﻿

auth:

* đăng kí, đăng nhập, logout
* khi đăng kí sẽ yêu cầu
  + họ, tên -> tên hiển thị
  + email + password-> để login
  + email -> gửi email trong tương lai
  + sđt
* đăng nhập bằng user name + password
  + mã hoá password
  + jwt
  + access + refresh token -> giữ trạng thái đăng nhập trong một khoảng thời gian nhất định

actor/users:

* restaurant
  + profile: tên quán, mô tả, ngày giờ mở cửa,
  + status: đóng mở cửa (socket io)
  + ví
    - biến động số dư
    - số dư
  + món ăn: CRUD, thống kê, bật tắt món
  + voucher: CRUD,
  + order: chấp nhận, từ chối, xử lý,đơn đã sẵn sàng (socket io)
    - chấp nhận đơn hàng
    - từ chối khi hết món hoặc lý do khác - > trừ elo
    - xác nhận đã chuẩn bị xong -> bắn socket cho tài xế thông báo đơn hàng đã chuẩn bị xong
  + thống kê, doanh thu, top bán chạy, category bán chạy…
* Custommer
  + Profile: tên hiển thị, avatar, bio, rank,
  + Thêm địa chỉ, (có thể nhiều địa chỉ tương ứng nhiều label)
    - Số nhà, tên đường
    - Phường
    - Thành phố
    - Ghi chú( đặc điểm nhận dạng, chỉ đường nếu khó tìm)
    - Cho chọn vị trí chính xác trên bản đồ
    - Đặt làm mặc định hay không
  + Thống kê tổng chi
    - Tổng tiền đã chi cho các đơn hàng ( set cho đến mức nào đó sẽ tăng bậc rank)
  + tìm kiếm món ăn gần đây
  + folow nhà hàng
  + đề xuất món ăn
    - từng mua trước đây
    - từng tìm kiếm
    - gần đây
    - bán chạy
  + thêm món vào giỏ hàng
    - cập nhật notification
  + tick những món cần mua
    - mặc định chọn hết
    - chọn theo nhà hàng
    - chọn riêng lẻ từng món
    - hệ thống tự tính tổng tiền
  + thiết lập thông tin giao hàng
    - chọn địa chỉ từ danh sách địa chỉ, có lựa chọn khác link tới trang thêm địa chỉ
    - khoảng cách đơn hàng được tính ở đây luôn ( hệ tống tính phí ship dựa trên khoảng cách)
    - tên , sdt người nhận
      * mặc định lấy từ user đó
      * nêu là đặt giùm thì người dùng có thể chỉnh sửa 2 trường này
      * ghi chú đơn hàng
  + chọn phương thức thanh toán, cod hay chuyển khoản
  + nhấn đặt hàng
    - tạo đơn hàng
    - nếu đơn hàng có nhiều quán thì tách ra nhiều đơn hàng tương ứng với số quán
    - phí ship đơn tương ứng với khoảng cách người dùng đó tới quán đó
    - tổng tiền bằng phí ship cọng tổng tiền các món hàng và phụ phí nếu có
  + Trạng thái đơn hàng
    - Được cập nhật realtime (socket io)
    - Vị trí tài xế nhận đơn đó (socket io)
    - Được huỷ đơn khi nhà hàng chưa xác nhận đơn (socket io)
    - Chat với tài xế (socket io)
    - Đã nhận đơn
    - Đánh giá món ăn
    - Đánh giá quán
    - Đánh giá tài xế
* Driver
  + Đăng nhập, đăng xuất, đăng kí
  + Profile
    - Họ tên
    - Avatar
    - Mã tài xế
    - Thâm niên
    - Dịch vụ, hợp đồng
    - Thông tin liên lạc sđt, email
    - Thuế
  + Hiệu suất hoạt động
    - Hiệu suất theo ngày
    - Thời gian khi check in đến khi check out
    - Thời gian hoạt động giờ cao điểm
    - Số đơn hoàn thành
    - Số đơn từ chối
    - Số đơn bỏ qua
    - Đạt hay không đạt
  + Ví
    - Số dư
    - Các biến động số dư
  + Lịch sử đơn hàng
  + Thu nhập
    - Thống kê thu nhập theo ngày, tháng, năm
  + Trung tâm trợ giúp
    - Các vấn đề tài xế hay mắc phải và đề xuất hướng giải quyết
  + Sẵn sàng hoạt động
    - Đơn hàng sẽ được phát cho tài xế, hiển thị số km, điểm lấy, điểm giao, số tiền nhận được,
    - Tài xế có quyền bỏ qua -> hệ thống sẽ âm thầm trừ điểm đánh giá ẩn của tài xế
    - Khi đã nhận đơn, có thể huỷ khi chờ lâu trên 20p để đơn hàng tiếp tục gán cho tài xế khác
    - Huỷ đơn khi gặp trường hợp bất khả kháng, không thể tiếp tục thực hiện đơn -> bị band 40p 😊))
    - Lấy đơn nhưng không giao cho khách -> trừ tiền giá trị đơn hàng đó trong ví đền cho khách
  + Các trạng thái khi nhận hàng
    - Đã đến nhà hàng: chỉ chuyển trạng thái khi mà khoảng cách nhà hàng với tài xế dưới 200m, nếu trên 200m hiển thị cảnh báo và yêu cầu chụp hình bằng chứng và ghi vào log
    - Đã lấy đơn: lấy đơn hàng
    - Đã đến điểm giao: xác thực giống bước 1 nếu vị trí không hợp lệ
    - Hoàn thành: có xác nhận thông báo hoang thành đơn hàng ( tài xế confirm tránh bấm nhầm) bước này mất 2 thao tác
    - Sau khi hoàn thành đơn hàng thì ẩn sđt, địa chỉ và đóng chat với khách
    - Các trạng thái phải theo trình tự, không nhảy vọt
* Admin
  + MAP
    - Hiển thị map realtime xem các quán đang hoạt động
    - Vị trí các tài xế
    - Vị trí khách đang đặt đơn
  + Thống kê
    - Số lượng tài xế đang hoạt động
    - Số lượng nhà hàng đang hoạt động
    - Số lượng món ăn đang hoạt động và không hoạt động (quán mở mới hiển thị món)
    - Các đơn hàng đang thực hiện
    - Các đơn hàng đã hoàn thành
    - Doanh thu từ nhà hàng
    - Doanh thu từ tài xế
  + Quản lý nhà hàng
    - Xem được chi tiết nhà hàng, tất cả mọi thứ của nhà hàng đó
    - Xử lý qua các đơn hàng đánh giá 1 sao
    - Tắt món đó nếu quá nhiều đánh giá 1 sao
    - Cảnh báo nếu có bất thường
    - Tạm dừng hoạt động nếu phát hiện vi phạm
  + Quản lý customer
    - Thống kê user top chi tiêu, top mua nhiều
    - Band nếu phát hiện vi phạm (spam , bùng đơn, lạm dụng voucher…)
    - Xem được tất cả thông tin chi tiết về khách hàng cụ thể
  + Quản lý tài xế
    - Thống kê (EDA)
    - Ban nếu phát hiện vi phạm( trục lợi bất chính, thô lỗ khách hàng, thô lỗ nhà hàng, tiết lộ thông tin khách hàng, thô lỗ trung tâm hỗ trợ….)
* Server
  + DB tối ưu, phân tán
    - Các bảng thiết kế không để tồn tại mối quan hệ nhiều nhiều, chuyển hết về 1-1 hoặc 1- nhiều
  + Đủ mạnh đáp ứng được số lượng user lớn
  + Hệ thống đề suất món ăn cá nhân hoá từng customer
  + Hệ thống gán đơn tối ưu, công bằng cho từng tài xế:
    - Khi phát sinh đơn
    - Hệ thống lấy ra các tài xế gần điểm lấy
    - Chọn ra tài xế có tỉ lệ đơn hoàn thành trong ca làm việc thấp so với trung bình các tài xế còn lại, sau đó đến tiêu chí điểm elo ẩn cao nhất nếu bằng nhau thì lấy tài xế gần hơn…
    - Bắn socket io cho tài xế, nếu tắt màn hình thì dùng fcm gửi thông báo
  + Lưu trữ log phòng khi hệ thống bị sập

