location/{location} → find users by village/city.
* Useful for managing distributors & workers in different areas.

**📊 9. Dashboard APIs**

* Aggregate APIs for frontend:
  + Total users per role (owner, manager, worker, etc.) → GET /users/stats/roles.
  + Worker count per company → GET /users/stats/company/{companyId}.

**🎯 Summary**

So far we only have **basic CRUD + role-based hiding of earnings**.  
We can extend with:

1. Earnings reports
2. User activity tracking
3. Location-based filtering
4. Dashboard stats

2\_COMPANY SERVICE

**Company Service 🏢** is already our **central aggregator/dashboard**, but we can make it far more powerful for your resume project.

**✅ Current Features**

* Store company info (name, owner, area, workers, earning).
* Fetch aggregated **Dashboard** (Finance, Milk, Users, Distributor, Health).
* Role-based earning visibility (only Owner sees it).
* Basic notification trigger (loss → alert).

**🚀 Extra Features We Can Add**

**📊 1. Advanced Dashboard KPIs**

* Milk productivity per cow.
* Worker efficiency (milk production / worker count).
* Cost per liter of milk (from Finance + Milk).
* Pending payments % (from Distributor).
* Health ratio (healthy vs sick animals).

**📈 2. Trend Analytics**

* Compare revenue vs expenses over months.
* Milk production trends (daily, weekly, monthly).
* Worker attendance vs production trends.
* Show graphs on frontend (via aggregated API).

**🧾 3. Report Generation**

* Generate PDF/Excel **company reports** (monthly/annual).
* Download option for owner.
* Auto-email reports using Notification Service.

**⏰ 4. Scheduled Insights**

* Daily 7 AM → summary of yesterday’s production, sales, expenses.
* Weekly → profit/loss update.
* Uses Spring Scheduler + Notification Service.

**👥 5. Multi-Company Support**

* Owner can create multiple companies (e.g., two dairy farms).
* Dashboard can switch between companies.

**🔔 6. Intelligent Alerts**

* Profit < X → alert.
* Worker absenteeism > Y days → alert.
* Sick animals > Z% → alert.
* Stock below threshold → alert.

**🌐 7. API Gateway Integration (for dashboard)**

* Company Service calls all services via gateway, not direct URLs.
* Easier to scale & deploy in cloud.

**🧑‍🤝‍🧑 8. Role-Based Dashboards**

* Owner → sees **everything**.
* Manager → sees production + finance (not full earnings).
* Vet → sees animal health + breeding stats.
* Worker → sees personal tasks, attendance.

**📉 9. Forecasting (Future AI Upgrade)**

* Predict milk yield for next month.
* Forecast expenses.
* Profitability projection per cow.

3\_ANIMAL SERVICE

**Animal Service** 🐄 can:

* Add animals
* View animals by company
* Update animal health

But in a **real dairy farm management system**, we can add a lot more.

**🛠️ Extra Features for Animal Service**

**🐮 1. Detailed Animal Profile**

* Add attributes like: photo, tag number, vaccination records, insemination history.
* API: GET /animals/{id}/profile → returns full profile.

**📈 2. Production History (per cow)**

* Track daily milk production logs.
* Store shift (morning/evening), fat %, SNF %.
* API:
  + POST /animals/{id}/milk → log production.
  + GET /animals/{id}/milk/history → return graph data.

👉 Note: This could also be handled in **Milk Service**, but storing per-animal data here makes reporting easy.

**❤️ 3. Health & Medical Records**

* Store vet visits, symptoms, medication, vaccination schedule.
* Integration with **Notification Service** for reminders.
* API:
  + POST /animals/{id}/health-record → add record.
  + GET /animals/{id}/health-history → show all medical history.

**🧬 4. Breeding & Reproduction Tracking**

* Heat cycle tracking (start date, expected ovulation).
* Pregnancy status, calving dates.
* Integration with **Breeding Service**.
* API:
  + POST /animals/{id}/breeding-log
  + GET /animals/{id}/breeding-history

**🏷️ 5. Animal Grouping & Categories**

* Group animals by breed, age, lactation cycle.
* Example: “Milking cows”, “Dry cows”, “Calves”.
* API: GET /animals/company/{companyId}?status=Milking

**📊 6. Performance & Productivity Reports**

* Calculate average milk production per breed.
* Identify low-performing cows.
* API: GET /animals/company/{companyId}/stats

**📅 7. Maintenance & Cost Tracking**

* Record feed cost, medical cost per cow.
* API: GET /animals/{id}/expenses → shows expenses tied to that animal.

