LCE 2.0 – Technical Specification (Backend + New Users Backend)

Version: Draft 1

0. Purpose & Scope
1. Product Overview
2. Website & User Flows
3. Scheduling & Service Model
4. Data Model (MySQL)
5. Business Logic
6. Users Backend (Admin / CSR Portal)
8. Migration & Backwards Compatibility
9. Non-Functional Requirements
10. Implementation Checklist (High-Level)

0. Purpose & Scope

This document defines the **functional** and **technical** specification for Laundry Care Express 2.0, including:

- New **consumer offerings**:
  - **Pay per Order (PPO by the pound)**
  - **Subscribe & Save (Subscription by the bag)**
- **Customer website flows** (public site + logged-in experience)
- **Scheduling model** (One Time Service / Weekly (Bi-Weekly))
- **Subscription logic** (bag banking, overages, annual discount, etc.)
- **Pricing model** and integration with existing price lists
- **Order, invoice, transaction, and promo** handling
- A new **Users Backend (Admin/CSR dashboard)** for internal staff
- Required changes and additions to the existing **MySQL schema** and **backend APIs**

The goal: a single source of truth that backend, frontend, and mobile engineers can implement from.

---

1. Product Overview

1.1 Offer Types

1. Pay per Order ("PPO")

- **Audience**: Residential and commercial "pay-as-you-go" customers.
- **Pricing:**
  - **Base W&F Rate**: `$2.99 / lb` (configurable).
  - **Minimum**: `$30` of laundry (**~10 lbs**), plus:
  - **P&D fee**: `$9.99`
  - **Service fee**: `$5.00`
  - So the minimum order total is `$*` for next-day turnaround.
- **Fees:**
  - **P&D (Pickup & Delivery): $9.99 per order.**

    - **Service fee: $5.00 per order.**
    - Both should be configurable via database.

- **Turnaround:**
    - **Standard next-business-day** for W&F.

- **Individually priced items:**
    - **Wash and Fold - Individual Items (WF)**
    - **Dry cleaning (DC)**
    - **Hang dry (HD)**

- **No-laundry fee:**
    - Existing "no laundry fee" is effectively replaced with P&D fee; logic must support charging P&D if driver shows up and no clothes.

2. Subscribe & Save (Subscription by the Bag)

- **Subscription plans** (per month, 1 bag = ~20–21 lbs):
    - 1 bag / month: **$70**
    - 2 bags / month: **$67 per bag**
    - 4 bags / month: **$65 per bag**
    - 8 bags / month: **$64 per bag**
    - Prices configurable; those values are defaults.

- **Bag details**:
    - **~2948 cubic inches** (roughly **20 gallon** bag), about **21 lbs** capacity baseline used for modeling.

- **Overages**:
    - If **bag weight > capacity threshold** (e.g. 20–21 lbs), the "overage" lbs are charged at **PPO per-lb rate**.
    - If **number of bags per month > plan bag count**, extra bags are charged at **per-bag subscription rate** (plan-level rate).

- **Banking**:
    - **Unused bags can be banked** and carried over as long as subscription remains active.

- **Annual subscription**:

        - **Annual prepay discount: 15%** off monthly pricing.
        - Early cancellation: pro-rated refund with **15% fee**, minimum **$100**.
- **Fees**:
    - **No P&D or service fees** for subscription orders. (“Subscriptions waive all P&D fees, Service Fees.”)
- **Included items**:
    - **W&F in subscription bags** is included.
    - **Individual DC/HD/LP items**:
        - Included if they **fit in the bag** (exact rules may be clarified at implementation).
        - If they don’t fit, they are charged per price list (PPO logic).
- **Scheduling**:
    - Subscription customers can schedule:
        - **One Time Service -** next available pickup or any day in the future
        - **Weekly (Bi-Weekly) -** recurring pickups

1.3 Up-Front Promotions & Credits

- **New user pop-up** on home page:
    - Offers **$20 credit**, added to user account after signup.
- **Promo codes**:
    - Existing promo code structure is reused, plus new types for subscription offers where needed.

1.4 Turnaround & Operations

- **Operating days**: Mon–Fri initially (configurable), with potential for Sat later.
- **Turnaround**:
    - Subscription: same as PPO for W&F; DC/HD may have longer turnaround (configurable).
- **Laundry handling**:

    - Cold water only, medium heat dryers by default.
    - DC/LP items handled according to price list instructions.
- **Labeling**:
    - **Wash and Fold (WF)**: flagged and returned in WF bag.
    - **Hang Dry (HD)**: flagged and returned in HD bag.
    - **Dry Cleaning (DC)**: flagged and returned in DC bag.

---

2. Website & User Flows

2.1 Navigation (Public Website)

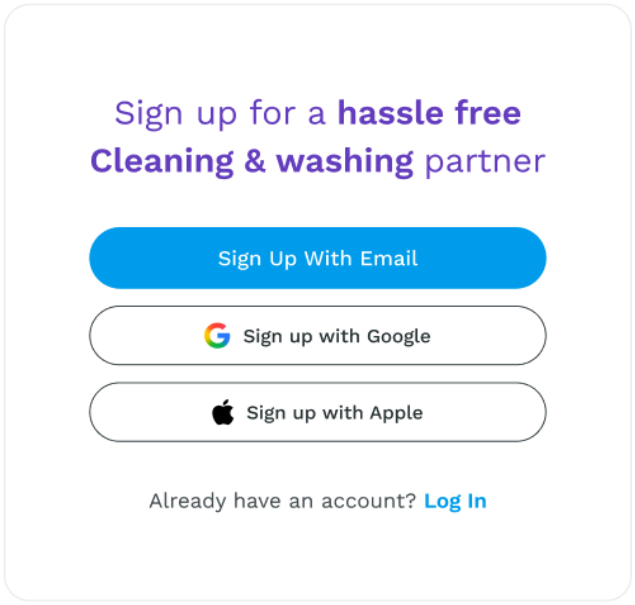
From **“LCE 2.0 Public Website navigation”**.

https://docs.google.com/drawings/d/1ub4Xxd2e7o8B_4j2XHcbaULDa4tKHVJZvGKQeNRf1GU/edit

2.2 Core Signup & Onboarding Flow

Goal: new user goes from website to scheduled pickup in 3 steps.

