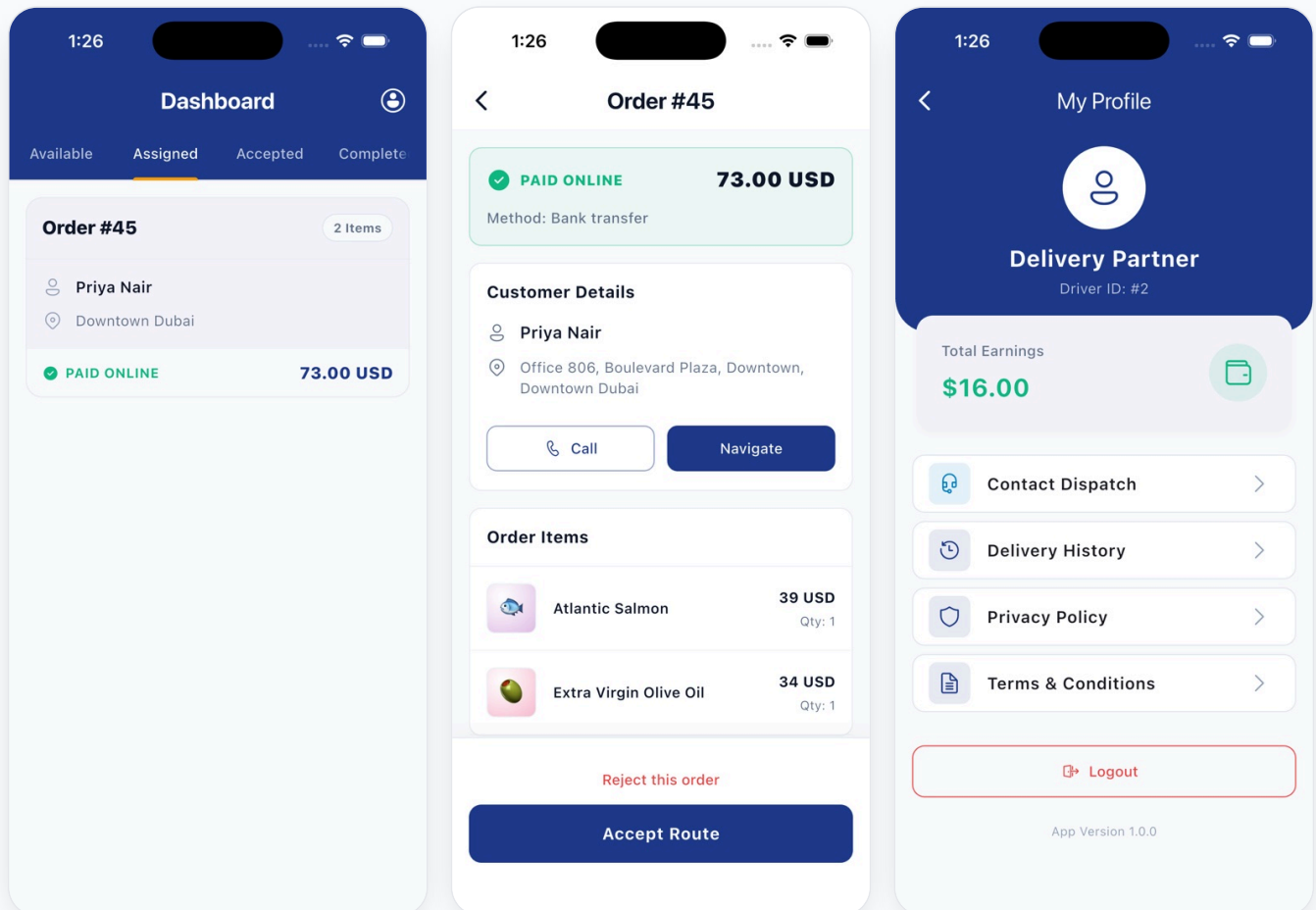


Delivery Driver App

A step-by-step guide for everyone, no coding needed: add drivers, set the rules, assign orders, set up push notifications and build your own Android app online. Developers will also find the source-code and REST API chapters.

1. Start here

In plain words: This product lets you hand **WooCommerce orders to delivery drivers**. You pick a driver on the order screen, the driver gets a notification on their phone, drives the order to the customer and marks each step in a small app. You see everything live in WordPress.



The driver app on a phone: the order list, an order, and the driver's earnings.

What you bought: three parts that work together

Part	What it is	Who needs it
1. The plugin delivery-driver-app-for-woocommerce-plugin.zip	Goes on your WordPress store. You assign orders, choose the rules, set up customer text messages and push messages, and build the app, all in one place.	Everybody
2. The online App Builder (a tab inside the plugin)	Makes your own branded Android app (your name, icon and colours) on our servers. You fill in a short form and press one button.	Most people
3. The app source code flutter-app-source.zip	The complete code of the driver app (Flutter, for Android and iPhone). Use it if you want to change the app or build it yourself.	Developers

Tip: You can start without any programming: install the plugin, add drivers, build the app online, install it on a driver's phone. Chapters 4 to 12 are written for shop owners. Chapters 13 and 14 are for developers.

2. How a delivery works

In plain words: Every order goes through the same four steps. The driver can only do the next step, so nothing gets skipped.

Step	Who	What happens
Assigned	You (or the driver, from the open pool)	The driver receives a push notification "New order assigned".
Accepted	Driver taps <i>Accept Route</i>	You know the driver took the job.
Picked up	Driver taps <i>Mark as Picked Up</i>	The customer can get an "on the way" text message.
Completed	Driver taps <i>Complete Delivery</i> (and takes a photo if you require one)	The WooCommerce order becomes <i>Completed</i> , the driver's commission is paid, the customer gets a "delivered" message.