**📸 8. Animal Photo Upload**

* Upload cow images for easy identification.
* Store in DB or cloud (S3, Firebase).

**🔔 9. Alerts & Notifications**

* Notify owner when:
  + Cow is due for vaccination.
  + Cow’s production drops suddenly.
  + Heat cycle expected.

**📍 10. Location Tracking (IoT Integration)**

* If cows have RFID/GPS collars → log movement, health sensors.
* Not essential for now, but great for future expansion.

4\_BREEDING SERVICE

**Breeding Service 🧬** right now can:

* Record **Heat / Insemination / Pregnancy Check / Calving** events
* Fetch breeding history per animal
* Trigger notifications for pregnancy check & calving

That’s **basic**, but in a real dairy system, we can add much more power 🚀

**🛠️ Features to Add in Breeding Service**

**📅 1. Cycle Tracking & Prediction**

* Auto-calculate **next expected heat cycle** (avg 21 days).
* Alert if heat cycle is missed → possible fertility issues.
* Maintain a calendar for each animal.

👉 API: GET /breeding/animal/{id}/next-cycle

**🧪 2. Pregnancy Management**

* Record **pregnancy test results** (positive/negative).
* Estimate **expected calving date** (≈280 days).
* Remind owners for checkups during pregnancy.

👉 API: GET /breeding/animal/{id}/pregnancy-status

**👶 3. Calving Records**

* Log calf details (gender, weight, health).
* Auto-insert calf into **Animal Service** as a new entry.

👉 API: POST /breeding/animal/{id}/calf

**📊 4. Breeding Analytics**

* Pregnancy success rate per company.
* Calving interval (time between two calvings).
* Fertility statistics by herd.

👉 API: GET /breeding/company/{companyId}/stats

**🩺 5. Integration with Health Service**

* If a breeding failure occurs, link health reasons (e.g., reproductive infection).
* Auto-sync vet reports into breeding history.

**🧑‍⚕️ 6. Vet Assignment**

* Assign which vet handled insemination or calving.
* Generate vet performance reports.

👉 API: GET /breeding/vet/{vetId}/success-rate

**🔔 7. Alerts & Notifications**

* Notify owner on:
  + Missed heat cycle.
  + Pregnancy confirmation.
  + Upcoming calving.
* Auto reminders via **Notification Service**.

**🧾 8. Cost Tracking**

* Store cost of insemination, pregnancy tests, and calving.
* Send this data to **Finance Service**.

👉 API: GET /breeding/company/{companyId}/expenses

**🌐 9. AI/ML Fertility Predictions (Future)**

* Use machine learning on historical data to predict fertility success chances.
* Suggest best insemination windows.

5\_DISTRIBUTOR SERVICE

**✅ Current Features**

* Add distributor (customer/vendor).
* Record deliveries (milk quantity, shift, price).
* Track pending balance per distributor.
* Mark payment received → reduces balance + adds income to **Finance Service**.
* Send notifications (deliveries, payments).

**🚀 Extra Features We Can Add**

**🧾 1. Invoice Generation**

* Auto-generate **invoice PDFs** for deliveries.
* Include distributor name, date, quantity, price, pending balance.
* Email invoices (future upgrade).

**📊 2. Delivery Reports**

* Total milk delivered per distributor (daily/weekly/monthly).
* Compare deliveries vs payments received.

**📉 3. Pending Payment Alerts**

* If balance > X days → notify owner.
* Example: “Distributor A pending ₹25,000 for 15 days.”

**📅 4. Delivery Scheduling**

* Pre-schedule deliveries for regular customers.
* Example: "Every day 50 liters at 7 AM to Distributor B."

**💰 5. Dynamic Pricing**

* Different distributors may have different rates.
* Support seasonal/dynamic pricing (summer/winter milk demand).

**🧑‍🤝‍🧑 6. Role-Based Access**

* Owner → add distributors, record deliveries, mark payments.
* Manager → record deliveries.
* Worker → only view assigned deliveries.

**🌐 7. Integration With Finance**

* Already logs income → extend to also **auto-generate monthly statements**.

**📍 8. Geo-Tracking (Future Upgrade)**

* Store distributor delivery locations.
* Later integrate maps for delivery routing.