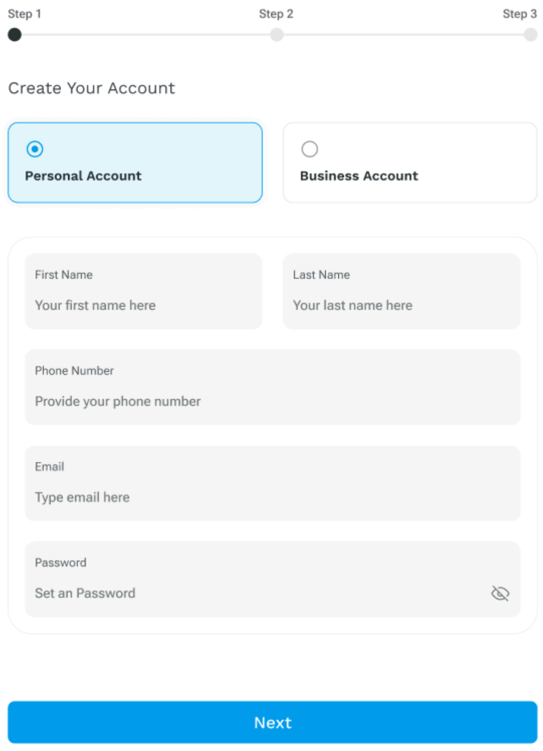
Step 1 – Account Creation (“Step 1 of 3”)


Sign up for a hassle free Cleaning & washing partner
Sign Up With Email
Sign up with Google
Sign up with Apple
Already have an account? Log In

- User sees signup with:
  - **Sign up with Email**
  - **Sign up with Google**
  - **Sign up with Apple**
  - **“Already a customer? Login”** link.
- If email:
  - Form fields:
  - First name
  - Last name
  - Mobile phone
  - Email
  - Password
  - Accept terms & privacy policy
    - By selecting continue, you agree to receive service and marketing auto-sent texts from Rinse. Opt-out anytime on your “My Account” page or text “STOP”. Message frequency varies. Message & data rates may apply. By continuing, you also agree to our Terms and Privacy

Policy.

- Create **User (auth)** + **Customer profile** records.
- Apply **$20 new-user credit** (if eligible).
- Create session and move to Step 2.


Step 1 Step 2 Step 3

Create Your Account

[✓ Personal Account] Business Account

First Name: Your first name here

Last Name: Your last name here

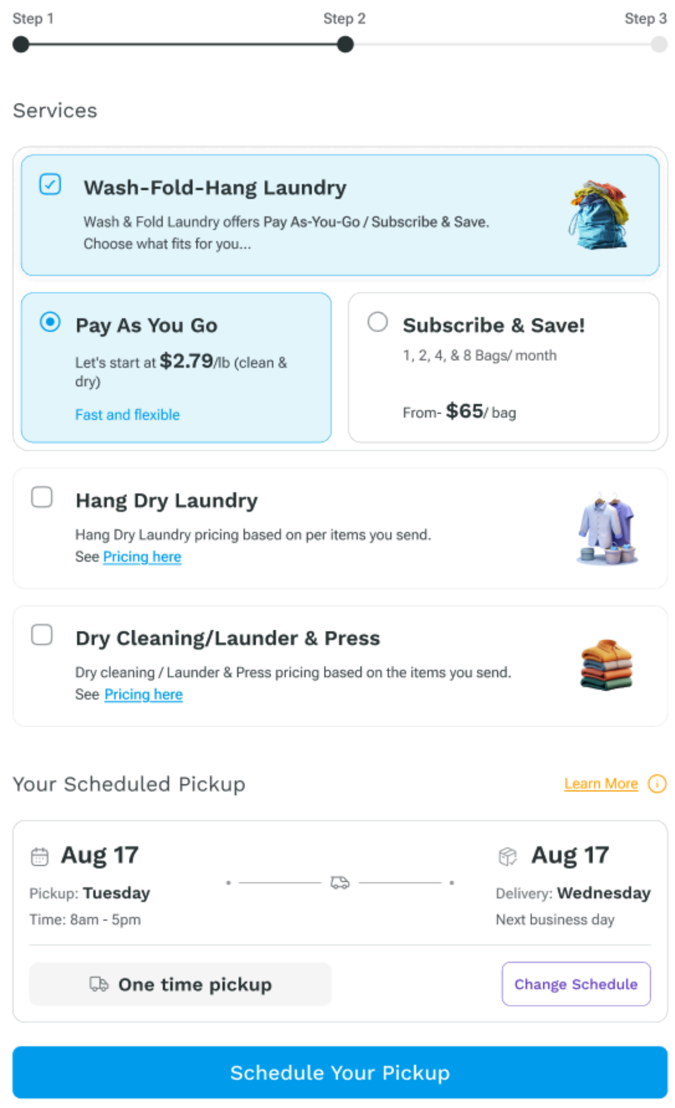
Phone Number: Provide your phone number

Email: Type email here

Password: Set an Password

Next

Step 2 – Address & Serviceability (“Step 2 of 3”)



Step 1 Step 2 Step 3

Services

[✓] **Wash-Fold-Hang Laundry**
Wash & Fold Laundry offers Pay As-You-Go / Subscribe & Save. Choose what fits for you...