Optional extras you can switch on or off: drivers can **claim** orders nobody has taken (the "open pool"), drivers can **reject** an order, and a **delivery photo** can be compulsory.

Important: A driver only ever sees the orders assigned to them (and, if you allow it, the open pool). Another driver's orders, wallet and position are never shown to them.

3. What you need before you start

You need	Why
A WordPress site with WooCommerce 8.0 or newer , PHP 7.4+	The plugin reads and updates your WooCommerce orders.
HTTPS (a padlock in the browser) and “pretty” permalinks	The phone talks to your store through https://yourstore.com/wp-json/ .
A free Firebase project (Google account)	Needed for push notifications. Chapter 7 shows every click.
Optional: the free TeraWallet plugin	Pays the driver’s commission into a wallet the driver can see in the app.
Optional: an account with an SMS provider (Twilio, Unifonic, 2Factor or OTPless)	Sends “on the way” and “delivered” messages to customers.
Your purchase code	Needed by the online App Builder (chapter 12).

4. Install the plugin

- 1 Unzip the package you downloaded. Inside you will find [delivery-driver-app-for-woocommerce-plugin.zip](#) and [flutter-app-source.zip](#). Do **not** unzip these two.
- 2 In WordPress open **Plugins → Add New → Upload Plugin**, choose [delivery-driver-app-for-woocommerce-plugin.zip](#), press **Install Now**, then **Activate**.
- 3 Open **Settings → Delivery App**. If you see the page with five tabs (Operations, Mobile App, SMS & WhatsApp, Push Notifications, App Builder), the plugin is ready.
- 4 Check **Settings → Permalinks**: anything except “Plain” is fine. Press *Save Changes* once if the app cannot reach your store.

Important: If you used an older “Delivery Boy for WooCommerce” plugin, or the separate “Taqnix App Builder” plugin, read chapter 17 first: this plugin replaces both.

5. Create your drivers

In plain words: A driver is just a WordPress user with the role **Delivery Boy**. The driver logs into the app with that username and password.

- 1 Open **Users → Add New**.

- 2 Fill in a username (for example `maria`), an email, the name and a strong password.
- 3 Set **Role** to **Delivery Boy** and press *Add User*.
- 4 Give the username and password to the driver. That is all.

Add User

Create a brand new user and add them to this site.

Username (required)

Email (required)

First Name

Last Name

Website

Password (required)

Hide

Repeat Password (required)

Type the password again.

Send User Notification

☒ Send the new user an email about their account

Role

Subscriber ▼

Add User

Add a driver: a normal new user with the role "Delivery Boy".

Tip: Wrong password five times in a row locks that username on that phone for 15 minutes. Use *Users* → *Edit* to reset a password at any time.

6. The settings, tab by tab

In plain words: Open **Settings** → **Delivery App**. Change what you need, press **Save Settings**. The app reads these rules when it starts, so drivers see the change the next time they open the app.

Operations: the rules of your delivery business

Delivery App Settings

Operations

Mobile App

SMS & WhatsApp

Push Notifications

App Builder

Operational Rules

Delivery History (Days)

1

How many days of completed orders should show in the app? (1 = today only, 7 = the last 7 days)

Driver Commission per Delivery

8

\$

Paid once into the driver's TeraWallet when an order is completed. Use 0 to pay nothing. (Needs the free TeraWallet plugin.)

Allow Self-Assignment (Open Pool)

☒ Turn this on to let drivers choose their own deliveries from an "Open Pool" of unassigned orders.

Allow Order Rejection

☒ Turn this on to allow drivers to say "No" to an order assigned to them.

Require Photo Proof

☒ Force the driver to take a picture (Proof of Delivery) before they can mark an order as completed.

Dispatch Hotline Phone Number

+971 4 555 0100

The main phone number for your dispatcher. Drivers can tap a button in the app to call this number if they have a problem.

Save Settings

Operations tab.

Setting	What it does
Delivery History (Days)	How many days of finished deliveries the driver can see. 1 = today only, 7 = the last week.
Driver Commission per Delivery	An amount paid into the driver's TeraWallet once when the order is completed. 0 = pay nothing.
Allow Self-Assignment (Open Pool)	Adds an Available tab to the app with processing orders nobody has taken. A driver taps <i>Claim Order</i> to take one.
Allow Order Rejection	Shows a <i>Reject this order</i> button on newly assigned orders. A rejected order goes back to "unassigned" (and will not be offered to the same driver again).
Require Photo Proof	The driver must take a photo before an order can be completed.
Dispatch Hotline Phone Number	Shown in the app under <i>Contact Dispatch</i> ; the driver taps to call you.

Mobile App

Delivery App Settings

- Operations
- Mobile App
- SMS & WhatsApp
- Push Notifications
- App Builder

Mobile App Configuration

Force App Update

Enter the lowest allowed version (like "1.0.5"). Anyone using an older version will be locked out and forced to update the app.

App Update Link

Where drivers go when an update is required - your Google Play (or App Store) page. Shown on the "Update Required" screen.

Privacy Policy Web Link

The web link to your Privacy Policy (e.g., <https://yourwebsite.com/privacy>).

Mobile App tab.

Setting	What it does
Force App Update	The lowest app version allowed, for example 1.2.0 . Older apps show an "Update Required" screen and cannot be used.
App Update Link	Where the "Update Now" button goes: your Google Play or App Store page.
Privacy Policy / Terms links	Opened from the driver's profile screen.

SMS & WhatsApp

Delivery App Settings

Operations

Mobile App

SMS & WhatsApp

Push Notifications

App Builder

Customer SMS & WhatsApp

Enable Automated Texts ☐ Automatically send text messages to customers when their order status changes.