6\_FINANCE SERVICE

**Finance Service (basic features)**

* Add income & expense transactions.
* Fetch all transactions.
* Profit/Loss calculation.
* Alerts for high expenses (via Notification Service).

**🚀 Features We Can Add to Finance Service**

**🧾 1. Invoices & Bills**

* Generate invoices for milk sales.
* Upload bills for expenses (vet, equipment, feed).
* Export as PDF/Excel.

**📊 2. Financial Reports**

* Monthly/quarterly income vs expense reports.
* Category-wise breakdown (feed, medicine, salary, equipment).
* Graphs for dashboard.

**📅 3. Recurring Transactions**

* Auto-log salaries, monthly feed purchases, rent.
* Use Spring @Scheduled jobs for automation.

**🔔 4. Alerts & Thresholds**

* Alert if expenses > X% of income.
* Alert if revenue drops below last month’s average.

**👥 5. Role-Based Finance Access**

* Owner → full financial details.
* Manager → limited access.
* Workers → no finance access.

**🏦 6. Bank/UPI Payment Tracking**

* Record mode of payment (cash, UPI, bank transfer).
* Integration with Distributor Service for pending payments.

**📉 7. Profitability Analytics**

* Feed cost vs milk income comparison.
* Profitability per cow/company.

**🌐 8. Integration with Other Services**

* Inventory → purchases auto-recorded as expenses.
* Distributor → payments auto-recorded as income.
* Feed → daily feed costs tracked automatically.
* Health → vet/medicine costs added as expenses.

**🧑‍💻 9. Audit Logs**

* Track who created/edited each transaction.
* Store change history for compliance.

**📈 10. Forecasting (Future Upgrade)**

* Predict next month’s income/expenses using historical trends.

7\_8HEALTH SERVICE

**Health Service ❤️** currently does:

* Add health records (diagnosis, medication, vet visit)
* View health history per animal
* Send notifications if a next visit is scheduled

That’s a good start, but in a **real-world dairy farm system**, we can add many more useful features.

**🛠️ Features to Add in Health Service**

**💉 1. Vaccination Management**

* Maintain vaccination schedule per animal.
* Store vaccine type, dose, due date.
* Auto-remind owners/managers when due.

👉 APIs:

* POST /health/{animalId}/vaccination
* GET /health/{animalId}/vaccinations
* GET /health/company/{companyId}/due-vaccinations

**🧾 2. Medical History Reports**

* Summarize animal health history: number of visits, recurring illnesses.
* Filter by disease/medication.

👉 API:

* GET /health/animal/{animalId}/report

**🩺 3. Vet Management**

* Link each health record to a **Vet (user)**.
* Generate reports: "Which vet treated which animals".

👉 API:

* GET /health/vet/{vetId}/animals

**⏰ 4. Scheduled Tasks (Auto Reminders)**

* Use Spring @Scheduled to run daily job → check animals with:
  + Vaccination due
  + Health check due
* Send notifications automatically.

**📊 5. Company Health Dashboard**

* Show % of healthy vs sick animals.
* Common illnesses across the company.
* Avg recovery time.

👉 API:

* GET /health/company/{companyId}/stats

**💰 6. Medical Expense Tracking**

* Record cost of treatment/medicine.
* Integration with **Finance Service** to calculate monthly medical expenses.

👉 API:

* GET /health/company/{companyId}/expenses

**📱 7. Emergency Alerts**

* If critical illness is logged → immediately alert owner via Notification Service.

👉 Example:

* If diagnosis = “Fever > 104°F” → sendNotification("Urgent: High fever in cow #12").

**🧬 8. Integration with Breeding Service**

* If illness affects fertility (e.g., reproductive infection), log note into Breeding Service.

**📂 9. File/Report Upload**

* Allow uploading vet prescriptions or lab reports (PDF, image).

**🌐 10. IoT Health Sensors (Future)**

* Auto-log health parameters (temperature, heart rate) from smart collars/sensors.

8\_FEED SERVICE

**Feed & Nutrition Service 🍀** can:

* Log feeding per animal (feed type, quantity, cost, time)
* Fetch feed history per animal
* Deduct stock from **Inventory Service**
* Send notification if cost is high

That’s a solid base ✅, but we can add many features to make it **resume-worthy and production-grade**.