(●) **Pay As You Go**
Let's start at **$2.79**/lb (clean & dry)
Fast and flexible

( ) **Subscribe & Save!**
1, 2, 4, & 8 Bags/ month
From- **$65**/ bag

[ ] **Hang Dry Laundry**
Hang Dry Laundry pricing based on per items you send.
See Pricing here

[ ] **Dry Cleaning/Launder & Press**
Dry cleaning / Launder & Press pricing based on the items you send.
See Pricing here

Your Scheduled Pickup — Learn More

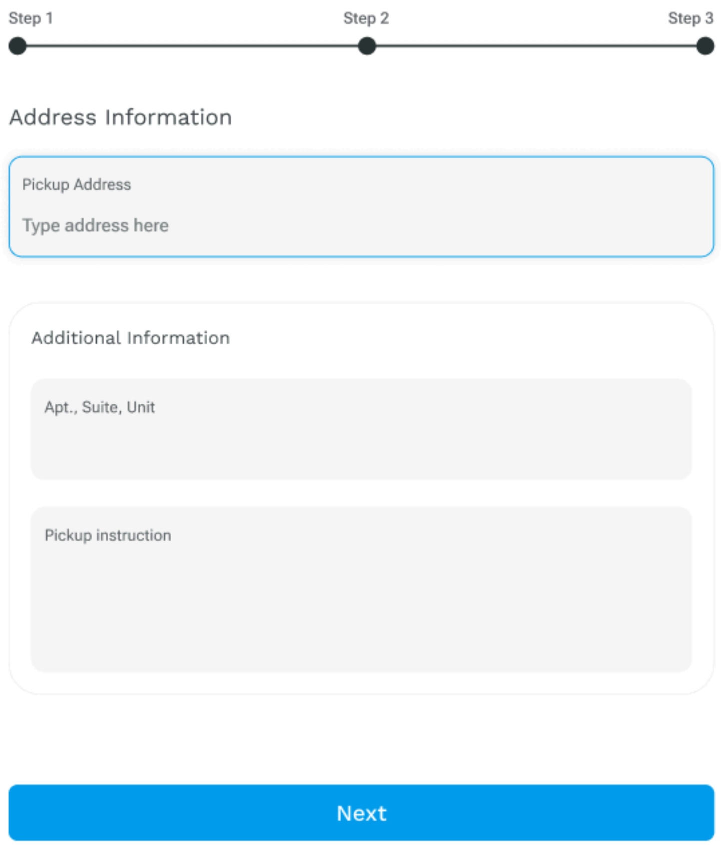
Aug 17
Pickup: **Tuesday**
Time: 8am - 5pm

Aug 17
Delivery: **Wednesday**
Next business day

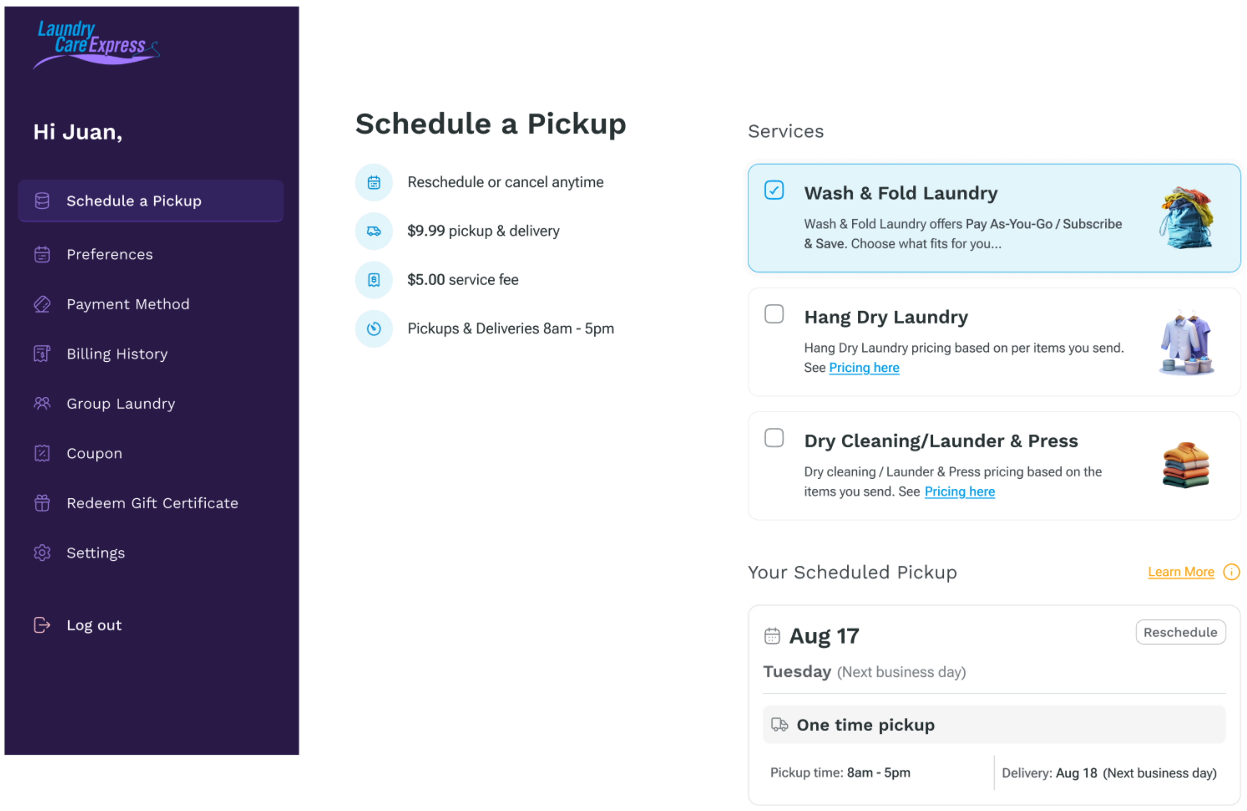
One time pickup — Change Schedule

Schedule Your Pickup

Step 3 – Scheduling ("Step 3 of 3")



2.3 Returning User Flows (Web/App)



Schedule a Pickup

- Reschedule or cancel anytime
- $9.99 pickup & delivery
- $5.00 service fee
- Pickups & Deliveries 8am - 5pm

Services

[✓] **Wash & Fold Laundry**
Wash & Fold Laundry offers Pay As-You-Go / Subscribe & Save. Choose what fits for you...