Your SMS Provider Twilio (Global SMS & WhatsApp) ▼

API Key (or Client ID)
Your main API Key or Client ID from your SMS provider's dashboard.

API Secret (or Auth Token)
Your secret password or Auth Token from your SMS provider. (Leave blank if your provider doesn't use one).

Sender Name / Phone Number
Who is the message from? Enter your brand name (e.g., "TAQNIX") or your provider phone number (e.g., "whatsapp:+123456789").

Message: Order is on the way
Sent when the driver starts the route. Use {order_id}, {customer_name}, and {total} to automatically insert details.

Message: Order successfully delivered

SMS & WhatsApp tab.

Switch on **Enable Automated Texts**, choose your provider, paste the key (and secret, if your provider has one), type the sender name or number, and edit the two messages. In the messages you can use `{order_id}`, `{customer_name}` and `{total}`. The “on the way” message is sent when the driver taps *Mark as Picked Up*; the “delivered” message when the order is completed.

Push Notifications

Delivery App Settings

Operations

Mobile App

SMS & WhatsApp

Push Notifications

App Builder

Firestore Cloud Messaging (Push Notifications)

Firestore Service Account
JSON

```
{
  "type": "service_account",
  "project_id": "your-app-name-123",
  "private_key_id": "a1b2c3d4e5...",
  "private_key": "-----BEGIN PRIVATE KEY-----\nYour Very Long Secret Key Here...\n-----END PRIVATE KEY-----\n",
  "client_email": "firebase-adminsdk@your-app.iam.gserviceaccount.com"
}
```

Firestore connects your website to the driver's app for push notifications.

Save Settings

Push Notifications tab: paste the Firestore service account file here.

This is where your Firestore key goes (chapter 7).

Tip: Secrets such as the SMS secret are never shown again after you save them. Leave the box empty to keep the saved one.

7. Set up push notifications (Firestore)

In plain words: Push notifications are the “ding” on the driver’s phone when you assign an order. Google provides them for free through **Firestore**. You set this up once. It needs two files: one tells the *app* which Firestore project to use, the other lets your *store* send messages.

A. Create the Firestore project

- 1 Go to console.firebase.google.com and sign in with a Google account.
- 2 Press **Add project**, type a name (for example “Acme Drivers”), and finish the wizard. You can turn Google Analytics off.

B. The file for the app: `google-services.json` (Android)

- 1 In your project press the **Android** icon (“Add app”).

- 2 **Android package name:** type the unique name of your app, for example `com.acme.drivers` . Write it down: you must use **exactly** the same name in the App Builder (chapter 12).
- 3 Press **Register app**, then **Download google-services.json**. Keep this file for chapter 12 or 13. You can skip the remaining Firebase screens.

C. The file for your store: a service account key

- 1 In Firebase press the gear icon → **Project settings** → tab **Service accounts**.
- 2 Press **Generate new private key** and confirm. A `.json` file downloads.
- 3 Open that file in a text editor, select **everything** and copy it.
- 4 In WordPress open **Settings → Delivery App → Push Notifications**, paste into the box, press **Save Settings**.

Important: The service-account file is a password for your Firebase project. Do not email it or put it in a public place.

D. iPhone only

For iPhone also add an **iOS** app in Firebase (use your bundle ID), download `GoogleService-Info.plist` , and upload an Apple **APNs key** under *Project settings → Cloud Messaging*. Details are in chapter 13.

Test it

Log into the app on a phone (chapter 9), then assign that driver an order (chapter 8). The phone should ring within seconds. The order notes on the order screen say "Push notification sent to driver..." or show the exact error from Google if something is wrong.

8. Give an order to a driver

In plain words: Open any order. On the right you will find a box called **Assign Delivery Boy**. Pick a driver and you are done. It saves immediately, no need to press Update.

Edit order

Add new order

Order #43 details

Payment via Direct bank transfer. Paid on October 22, 2026 @ 9:03 am

General

Date created:

2026-10-22 @ 09 : 03

Status:

Completed

Customer:

Billing

Layla Hassan
Marina Promenade, Tower 3, Apt 1204
Dubai Marina
DU

Email address:
layla@example.com

Phone:
+971 50 123 4567

Shipping

Layla Hassan
Marina Promenade, Tower 3, Apt 1204
Dubai Marina
DU

Phone:
+971 50 123 4567

Order actions

Choose an action...

Move to Trash

Update

Order attribution

Origin

Unknown

Customer history

Total orders 1

Total revenue \$37.00

Average order value \$37.00

Assign Delivery Boy

Item	Price	Qty	Total
 Fresh Red Apples	\$6.50	x 2	\$13.00
 Whole Milk	\$7.50	x 2	\$15.00

The order screen: the Assign Delivery Boy box is in the right column.

- 1

Open **WooCommerce** → **Orders** and click an order that is *Processing*.
- 2

In **Assign Delivery Boy**, open the list and choose the driver.
- 3

The status line shows “Assigned” and the driver gets a push message.

You can change the driver as long as the delivery is not finished. The new driver starts again at “Assigned”. Choosing the empty line removes the driver.

What you see on the order afterwards

Item	Meaning
Current Status	Assigned, Accepted, Picked up or Completed.
Track Driver on Map	Opens Google Maps at the driver’s last position, with the time of the last update.
Proof of Delivery	The photo (and signature) the driver uploaded.
Order notes	Every step is written to the order notes, for example “Order assigned to Ahmed Khan”.