**🛠️ Features to Add in Feed & Nutrition Service**

**📅 1. Feeding Schedules**

* Define daily feeding plan per animal (e.g., Morning: 5kg Silage, Evening: 2kg Concentrate).
* Auto-check if feeding log matches schedule.  
  👉 API: POST /feed/schedule/{animalId}

**📊 2. Nutrition Analysis**

* Track nutrients (Protein, Fat, Fiber, Energy) per feed type.
* Calculate daily nutrient intake per animal.
* Compare with recommended values → detect deficiencies.  
  👉 API: GET /feed/animal/{id}/nutrition-report

**🐄 3. Group Feeding**

* Instead of logging for each cow, allow **batch feeding** for a group/herd.  
  👉 API: POST /feed/group/{groupId}

**💰 4. Cost Tracking & Finance Integration**

* Send total daily/monthly feeding costs to **Finance Service**.
* Compare feed cost vs milk production → find ROI.  
  👉 API: GET /feed/company/{id}/cost-report

**🔔 5. Missed Feeding Alerts**

* If no feed log for an animal within scheduled time → notify owner.
* “Cow #12 missed evening feeding.”

**📉 6. Feed Waste Tracking**

* Record leftover feed per animal.
* Identify overfeeding or wasted feed.  
  👉 API: POST /feed/{id}/waste

**📦 7. Stock Integration (Inventory Service)**

* Auto-deduct feed stock.
* Alert if stock < threshold.  
  👉 Already partially implemented ✅

**🧑‍⚕️ 8. Vet Recommendations**

* Vet can prescribe special diet (extra minerals, reduced concentrate).
* Link feeding plans with **Health Service**.

**📈 9. Performance Analytics**

* Correlate feed type & milk production.
* Detect which feed combination produces best yield.  
  👉 API: GET /feed/company/{id}/feed-vs-milk

**🌐 10. IoT/Automation (Future)**

* Auto-log feed from **smart feeders**.
* Integrate with sensors to measure real-time consumption.

9\_INVENTORY SERVICE

**Inventory Service 📦** is doing:

* Add/update items (feed, medicine, equipment)
* Deduct stock when used
* Alert if stock < threshold

That’s a strong base ✅, but let’s make it **resume-level** by adding advanced features 🚀

**🛠️ Advanced Features for Inventory Service**

**🧾 1. Stock History & Audit Logs**

* Track every stock change (Added, Consumed, Updated, Deleted).
* Store **who** made the change (owner, manager, vet, worker).  
  👉 API: GET /inventory/{itemId}/history

**📅 2. Expiry Management (for Medicines & Feed)**

* Add expiry date field to items.
* Auto-alert if items are about to expire in X days.  
  👉 Example: "Vaccine Batch #123 expiring in 7 days."

**📊 3. Category-Level Analytics**

* Feed stock usage/month.
* Medicine consumption trends.
* Equipment purchases vs repairs.  
  👉 API: GET /inventory/stats/category

**🛒 4. Purchase Orders (PO) & Restocking**

* Allow creating purchase orders when stock is low.
* Integration with vendors (later).  
  👉 API: POST /inventory/restock

**🔄 5. Auto-Replenishment Rules**

* Define rules → if stock < threshold, auto-create restock request.
* Example: "If Silage < 200Kg → auto restock 1000Kg".

**💰 6. Finance Integration**

* Send stock purchase costs → **Finance Service 💰**.
* Generate expense reports by category.

**🧑‍🤝‍🧑 7. Role-Based Permissions**

* Owner → full access.
* Manager → add/update stock.
* Worker → only view/consume.

**📦 8. Batch Tracking**

* For medicines: batch number, manufacturing date, expiry date.
* Useful for recalls & compliance.

**📉 9. Stock Forecasting**

* Predict stock-out dates based on historical usage.  
  👉 Example: "Silage will run out in 12 days at current usage."

**🌐 10. IoT Integration (Future)**

* Smart silos or weighing machines auto-update stock levels.

10\_MILK SERVICE

**Milk Service** 🥛 can:

* Record daily milk logs per animal
* View logs by animal
* View daily report by company

That’s a good start, but we can make it much richer.

**🛠️ Features to Add in Milk Service**

**📊 1. Reports & Analytics**

* **Daily, Weekly, Monthly reports** per company or per animal.
* Calculate **average milk per cow**.
* Identify **top-performing animals**.
* Detect sudden **drop in milk production** (alert owners).

👉 APIs:

* GET /milk/company/{companyId}/monthly
* GET /milk/animal/{id}/stats

**🥛 2. Milk Quality Tracking**

* Store **fat% and SNF%** (already there).
* Add **CLR (lactometer reading)**.
* Grade milk as **A, B, C** based on fat/SNF ranges.