[ ] **Hang Dry Laundry**
Hang Dry Laundry pricing based on per items you send. See Pricing here

[ ] **Dry Cleaning/Launder & Press**
Dry cleaning / Launder & Press pricing based on the items you send. See Pricing here

Your Scheduled Pickup — Learn More

Aug 17 — Reschedule
Tuesday (Next business day)

One time pickup

Pickup time: 8am - 5pm | Delivery: Aug 18 (Next business day)

---

3. Scheduling & Service Model

3.1 Scheduling Matrix Summary

From **Scheduling Matrix** sheet:

- **LCE 1.0**:
    - PPO by pound (existing).
    - SCHEDULING:
        - One-time (ASAP/future).
        - Weekly (Bi-weekly) for all customers.
- **LCE 2.0**:
    - For Residential & Commercial:

    - **Payment types**:
        - PPO (By the Pound) with P&D + service fee.
        - Subscription (By the Bag, no fees).
    - **SCHEDULING**:
        - For **new customers**:
            - One-time (ASAP/future).
            - (Bi-)Weekly recurring.
    - **Note 1 (Overages)**:
        - Overages charged **by the pound at PPO rate**.

3.2 Scheduling Rules

1. **ASAP**:
    - Choose earliest available date given:
        - Zone service days (Mon–Fri).
        - Non-working days / holidays (see DB).
        - Daily cutoff time (configurable).
2. **Future date**:
    - Customer can choose any available date on scheduling calendar.
    - Calendar must exclude:
        - Non-service days for user’s zone.
        - Non-working days/holidays.
3. **Weekly (Bi-weekly) recurring**:
    - User chooses:
        - Weekly or Bi-Weekly
        - Day of week (within zone service days).

    - System creates **recurring schedule** record for user.
    - Upcoming pickups generated automatically as events.

---

4. Data Model (MySQL)

The current production schema is in `lce_site` database. Key existing tables (abbreviated):

- `f29om_users` – **Legacy** Auth users (email, password, block, etc.)
- `lce_user_info` – Customer details, contact info, addresses, payment, preferences.
  New Fields: Auth users (email, password, block, etc.)
- `lce_pickup_zones` – Zones by ZIP/city/state + days of operation + geometry.
- `lce_pickup_nonworking_days`, `lce_holidays_logs` – Non-working days and any holiday email logs.
- `lce_prices`, `lce_prices_lists` – Price lists and items.
- `lce_processing_sites` – Processing facilities and and price list assignments.
- `lce_user_pickup` – Core pickup/order record (per pickup).
- `lce_user_invoice`, `lce_user_invoice_line` – Invoice header and lines.
- `lce_user_transactions`, `lce_payment` – Transaction and payment logs.
- `lce_promo_codes`, `lce_user_promocode`, `lce_user_promocodes` – Promo definitions and usage.
- `lce_user_rs` – Recurring schedule.
- `lce_user_cs`, `lce_user_cs_log` – Customer service tickets and logs.
- `lce_user_group_*` tables – Group admin, members, logs, and history.
- `lce_communication_settings` – Email/SMS communication preferences.
- `lce_users_vacation_logs` – Vacation holds.
- `lce_waiting_list`, `lce_tmp_*` tables – Waiting list and temp data.

4.1 High-Level Entity Model

New and existing entities:

- **UserAuth**
  - Backed by `lce_user_info`.
  - Fields: id, name, username (email), password hash, block, registerDate, lastvisitDate, etc.
- **CustomerProfile**
  - Backed by `lce_user_info`.
  - Contact info, primary address, billing address, payment profiles, preferences, price list, etc.
- **PickupZone**
  - Backed by `lce_pickup_zones`.
- **Order / Pickup**
  - Backed by `lce_user_pickup`.
- **Invoice** and **Invoice Line**
  - `lce_user_invoice`, `lce_user_invoice_line`.
- **Transaction**
  - `lce_user_transactions`, `lce_payment`.
- **PriceList** / **PriceItem**
  - `lce_prices_lists`, `lce_prices`.
- **Promo & Credit**
  - `lce_promo_codes`, `lce_user_promocode`.
- **SubscriptionPlan** (NEW)
- **UserSubscription** (NEW)
- **SubscriptionBagUsage** (NEW)

- **UserCredits** (NEW or extension of transactions)
- **CommunicationSettings**
  - `lce_communication_settings`.
- **RecurringSchedule**
  - `lce_user_rs`.
- **VacationHold**
  - `lce_users_vacation_logs`.
- **GroupAccount**
  - `lce_user_group_admin`, `lce_user_group_members`, `lce_user_group_members_history`.

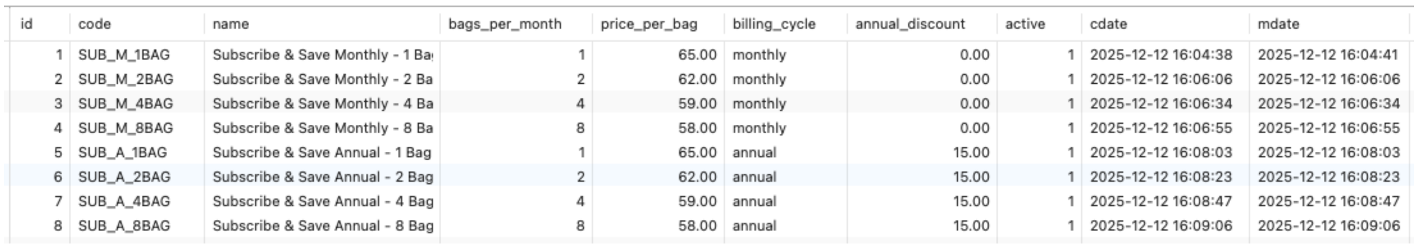
4.2 Required New Tables

Below are proposed new tables (names and columns are suggestions; adjust as needed).