Assign Delivery Boy

Select Driver:

Ahmed Khan

Current Status: **Completed**

Track Driver on Map

Last updated: Oct 22, 9:11 am

Proof of Delivery

Delivery Photo:



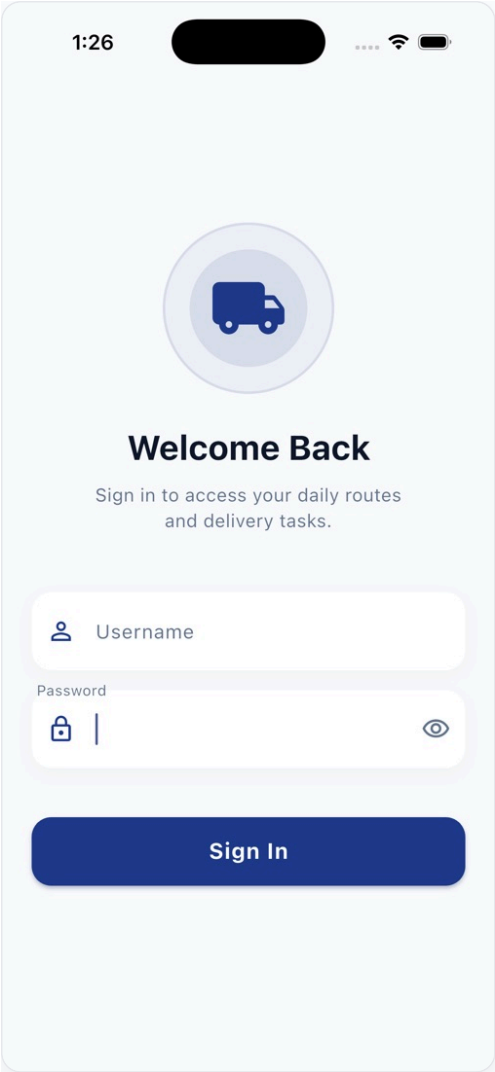
A finished delivery: status, driver position and the delivery photo.

9. The driver's guide (give this to your drivers)

In plain words: The app has four tabs and one button per order. The driver always sees the single next step.

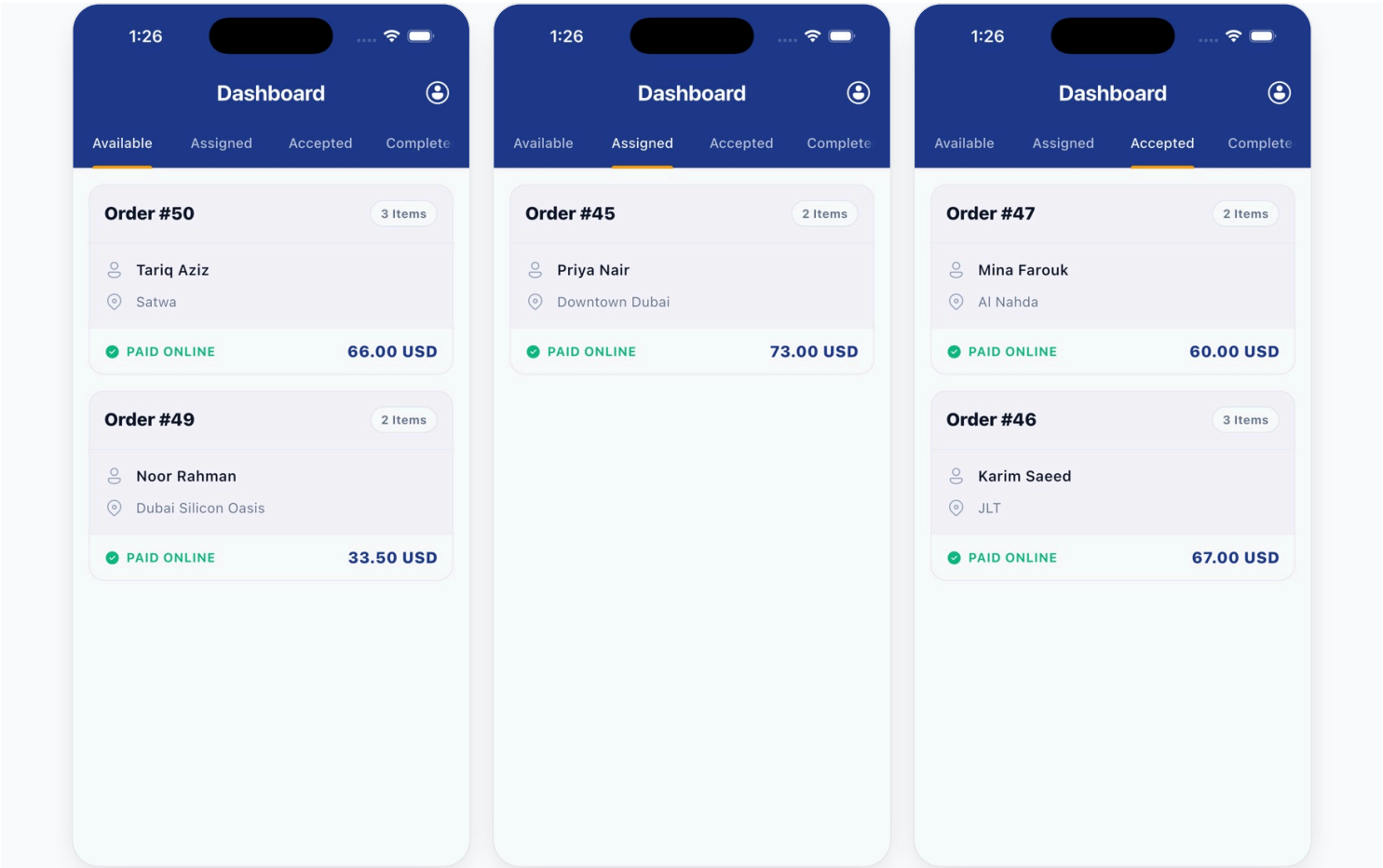
Install and log in

- 1 Install your app (the APK file from the App Builder email, or from Google Play).
- 2 Open it, allow **notifications** and **location** when asked, and log in with the username and password you created.
- 3 The dashboard opens. The driver stays logged in until they press *Logout* or you change their password.