👉 APIs:

* GET /milk/company/{companyId}/quality-report

**📈 3. Shift & Worker Productivity**

* Track milk logs **per shift (morning/evening)**.
* Track **which worker recorded the entry**.
* Create **worker productivity reports**.

👉 APIs:

* GET /milk/company/{companyId}/shift/{shift}
* GET /milk/worker/{userId}

**💰 4. Revenue Estimation**

* If milk price per liter is set in **Finance Service**,  
  Milk Service can send **total milk quantity per day** to Finance for revenue calculation.

👉 APIs:

* GET /milk/company/{companyId}/revenue?pricePerLitre=40

**🔔 5. Alerts & Notifications**

* Notify owner if:
  + Production drops by >20% from average.
  + Fat% falls below a threshold.
  + An animal misses milk logging for a day.

**📱 6. Integration with Distributors**

* Milk Service → Distributor Service → auto-generate **delivery schedules** based on available milk.
* Example: “Company X has 100 liters available today → distribute to 5 vendors.”

**🧾 7. Export Reports**

* Export reports in **PDF / Excel** for accounting.
* GET /milk/company/{companyId}/report?format=pdf

**🌐 8. IoT Integration (Future)**

* Auto-log milk production from **milking machines**.
* Store real-time data via API.

11\_NOTIFICATION SERVICE

**Notification Service 🔔** a strong part of your project.

**✅ Current Features**

* Accept notification requests (POST /notifications).
* Store them in DB.
* Basic sending (currently logs, later can email/SMS).
* Retrieve history (GET /notifications).

**🚀 Extra Features to Add**

**📩 1. Multiple Channels**

* Email (via SMTP or SendGrid).
* SMS (via Twilio, Nexmo).
* In-app notifications (appear on dashboard in React frontend).
* Push notifications (mobile app future upgrade).

**⏰ 2. Scheduled & Recurring Notifications**

* Daily summary at 7 AM → “3 cows need vaccination today, 2 distributors pending payments”.
* Weekly finance summary → revenue vs expenses.
* Use Spring @Scheduled jobs or Quartz Scheduler.

**🔔 3. Event-Based Alerts**

* **Finance Service** → high expenses, profit drop.
* **Distributor Service** → pending payments > X days.
* **Health Service** → vaccination/medicine reminders.
* **Inventory Service** → stock below threshold.
* **Feed Service** → feed schedule missed.

**📊 4. Notification Dashboard**

* Show categorized notifications: Finance, Health, Distributor, Feed, etc.
* Filters: unread/read, by date, by type.
* Mark as read/unread.

**👥 5. Role-Based Notification Delivery**

* Owner → sees all notifications.
* Manager → finance + workers.
* Vet → only health-related alerts.
* Worker → only assigned tasks.

**🔐 6. Security & Personalization**

* Each notification stored with userId → only that user sees it.
* Store priority (HIGH, MEDIUM, LOW).

**📂 7. Integration with Other Services**

* Expose a **Feign Client** so services like Finance/Distributor can directly call sendNotification().
* Example:
  + Distributor Service calls → "Payment pending for Distributor A".
  + Finance Service calls → "Expenses exceeded 50% of income this month".

**📱 8. Future Upgrades**

* WebSocket support → instant real-time notifications in frontend React app.
* Mobile app integration → push notifications.

12\_EUREKA SERVICE

**🧩 Features We Can Add in Eureka Service**

Right now, Eureka is just running as a **basic service registry**.  
We can make it more powerful by adding features:

**1. High Availability Setup**

* Run **multiple Eureka servers** in a cluster so if one goes down, services still register.

**2. Self-Healing**

* Enable **self-preservation mode** (or keep disabled for dev).
* Protects registry from network glitches by not removing instances too quickly.

**3. UI Customization**

* Customize **Eureka Dashboard** (logo, themes) to make your project stand out.

**4. Service Health Monitoring**

* Integrate with **Spring Boot Actuator** to show each service’s /health status directly in Eureka.

**5. Metrics Export**

* Add Micrometer + Prometheus → see service discovery metrics in Grafana.

**6. Secure Eureka**

* Protect the Eureka dashboard with **Spring Security** (only admins can access http://localhost:8761).

**7. Cross-Region Discovery**

* Configure Eureka to work across multiple zones/regions (for cloud deployment).