4.2.1 `lce_subscription_plans`

Stores configuration for subscription plans (Subscribe & Save).

```
CREATE TABLE lce_subscription_plans (
  id                  INT AUTO_INCREMENT PRIMARY KEY,
  code                VARCHAR(32) NOT NULL,  -- e.g. 'SUB_M_1BAG', 'SUB_A_2BAG'
  name                VARCHAR(64) NOT NULL,  -- e.g. 'Subscribe & Save – 1 Bag'
  bags_per_month      INT NOT NULL,          -- 1, 2, 4, 8
  price_per_bag       DECIMAL(10,2) NOT NULL,
  billing_cycle       ENUM('monthly','annual') NOT NULL DEFAULT 'monthly',
  annual_discount     DECIMAL(5,2) DEFAULT 15.00, -- percent
  active              TINYINT(1) NOT NULL DEFAULT 1,
  cdate               DATETIME,
  mdate               DATETIME
);
```



id	code	name	bags_per_month	price_per_bag	billing_cycle	annual_discount	active	cdate	mdate
1	SUB_M_1BAG	Subscribe & Save Monthly - 1 Ba	1	65.00	monthly	0.00	1	2025-12-12 16:04:38	2025-12-12 16:04:41
2	SUB_M_2BAG	Subscribe & Save Monthly - 2 Ba	2	62.00	monthly	0.00	1	2025-12-12 16:06:06	2025-12-12 16:06:06
3	SUB_M_4BAG	Subscribe & Save Monthly - 4 Ba	4	59.00	monthly	0.00	1	2025-12-12 16:06:34	2025-12-12 16:06:34
4	SUB_M_8BAG	Subscribe & Save Monthly - 8 Ba	8	58.00	monthly	0.00	1	2025-12-12 16:06:55	2025-12-12 16:06:55
5	SUB_A_1BAG	Subscribe & Save Annual - 1 Bag	1	65.00	annual	15.00	1	2025-12-12 16:08:03	2025-12-12 16:08:03
6	SUB_A_2BAG	Subscribe & Save Annual - 2 Bag	2	62.00	annual	15.00	1	2025-12-12 16:08:23	2025-12-12 16:08:23
7	SUB_A_4BAG	Subscribe & Save Annual - 4 Bag	4	59.00	annual	15.00	1	2025-12-12 16:08:47	2025-12-12 16:08:47
8	SUB_A_8BAG	Subscribe & Save Annual - 8 Bag	8	58.00	annual	15.00	1	2025-12-12 16:09:06	2025-12-12 16:09:06

4.2.2 `lce_user_subscriptions`

Per-customer subscription.

```
CREATE TABLE lce_user_subscriptions (
  id                    INT AUTO_INCREMENT PRIMARY KEY,
  user_id               INT NOT NULL,
  plan_id               INT NOT NULL,  -- FK to lce_subscription_plans
  status                ENUM('pending','active','paused','cancelled', ‘upgraded’) NOT NULL
                         DEFAULT 'active',
  billing_cycle         ENUM('monthly','annual') NOT NULL,
  start_date            DATE NOT NULL,
  end_date              DATE NOT NULL,
  next_renewal_date     DATE NOT NULL,
  bags_plan_period      INT NOT NULL,  -- per month, from plan
  bags_plan_total       INT NOT NULL,  -- total, 1,12,24,...
  bags_plan_balance     INT NOT NULL DEFAULT 0, -- total minus used bags
  bags_plan_used        INT NOT NULL DEFAULT 0,  -- used since beginning
  bags_available        INT NOT NULL DEFAULT 1,  -- available now
  created_via           ENUM('web','intra','other') DEFAULT 'web',
  payment_last          DECIMAL(10,2) DEFAULT 0.00,
  payment_discount      DECIMAL(10,2) DEFAULT 0.00,
  payment_balance       DECIMAL(10,2) DEFAULT 0.00,  -- payment_last minus used
  notes                 TEXT,
  cdate                 DATETIME,
  mdate                 DATETIME,
  INDEX idx_user_id (user_id),
  INDEX idx_status (status)
);
```

4.2.3 `lce_subscription_bag_usage`

Tracks bag usage per subscription per period.

```
CREATE TABLE lce_subscription_bag_usage (
  id                     INT AUTO_INCREMENT PRIMARY KEY,
  user_subscription_id INT NOT NULL,
  invoice_id             INT NULL,
  pickup_id              INT NULL,
  bags_used              INT NOT NULL DEFAULT 1,  -- used for this pickup
  cdate                  DATETIME,
  mdate                  DATETIME,
  INDEX idx_user_sub_period (user_subscription_id, period_start, period_end)
);
```

4.2.4 `lce_user_credits`

Handles credits (like the $20 new-user credit) and other account credits.

```
CREATE TABLE lce_user_credits (
  id              INT AUTO_INCREMENT PRIMARY KEY,
  user_id         INT NOT NULL,
  type            ENUM('welcome','promo','manual','refund') NOT NULL,
  description     VARCHAR(255) NOT NULL,
  amount          DECIMAL(10,2) NOT NULL,
  balance       DECIMAL(10,2) NOT NULL,
  expires_at    DATETIME NULL,
  used          TINYINT(1) NOT NULL DEFAULT 0,
  cdate         DATETIME,
  mdate         DATETIME,
  INDEX idx_user (user_id)
);
```

The billing logic will subtract from `lce_user_credits.balance` before charging payment method.