The login screen.

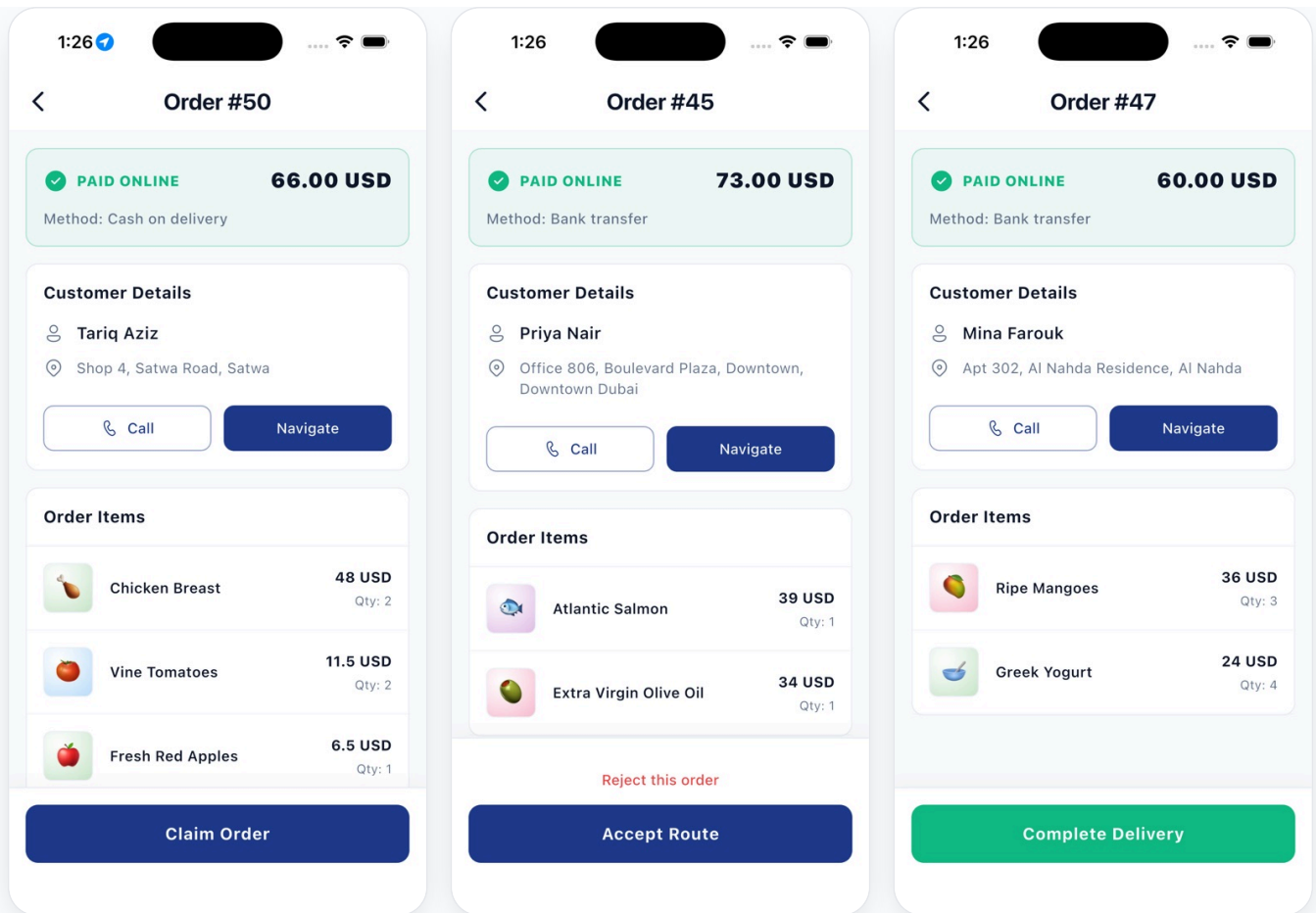
The dashboard



Tabs: Available (only if the open pool is on), Assigned, Accepted and Completed.

Tab	Contains
Available	Orders nobody has taken yet. Tap one, then <i>Claim Order</i> .
Assigned	New orders for this driver.
Accepted	Orders the driver has accepted or picked up.
Completed	Today's deliveries (or the last days you chose in Operations).