**TỔNG QUAN HỆ THỐNG GIAO ĐỒ ĂN – KIẾN TRÚC & CHUẨN HÓA**

**Tổng thể**

* Hiện trạng: Phạm vi đầy đủ, chia vai rõ ràng, luồng đơn hàng hợp lý, có realtime và thống kê.
* Mục tiêu hoàn thiện: Chuẩn hóa về bảo mật, tính sẵn sàng và ranh giới giữa các service.

**1. Auth & User Service**

**Bắt buộc**

* Xác thực hai lớp (2FA): OTP qua SMS/email, mã khôi phục, nhớ thiết bị tin cậy.
* Khôi phục mật khẩu: token dùng một lần có thời hạn (TTL), giới hạn tần suất yêu cầu.
* Rate limit và lockout: kiểm soát theo IP và tài khoản, bật CAPTCHA khi vượt ngưỡng.
* RBAC chi tiết: middleware kiểm tra scope/permission cho từng endpoint, cần tài liệu hóa ma trận quyền.
* Chuẩn hóa định danh: thống nhất đăng nhập bằng email hoặc số điện thoại, đảm bảo tính duy nhất và hợp lệ.

**Nên có**

* Phát hiện đăng nhập bất thường: gửi email cảnh báo khi có thiết bị mới.
* Token lifecycle chuẩn: refresh rotation và revoke list để thu hồi token cũ.

**2. Restaurant Service**

**Bắt buộc**

* Review và phản hồi: có cơ chế chống spam, giới hạn tần suất, áp dụng trọng số thời gian để tránh thao túng xếp hạng.
* Activity log: ghi lại mọi thay đổi menu, giá, trạng thái và cho phép xuất file.
* Wallet ledger: lưu mã giao dịch, loại giao dịch, đơn tham chiếu (#), đối soát định kỳ.
* Giới hạn ELO: có ngưỡng tối thiểu, giảm dần theo thời gian, ngăn lạm dụng.
* Thống kê linh hoạt: cho phép lọc theo thời gian, danh mục, kênh thanh toán, cohort khách hàng.