4.3 Changes / Extensions to Existing Tables

4.3.1 `lce_user_pickup` (Orders)

Add fields for subscription vs PPO and meta:

- `order_type` ENUM('PPO','subscription','business') – differentiate service model.
- `subscription_bags` INT – number of bags in this pickup for subscription.
- `subscription_period_start` / `subscription_period_end` DATE – for linking usage.
- `subscription_overweight_lbs` DECIMAL(10,2).
- `subscription_overweight_charge` DECIMAL(10,2).
- `subscription_bag_capacity_lbs` DECIMAL(10,2).

PENDING TO DEFINE

4.3.2 `lce_user_invoice` & `lce_user_invoice_line`

Existing columns handle most needs (sub_total, promo_amount, etc.) but add:

- `order_type` (same enum as above).
- `subscription_id` (FK to `lce_user_subscriptions`) for invoices containing subscription charge.
- `is_subscription_invoice` TINYINT(1).

Invoice lines:

- Use `type` to distinguish:
  - `WF`, `DC`, `HD`, `FEE_PND`, `FEE_SERVICE`, `SUBSCRIPTION_BAG`, `SUB_OVERWEIGHT_LBS` etc.

PENDING TO DEFINE

4.3.3 `lce_user_info`

Add fields:

- `default_order_type` ENUM('PPO','subscription') – what to preselect.
- `subscription_id` (optional pointer to current `lce_user_subscriptions.id`).

PENDING TO DEFINE

---

5. Business Logic

5.1 Welcome Credit ($20)

- Triggered when:
    - New user signs up via web with email/Google/Apple.
    - Implementation:
        - Create `lce_user_credits` row:
            - `type='welcome'`.
            - `amount=20`.
            - `balance=20`.
    - Usage:
        - During invoice creation for first PPO or subscription order:
            - Deduct from `balance`, mark credit as partially/fully used.
            - Support multiple credits (e.g., promo + welcome).

5.2 PPO Pricing Logic

Given:

- `weight_lbs` (sum of all W&F weight in order).
- `PPO_rate_per_lb` (from config table `lce_configurations` or `lce_prices`).
- `pn_d_fee` and `service_fee` from config.
- `minimum_laundry_amount_usd` = 30.

Algorithm:

1. `wf_amount = max(weight_lbs * rate_per_lb, minimum_laundry_amount_usd)`.
2. `order_base_total = wf_amount + pnd_fee + service_fee`.
3. Add price-list based items (DC, HD, LP) as additional invoice lines.
4. Apply credits/promos.
5. Store totals in `lce_user_invoice` and line details in `lce_user_invoice_line`.

5.3 Subscription Create New

For **subscription CREATE (if CC on file and successful charge of payment amount)**:

Monthly	**Yearly**

TABLE: `lce_user_subscriptions`
 	
`INSERT`
`user_id: 117530`
`plan_id: 1`
`status: active`
`billing_cycle: 'monthly'`
`start_date: '2025-12-12'`
`end_date: '2026-01-12'`
`next_renewal_date: '2026-01-12'`
`bags_plan_period: 1`
`bags_plan_total: 1`
`bags_plan_balance: 1`
`bags_available: 1`
`payment_last: {amount}`
	TABLE: `lce_user_subscriptions`
	
	`INSERT`
	`user_id: 117530`
	`plan_id: 5`
	`status: active`
	`billing_cycle: 'yearly'`
	`start_date: '2025-12-12'`
	`end_date: '2026-12-12'`
	`next_renewal_date: '2026-12-12'`
	`bags_plan_period: 1`
	`bags_plan_total: 12`
	`bags_plan_balance: 12`
	`Bags_available: 1`
	`payment_last: {amount}`

payment_discount: 0.00 payment_balance: {amount} mdate: NOW() cdate: NOW() **AMOUNT:** bags_per_month * price_per_bag	payment_discount: {discount} payment_balance: {amount} mdate: NOW() cdate: NOW() **AMOUNT:** (bags_per_month * 12 * price_per_ba (100 - annual_discount) / 100 **DISCOUNT:** (bags_per_month * 12 * price_per_ba (annual_discount) / 100
TABLE: lce_user_transactions INSERT user_id: 117530 Type: Debit subscription_id: 12345 transactionId = {cc_trans_id} Name: Credit Card amount: {amount} description: Subscription Payment mdate: NOW() Cdate: NOW() **AMOUNT:** bags_per_month * price_per_bag	**TABLE:** lce_user_transactions INSERT user_id: 117530 Type: Debit subscription_id: 12345 transactionId = {cc_trans_id} Name: Credit Card amount: {amount} description: Subscription Payment mdate: NOW() Cdate: NOW() **AMOUNT:** (bags_per_month * 12 * price_per_ba (100 - annual_discount) / 100

For **subscription CREATE (no CC on file)**:

Monthly	**Yearly**
TABLE: lce_user_subscriptions INSERT user_id: 117530	**TABLE:** lce_user_subscriptions INSERT user_id: 117530

`plan_id: 1`
`status: pending`
`billing_cycle: 'monthly'`
`start_date: '2025-12-12'`
`end_date: '2026-01-12'`
`next_renewal_date: '2026-01-12'`
`bags_plan_period: 1`
`bags_plan_total: 1`
`bags_plan_balance: 0`
`payment_last: 0`
`payment_balance: 0`
`mdate: NOW()`
`cdate: NOW()`
	`plan_id: 5`
	`status: pending`
	`billing_cycle: 'yearly'`
	`start_date: '2025-12-12'`
	`end_date: '2026-12-12'`
	`next_renewal_date: '2026-12-12'`
	`bags_plan_period: 1`
	`bags_plan_total: 12`
	`bags_plan_balance: 0`
	`payment_last: 0`
	`payment_balance: 0`
	`mdate: NOW()`
	`cdate: NOW()`