Doing a delivery

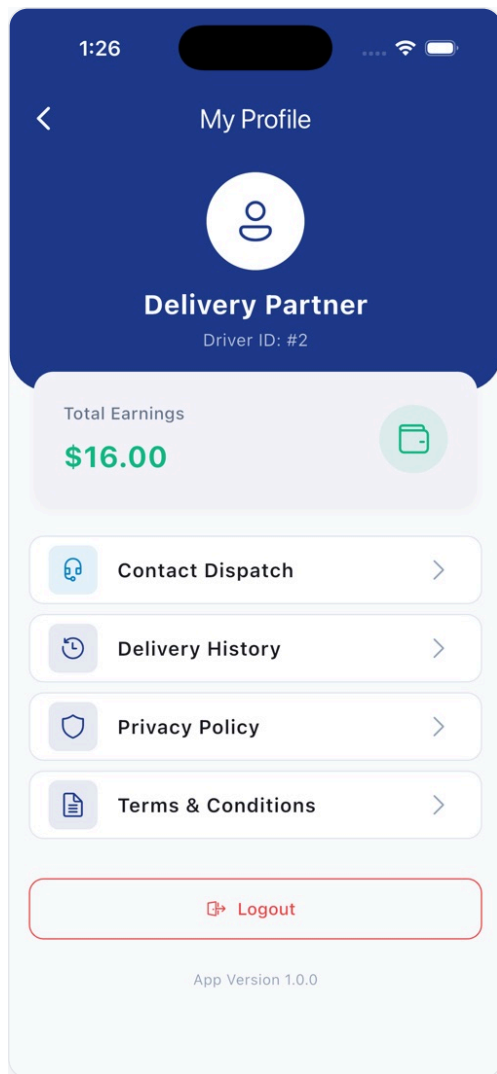


The same order screen at different steps. The big button at the bottom always shows the next action.

- 1 Tap the order. **Call** rings the customer, **Navigate** opens the address in the maps app.
- 2 Tap **Accept Route** (or *Reject this order*, if you allowed it).
- 3 At the shop tap **Mark as Picked Up**. The customer can now get the “on the way” text.
- 4 At the door tap **Complete Delivery**. If the customer has not paid yet (cash on delivery) the button says **Collect Cash & Complete**. If you turned on photo proof, the camera opens: take a photo of the parcel at the door and the delivery is completed.

Earnings and profile

The profile screen (person icon, top right) shows **Total Earnings** from TeraWallet, the dispatch phone number, delivery history, privacy and terms, the app version and **Logout**.



Profile and earnings.

Tip: The app shares the driver's position while it is open, about every 10 minutes. It does not track a phone that is closed or logged out.

10. Driver commission and earnings

Set an amount under **Operations → Driver Commission per Delivery**. When a delivery is completed the plugin adds that amount to the driver's **TeraWallet** balance. It is paid **once per order**, even if the order is edited afterwards. The order notes show "Credited ... to driver". The driver sees the balance as *Total Earnings* in the app.

Install and activate the free **TeraWallet** plugin (WooCommerce wallet) for this. Without it deliveries still work; the order note says "Commission not paid: TeraWallet is not active" and the app shows 0.00. To pay your drivers out, use TeraWallet's own screens (*Wallet* menu → adjust or withdraw).

11. Messages to customers

Turn on **Settings → Delivery App → SMS & WhatsApp → Enable Automated Texts**. The customer's billing phone number is used, so make it a required checkout field and ask for the country code.

Provider	Good for	Key / Secret
Twilio	Worldwide SMS and WhatsApp	Account SID / Auth Token; sender = your Twilio number (for WhatsApp: <code>whatsapp:+1...</code>)
Unifonic	Saudi Arabia, UAE, Gulf	AppSid; sender = approved sender name
2Factor	India transactional SMS	API key; sender id
OTPlless	WhatsApp messages	Client ID / Client Secret

Tip: If messages do not arrive, open the order: a failed send is written to the order notes with the answer of your provider.

12. Build your own branded app online (no coding)

In plain words: The **App Builder** tab makes an Android file (APK) with *your* app name, icon and colours, pointing at *your* store. You fill in a form; a few minutes later you get an email with the download link. You do not need to install anything on your computer.

Delivery App Settings

Operations

Mobile App

SMS & WhatsApp

Push Notifications

App Builder

Connected. Your app settings are loaded below.

Step 1 · Connect your purchase code

Enter your website address and the purchase code you received with your order. Your code is stored on this server and is never shown again.

Website address

https://grocer.example

Purchase code

..... saved - leave blank to keep it

Connect & load my app

● Connected

Step 2 · Name your app

App name

Grocer Drivers

The name under the icon on the driver's phone, for example "Acme Drivers".

Package name

com.grocer.drivers

A unique ID for Google Play: lowercase letters and dots, like com.yourbrand.drivers. You cannot change it after publishing.

Your email

dispatch@grocer.example

We email you the download link when your app is ready.

Phone

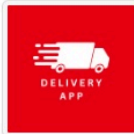
AE

+971 4 555 0100

Step 3 · Add your branding

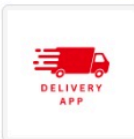
App icon 1024x1024

Choose File No file chosen



Logo

Choose File No file chosen



Shown in the middle of the screen while the app opens.

Splash image

Choose File No file chosen

Settings → Delivery App → App Builder.

Before you start

- Your **purchase code**.

- Your app **icon** (square PNG, 1024×1024), a **logo** and a **splash image**.
- The `google-services.json` file from chapter 7, made for the package name you will type below.

Step by step

- 1 Open **Settings → Delivery App → App Builder**.
- 2 **Step 1.** Check your website address (https, no slash at the end), paste your purchase code and press **Connect & load my app**. The green “Connected” sign appears. Your code is stored on your server and is not shown again.
- 3 **Step 2.** Type the **App name** (shown under the icon on the phone), the **Package name** (the same one you gave Firebase, for example `com.acme.drivers`) and your email. We send the download link to that email.
- 4 **Step 3.** Choose the icon, logo and splash images (they upload as soon as you pick them), choose the splash colour and upload `google-services.json` .
- 5 **Step 4.** Choose **APK** (to install straight on a phone and test) or **App Bundle** (to upload to Google Play). Press **Save details**, then **Start Android build**.
- 6 Wait for the email (usually a few minutes; builds are queued one after another). Open the link on the phone to download and install the APK.

Important: The package name in the builder and in Firebase must be **identical**. If they differ the build stops with a message that tells you which names it found.