**Vận hành**

* Hiển thị SLA cảnh báo khi nhà hàng thường xuyên từ chối đơn.

**3. Customer Service**

**Voucher**

* Quy định rõ: áp dụng một hay nhiều voucher trên mỗi đơn; thứ tự áp dụng; điều kiện sử dụng (giá trị tối thiểu, danh mục, nhà hàng).
* Quản lý hạn sử dụng, số lần dùng, và quy tắc chống cộng dồn (stacking matrix).

**Thanh toán**

* Phân tách rõ COD và Online:
  + Trạng thái thanh toán độc lập với trạng thái đơn hàng.
  + Webhook/callback an toàn, có idempotency key và HMAC.
* Chính sách hoàn tiền (refund) và xử lý tranh chấp (chargeback), hỗ trợ hoàn một phần.

**Hệ thống Rank**

* Định nghĩa rõ ngưỡng điểm, quyền lợi (miễn phí ship, ưu tiên CSKH), và ảnh hưởng đến đề xuất món.

**Follow / Feed**

* Người dùng có thể theo dõi nhà hàng để nhận thông báo món mới hoặc khuyến mãi.
* Cài đặt tùy chọn thông báo (notification preference).

**Chat**

* Cho phép trò chuyện giữa khách - tài xế và khách - nhà hàng.
* Lưu transcript với chính sách lưu trữ có thời hạn.

**Gợi ý món (Recommendation)**

* Tách thành module riêng, sử dụng thuật toán CF hoặc Content-based.
* Hỗ trợ thử nghiệm A/B và bật/tắt theo feature flag.

**4. Driver Service**

* Check-in/out: ghi nhận ca làm và thời gian hoạt động.
* Đánh giá khách: có cơ chế chống boom hàng, tự động xử lý khi điểm đánh giá thấp liên tục.
* GPS giả lập: có flag riêng cho môi trường dev/test.
* Quy trình bồi thường: cần xác nhận bởi admin thứ hai (nguyên tắc 4-eyes).
* Trung tâm hỗ trợ/đào tạo: có bài kiểm tra hoặc checklist onboarding.

**5. Admin Service**

* Dashboard SRE: theo dõi tỷ lệ lỗi, cảnh báo socket, queue depth, độ trễ hệ thống.
* Phân quyền nội bộ: chia rõ Superadmin, Admin, Mod và phạm vi chức năng.
* Audit log: lưu lịch sử hành động không thể chỉnh sửa (immutable store).
* Báo cáo định kỳ: xuất file CSV/PDF, hỗ trợ drilldown chi tiết.
* Manual override: cho phép giao đơn thủ công, “pin” tài xế cụ thể.

**6. System Design**

**Kiến trúc**

* Giai đoạn đầu: xây dựng dạng Monolith module hóa (NestJS).
* Giai đoạn mở rộng: tách thành microservice theo domain (Auth, Order, Payment, Notification, Recommendation).
* Sử dụng hàng đợi (Kafka hoặc RabbitMQ) cho các tác vụ bất đồng bộ như gán đơn, gửi thông báo, cập nhật chỉ số.

**Hạ tầng**

* Redis: lưu session, cache danh mục/nhà hàng phổ biến, rate limit.
* CDN: lưu trữ ảnh món và avatar, hỗ trợ resize động.
* Bảo mật: bật CORS nghiêm ngặt, dùng Helmet, Joi/Zod để validate input, rotation secret định kỳ, mTLS giữa các service.
* Backup/DR: sao lưu DB hàng ngày và kiểm thử restore định kỳ.
* Monitoring: Prometheus/Grafana cho metric, ELK hoặc OpenSearch cho log, Jaeger/OpenTelemetry cho tracing.
* Idempotency: áp dụng cho các API thanh toán, tạo đơn, webhook.
* Multi-tenancy/Region: tách dữ liệu theo vùng khi mở rộng quy mô.

**7. Data Model & Consistency**

**Đơn hàng**

* Tách rõ paymentStatus và orderStatus.
* Có timeline/tracking history thống nhất.
* Lưu thông tin người nhận (recipientName, phone) và người đặt (purchaserPhone).

**Voucher & Wallet**

* Ledger bất biến (append-only).
* Có job đối soát định kỳ để đảm bảo nhất quán.

**Review**

* Chống review spam bằng rate limit và cooldown.
* Áp dụng trọng số theo lịch sử mua hàng.

**8. Quality & Compliance**

* Kiểm thử đầu-cuối (E2E) và chaos testing: mô phỏng mất kết nối, kiểm tra chiến lược retry và backoff.
* Đặt mục tiêu hiệu năng (performance budget) cho các API quan trọng ở mức P95/P99.
* Đảm bảo quyền riêng tư và tuân thủ: ẩn/mask dữ liệu nhạy cảm (số điện thoại, địa chỉ) cho các vai trò không liên quan.