5.4 Subscription Cancel

For **subscription CANCEL**:

Monthly	**Yearly**

TABLE: `lce_user_subscriptions`
 	
`UPDATE`
`id = 12345`
`status: cancelled`
`mdate: NOW()`
	TABLE: `lce_user_subscriptions`
	
	`UPDATE`
	`id = 12345`
	`status: cancelled`
	`mdate: NOW()`

TABLE: `lce_user_transactions`
 	
`INSERT`
`user_id: 117530`
`Type: Debit`
`subscription_id: 12345`
`transactionId = {cc_trans_id}`
`Name: Credit Card`
`amount: {refund}`
`description: Subscription Refund`
	TABLE: `lce_user_transactions`
	
	`INSERT`
	`user_id: 117530`
	`Type: Debit`
	`subscription_id: 12345`
	`transactionId = {cc_trans_id}`
	`Name: Credit Card`
	`amount: {refund}`
	`description: Subscription Refund`

mdate: NOW() Cdate: NOW() **REFUND:** if payment_balance > 0.00 and NOW() - start_date < 5 days **refund** = payment_balance	mdate: NOW() Cdate: NOW() **REFUND:** if payment_balance > 0.00 and NOW() - start_date < 5 days **refund** = payment_balance if payment_balance > 0.00 and NOW() - start_date > 5 days **refund** = max(0, payment_balance - 100.00)

5.5 Subscription Upgrade/Downgrade

For **subscription upgrade and downgrade**:

Monthly	**Yearly**
TABLE: lce_user_subscriptions UPDATE id = 12345 status: upgraded mdate: NOW()	**TABLE:** lce_user_subscriptions UPDATE id = 12345 status: upgraded mdate: NOW()
TABLE: lce_user_subscriptions INSERT user_id: 117530 plan_id: 1 status: active billing_cycle: 'monthly' start_date: '2025-12-20' end_date: '2026-01-20' next_renewal_date: '2026-01-20' bags_plan_period: 1	**TABLE:** lce_user_subscriptions INSERT user_id: 117530 plan_id: 5 status: active billing_cycle: 'yearly' start_date: '2025-12-20' end_date: '2026-12-20' next_renewal_date: '2026-12-20' bags_plan_period: 1

bags_plan_total: 1 bags_plan_balance: 1 bags_available: 1 payment_last: {amount} payment_discount: 0.00 payment_balance: {amount} mdate: NOW() cdate: NOW() **AMOUNT:** max(0,(bags_per_month * price_per_bag) - previous_sub->payment_balance) (Before max) positive: Charge CC (Before max) negative: Refund	bags_plan_total: 12 bags_plan_balance: 12 Bags_available: 1 payment_last: {amount} payment_discount: {discount} payment_balance: {amount} mdate: NOW() cdate: NOW() **AMOUNT:** max(0,((bags_per_month * 12 * price_per_bag) * (100 - annual_discount) / 100) - previous_sub->payment_balance) (Before max) positive: Charge CC (Before max) negative: Refund **DISCOUNT:** max(0,((bags_per_month * 12 * price_per_bag) * (annual_discount) / 100) - previous_sub->payment_balance)
TABLE: lce_user_transactions INSERT user_id: 117530 Type: Debit subscription_id: 12345 transactionId = {cc_trans_id} Name: Credit Card amount: {refund} description: Subscription Refund mdate: NOW() Cdate: NOW() **REFUND:** abs((bags_per_month * price_per_bag) - previous_sub->payment_balance)	**TABLE:** lce_user_transactions INSERT user_id: 117530 Type: Debit subscription_id: 12345 transactionId = {cc_trans_id} Name: Credit Card amount: {refund} description: Subscription Refund mdate: NOW() Cdate: NOW() **REFUND:** abs((bags_per_month * 12 * price_per_bag) * (100 - annual_discount) / 100) - previous_sub->payment_balance))

Subscription Logic

- Determine **refund amount** for current subscription:
    - `bags_included = plan.bags_per_month + banked_bags`.
    -
- When a pickup is processed:
        - `bags_used += bags_this_pickup`.
        - If `bags_used <= bags_included`:
        - No extra bag charge (the bag is “included”).
        - If `bags_used > bags_included`:
        - `bags_extra = bags_used - bags_included` (excess).
- Extra bags charged at **plan.price_per_bag**.
- For overweight:
- If any bag weight > `bag_capacity_lbs`:
- `overweight_lbs = sum(max(0, bag_weight - bag_capacity_lbs))`.
- Charge `overweight_lbs * PPO_rate_per_lb`.
- Banking:
- At period end:
- `unused_bags = bags_included - bags_used` (if positive).
- Add to `bags_bank_balance`.

- Invoicing:
  - Create invoice lines:
    - `type='SUBSCRIPTION_BAG'` with count = `bags_used_in_period` (or aggregated per month).
    - `type='SUB_OVERWEIGHT_LBS'` with `quantity = overweight_lbs`, price per lb.

5.4 Annual Subscription Billing

- When user chooses annual billing:
  - Compute:
    - `annual_total = monthly_price_per_bag * bags_per_month * 12`.
    - `discounted_price = annual_total * (1 - 0.15)`.
  - Charge once (create invoice & transaction).
  - Set `annual_paid=1`, `next_renewal_date = start_date + 12 months`.
- Cancellation:
  - Calculate unused portion (remaining months).
  - Refund = `unused_portion - max(0.15 * original_amount, $100)`.
  - Create negative invoice/credit.

---

6. Users Backend (Admin / CSR Portal)

7.1 Purpose

Internal web app for:

- Customer service reps (CSR).
- Operations staff.
- Accounting / finance.
- Sales for business services.