Good to know

- The builder makes **Android** apps. For iPhone use the source code (chapter 13).
- Every purchase code can build one app. To change the name, icon or screens later, save the form again and press Build again.
- What is sent to the build server: your site address, purchase code, app name, package name, your email and phone, the splash colour, your three images and the Firebase file. No store passwords or WooCommerce keys are needed.
- Only administrators can use this tab, and every request is protected by a security token.

Install the APK on a phone: open the link in the phone’s browser, allow “install from this source” if Android asks, and open the app. Log in with a driver account from chapter 5.

13. For developers: build the app from source

In plain words: Skip this chapter if you use the online builder. The full source code of the driver app is in `flutter-app-source.zip` : Flutter (Dart), Dio for networking, Firebase Messaging, Geolocator, Secure Storage.

Requirements

- Flutter 3.41 or newer (`flutter doctor` should be green), Android Studio for Android, Xcode and CocoaPods for iPhone.

1. Open the project and point it to your store

```
unzip flutter-app-source.zip -d driver_app && cd driver_app
flutter pub get

# run on a connected phone or emulator
flutter run --dart-define=API_DOMAIN=https://yourstore.com
```

Or change the default permanently in `lib/core/constants/app_config.dart` :

```
class ApiConstants {
  static const String domain = String.fromEnvironment(
    'API_DOMAIN',
    defaultValue: 'https://yourstore.com', // <- your store, no slash at the end
  );
  ...
}
```

2. Add your Firebase files

- Android: `android/app/google-services.json` (the Gradle plugin switches on automatically when the file exists).
- iPhone: `ios/Runner/GoogleService-Info.plist` , then open `ios/Runner.xcworkspace` in Xcode and drag it into the Runner group if it shows in red.

3. Name, package and icons

```
# unique app id (must match Firebase)
dart run change_app_package_name:main com.acme.drivers

# replace assets/images/icon.png (1024x1024) and assets/images/splash.png, then:
dart run flutter_launcher_icons
dart run flutter_native_splash:create
```

The app name shown on the phone is in `android/app/src/main/AndroidManifest.xml` (`android:label`) and `ios/Runner/Info.plist` (`CFBundleDisplayName`). Brand colours are in `lib/core/constants/app_colors.dart` .

4. Sign and build (Android)

```
# create a signing key once and keep it safe
keytool -genkey -v -keystore ~/acme-drivers.jks -keyalg RSA -keysize 2048 -validity 10000 -alias upload

# android/key.properties
storeFile=/Users/you/acme-drivers.jks
storePassword=...
keyAlias=upload
keyPassword=...
```

In `android/app/build.gradle.kts` read these values in a `signingConfigs { create("release") { ... } }` block and use it in `buildTypes.release` instead of the debug key (the file has a comment where). Then:

```
flutter build apk --release --dart-define=API_DOMAIN=https://yourstore.com      # APK to test
flutter build appbundle --release --dart-define=API_DOMAIN=https://yourstore.com  # AAB for Google Play
```

5. iPhone

```
cd ios && pod install --repo-update && cd ..
flutter build ipa --release --dart-define=API_DOMAIN=https://yourstore.com
```

In Xcode choose your **Team**, set a unique **Bundle Identifier** (the same as in Firebase) and keep the *Push Notifications* capability. Upload an APNs key in Firebase for push.

Where things are

Folder	What is inside
<code>lib/core/network/dio_client.dart</code>	Network client; adds the login token to every request and logs the driver out when the server says the login is over.
<code>lib/features/auth</code>	Login, splash screen, session.
<code>lib/features/orders</code>	Dashboard, order list, order detail, photo proof, history.
<code>lib/features/profile</code>	Earnings, contact, legal links, logout.
<code>lib/core/services</code>	Location sharing, store settings, secure storage.
<code>default_files/</code>	Templates used by the online builder. Ignore.

14. For developers: the REST API

In plain words: The app and the plugin talk through a small REST API. You can use it to build your own driver app, a dispatch board or an integration.

Base address: `https://yourstore.com/wp-json/taqnix/v1` . Everything except `/login` and `/settings` needs the token you get from login, sent in the header `X-Driver-Token` (`Authorization: Bearer ...` works too). Tokens last 90 days, a driver can be logged in on up to 5 phones, and `/logout` ends one.

Method and path	What it does
<code>POST /login</code>	<code>username</code> , <code>password</code> → <code>boy_id</code> , <code>token</code> , name, email. 429 after too many wrong passwords.
<code>POST /logout</code>	Ends this token and stops pushes to this phone.
<code>GET /settings</code>	Public rules: minimum version, rejection, photo proof, open pool, dispatch phone, legal and update links.
<code>GET /orders</code>	The driver's active orders (+ the open pool when on).
<code>GET /history</code>	The driver's completed orders for the chosen number of days.
<code>POST /orders/{id}/status</code>	<code>status</code> = <code>assigned</code> (claim from the pool), <code>accepted</code> , <code>picked_up</code> or <code>completed</code> . Only the next step is allowed; otherwise 409.
<code>POST /orders/{id}/reject</code>	Optional <code>reason</code> . Only when rejection is on and the order is still "assigned".
<code>POST /orders/{id}/pod</code>	Multipart upload: <code>pod_photo</code> (JPG/PNG/WebP), optional <code>pod_signature</code> . Completes the order.
<code>POST /location</code>	<code>latitude</code> , <code>longitude</code> .
<code>POST /update-fcm</code>	<code>fcm_token</code> of the phone.
<code>GET /wallet</code>	Balance from TeraWallet (<code>balance_raw</code> , <code>balance_formatted</code>).