7.2 Authentication & Roles

- Reuse `f29om_users` with roles assigned via:
  - New table `lce_user_roles` or use existing `params` JSON/ACL.
- Roles:
  - `CSR`, `Operations`, `Accounting`, `Admin`.

7.3 Screens

7.3.1 Dashboard

- KPIs:
  - Today's pickups/deliveries.
  - Orders by status.
  - New vs returning customers.
  - Subscription metrics (active subs, bag usage).
- Quick links:
  - Search customer.
  - Add manual credit.
  - Create CS ticket.

7.3.2 Customer Search

- Filter by:
  - Name, email, phone.
  - ZIP, city, group.
  - Subscription status.
- Results:
  - Basic info with icons for active sub, recurring schedule, vacation, etc.

7.3.3 Customer Detail View

Tabs:

1. **Overview**
    - Contact info, address, zone details.
    - Subscription summary.
    - Recurring schedule + vacation status.
    - Active credits.
2. **Orders / Pickups**
    - List `lce_user_pickup` records.
    - For each:
        - Pickup & delivery timestamps.
        - Weight, items, site, driver.
        - Link to invoice.
3. **Invoices**
    - List `lce_user_invoice`.
    - Show status, amount, discounts, promos.
4. **Transactions**
    - `lce_user_transactions` and `lce_payment` entries.
5. **Subscription**
    - Current plan, bag usage details (from `lce_subscription_bag_usage`).
    - Buttons:
        - Change plan.
        - Pause / Resume.
        - Cancel subscription.
        - Apply manual bag credits.
6. **Preferences & Comms**

- From `lce_user_info`:
  - Laundry preferences.
  - Driver/laundry instructions.
- From `lce_communication_settings`:
  - Opt-in/out for SMS/email per event type.

7. **CS Tickets**
   - `lce_user_cs` & `lce_user_cs_log`.
   - Add CS note, change status.

7.3.4 Group / Business Management

- List all group admins (`lce_user_group_admin`).
- Drill into:
  - Members (`lce_user_group_members`).
  - Transaction history (`lce_user_group_members_history`).
  - Group-level limits & pricing.

7.3.5 Pricing Management

- Manage `lce_prices_lists` & `lce_prices`:
  - UI to edit price list names, default per-zip.
  - Manage items (SKU, type, name, description, prices).
- Manage `lce_subscription_plans`:
  - Add/edit plans, per-bag price, bag count, annual discount.

7.3.6 Scheduling & Operations

- Calendar view:
  - Pickups/deliveries by zone and driver.

- Manage:
  - `lce_pickup_zones` – add/edit ZIPs, days, geometry.
  - `lce_pickup_nonworking_days` – non-working dates by area.
  - `lce_processing_sites` – assign price lists to sites.

7.3.7 Promo & Credit Management

- Manage `lce_promo_codes`:
  - Types: percent, fixed amount, subscription-limited, etc.
- Issue manual credits:
  - Create `lce_user_credits` entry.
  - Choose type and optional expiry.

---

8. Migration & Backwards Compatibility

8.1 User & Order Data

- Existing users in `f29om_users` and `lce_user_info` remain valid.
- For existing orders:
  - `order_type` default = `PPO`.
- Historical behavior of fees and pricing remains unaffected.

8.2 Subscription Introduction

- Initially, `lce_subscription_plans` can be populated with:
  - 1/2/4/8 bag monthly plans only.
- `lce_user_subscriptions` will be empty and filled as new users subscribe.

8.3 Configurations

- Use `lce_configurations` to store:
  - PPO per-lb rate.
  - Minimum laundry charge.
  - P&D and service fees.
  - Bag capacity threshold.
  - Daily cutoff times for scheduling.
- These configurations will be read by API services instead of hardcoding values.

---

9. Non-Functional Requirements

9.1 Security

- Passwords stored hashed (e.g., bcrypt/argon2).
- All auth/API endpoints via HTTPS.
- PCI-compliant card storage via payment gateway; no raw card numbers in DB.
- RBAC for Users Backend.

9.2 Logging & Analytics

- Log key events:
  - Signup, login.
  - Pickup created/updated.
  - Subscription created/cancelled.
  - Payments and refunds.
- Instrument front-end for analytics (GA4, etc.) with events like:
  - `signup_started`, `signup_completed`, `order_scheduled`, `sub_selected`, `sub_cancelled`.

9.3 Performance & Scalability

- Indexes:
  - `user_id` on all user-related tables (`lce_user_pickup`, `lce_user_invoice`, `lce_user_transactions`, etc.).
  - Composite indexes for subscription period queries.
- All queries must be written in a way that supports growth to tens of thousands of active users.

---

10. Implementation Checklist (High-Level)

1. **Database**
   - Create new tables: `lce_subscription_plans`, `lce_user_subscriptions`, `lce_subscription_bag_usage`, `lce_user_credits`.
   - Apply ALTERs to existing tables for fields described above.
   - Seed initial configuration and subscription plans.
2. **Backend APIs**
   - Implement auth, address, zone, scheduling, pricing, subscription, billing APIs as specified.
   - Integrate with payment provider.
   - Add credit consumption logic to invoice creation.
3. **Web Frontend**
   - Implement new navigation and pages:
     - Home (with pop-up promo and video).
     - Services & Prices.
     - Subscribe & Save Savings.
     - Business Services.
   - Implement 3-step signup + scheduling flow.
   - Logged-in dashboard with subscription & order management.
4. **Users Backend**

- Implement separate SPA (or module) with:
  - Dashboard.
  - Customer search & detail.
  - Group management.
  - Pricing & plan management.
  - Scheduling calendar.
  - Promo & credit management.
  - CS ticket console.

5. **Testing**
   - Unit tests for pricing and subscription logic (bag usage, overages, banking).
   - Integration tests for the 3-step flow and subscription change cancellations.
   - End-to-end tests for scheduling and invoice generation.