Try it

```
# 1. log in
curl -s -X POST https://yourstore.com/wp-json/taqnix/v1/login \
  -H 'Content-Type: application/json' \
  -d '{"username":"maria","password":"your-password"}'
# -> {"success":true,"data":{"boy_id":7,"token":"7.3f9c...","display_name":"Maria Lopez",...}}

# 2. list my orders
curl -s https://yourstore.com/wp-json/taqnix/v1/orders -H 'X-Driver-Token: 7.3f9c...'

# 3. accept an order
curl -s -X POST https://yourstore.com/wp-json/taqnix/v1/orders/123/status \
  -H 'X-Driver-Token: 7.3f9c...' -H 'Content-Type: application/json' -d '{"status":"accepted"}'
```

Assign an order from your own code

```
// PHP, inside WordPress
$order = wc_get_order( 123 );
$result = TDB_Flow::assign( $order, $driver_user_id ); // sends the push message too
if ( is_wp_error( $result ) ) {
    error_log( $result->get_error_message() );
}
```

Order data you can read

Meta key	Value
<code>_assigned_delivery_boy_id</code>	User ID of the driver.
<code>_delivery_status</code>	<code>assigned</code> , <code>accepted</code> , <code>picked_up</code> , <code>completed</code> .
<code>_tdb_pod_photo_id</code>	Media ID of the delivery photo.
<code>_tdb_commission_paid</code>	Commission paid for this order (empty = not paid).

15. Security and privacy

- **Every request needs a login token.** Without it the API answers "401". The driver id is taken from the token, never from the request, so one driver cannot read or change another driver's orders, wallet or position.
- Tokens are random and stored as a one-way hash. Logging out, deleting the user or removing the role ends access immediately.
- The step order is enforced on the server, commission is paid once per order, and uploads must be real JPG, PNG or WebP images.
- Login attempts are limited (5 wrong passwords, then 15 minutes).
- The App Builder tab and every admin action need an administrator *and* a security token; nothing is available to visitors.

- Data stored: the driver’s last 100 positions, push token and login tokens (user meta), delivery photos (media library), order notes. Ask your drivers for consent to location sharing and mention it in your privacy policy.

16. Troubleshooting

Problem	What to do
The app says “Cannot reach the server”	Check the store address in the app (<code>API_DOMAIN</code>), HTTPS, and open <code>https://yourstore.com/wp-json/taqnix/v1/settings</code> in a browser: you should see text starting with <code>{"success":true}</code> . If not, re-save <i>Settings</i> → <i>Permalinks</i> .
“Access denied” at login	The user’s role must be Delivery Boy (not Customer or Subscriber).
“Too many failed attempts”	Wait 15 minutes. It protects the account from password guessing.
The app suddenly shows the login screen	The login ended (logged out, password changed, 90 days passed). Log in again.
No push notification	Read the order notes: they show the answer from Google. Check the service-account file (chapter 7C), that the driver logged in on that phone at least once, and that notifications are allowed on the phone.
The driver does not see the order	The order must be <i>Processing</i> and assigned to that driver. Completed orders move to the Completed tab.
“Available” tab is missing	Switch on Allow Self-Assignment and restart the app.
Complete Delivery is refused with “photo required”	Photo proof is on. The app opens the camera; allow camera access in the phone’s settings.
Earnings stay 0.00	Install TeraWallet and set a commission amount. Orders completed before that are not paid retroactively.
The App Builder says “Not allowed”	Your admin session expired: reload the page.
The App Builder says the package name does not match	Make the package name in the builder identical to the one in Firebase, or download a new <code>google-services.json</code> .
Cache plugins (LiteSpeed, WP Rocket, Cloudflare)	Exclude <code>/wp-json/taqnix/</code> from caching. The app already asks for fresh data.

17. Updating, and moving from the older plugins

Updating

Upload the new plugin zip with **Replace current with uploaded**; your settings and drivers stay. Build a new app version (chapter 12 or 13) and ask drivers to update. Raise *Force App Update* to the new version number when you want everyone to update.

If you used “Delivery Boy for WooCommerce” (1.0.x) and “Taqnix App Builder”

- This plugin **replaces both**. Deactivate and delete “Taqnix App Builder”; its builder is now the *App Builder* tab here. Enter your purchase code there once.
- Upload this plugin over the old one. Your drivers, settings and orders are kept.
- **Old driver apps stop working on purpose**. Version 1.1 closed a serious hole: until then anybody who knew a driver’s id could read and change orders. The new app logs in with a token. Build and install the new app, then raise *Force App Update*.
- The commission used to be a fixed 15, paid again every time an order was saved as completed. It is now a setting (default 0) and is paid once per order.

Removing

Deactivate and delete the plugin. Your drivers (users), orders and photos stay. Plugin settings are removed.

18. Support

Installation help and bug fixes are included for 6 months, with lifetime updates. Contact us through the marketplace item page where you bought the package. Please include your store address, what you did, and what you saw on the screen (a screenshot helps).

Changelog

1.1.0 – Initial marketplace release

- Flutter driver app for Android and iOS: login, Available / Assigned / Accepted / Completed tabs, order details with Call and Navigate, claim and reject, photo proof, history, earnings, GPS sharing, forced update
- WordPress plugin 1.1.0: Assign Delivery Boy box on orders, push notifications (FCM v1), driver GPS and Track on Map, proof of delivery, TeraWallet commission, customer SMS / WhatsApp, open pool and rejection rules
- App Builder tab (merged from the Taqnix App Builder plugin): build your Android APK or App Bundle online
- Secure REST API under /wp-json/taqnix/v1 with token login; drivers only see their own orders

- Complete Flutter source for Android and iOS with the Android release permissions and optional Firebase Gradle plugin

© Taqnix. Licensed under GPL-2.0-or-later.