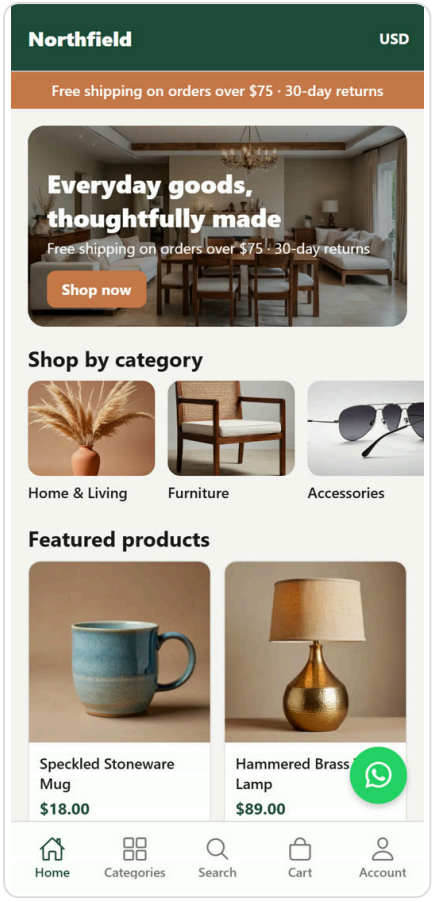
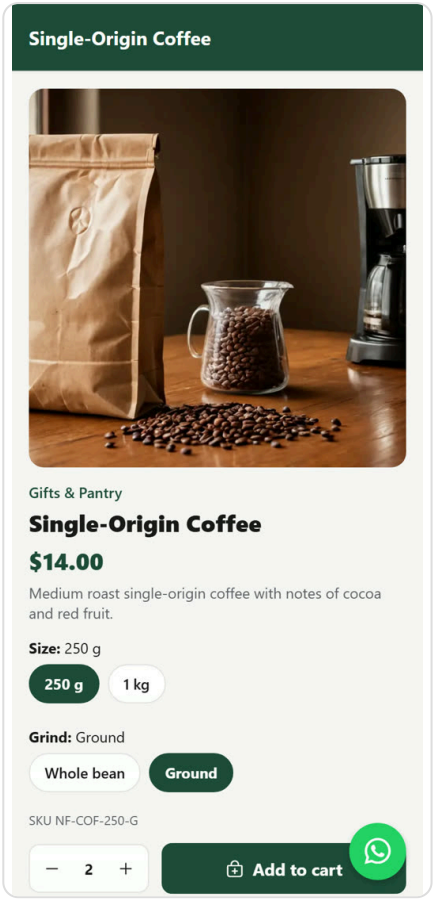
CODE CRAFT · SETUP AND PUBLISHING GUIDE

React Native Shopping App

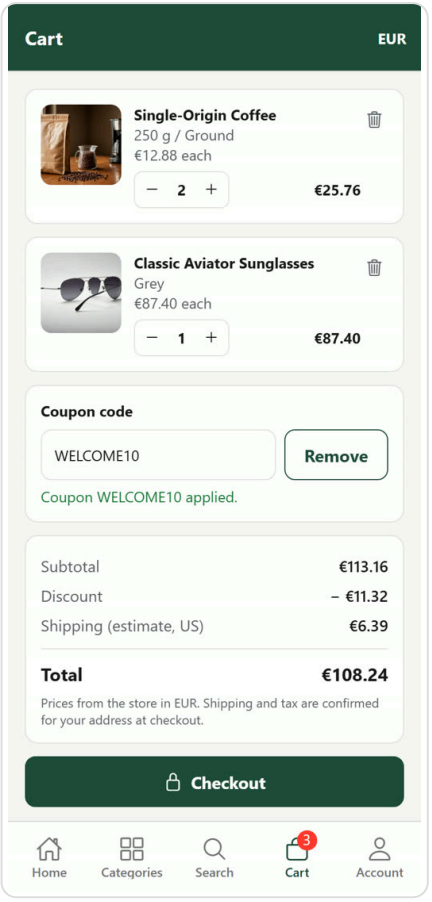
Northfield Shop 1.0: an Expo (SDK 58) + TypeScript app for Android and iOS that runs on your Northfield Node.js store. This guide covers setup, pointing the app at your store, branding, building with EAS and publishing to Google Play and the App Store.



Home: announcement, banner, categories and featured products from your store.



Product: gallery and variant picker. Sold-out options are greyed out.



Cart: every price, the coupon and the total come from the store's quote.

> **Backend required.** The app has no database of its own. It needs the Northfield Node.js store (sold separately) or a server with the same `/api/v1` REST API.

1. What is in the download

File / folder	What it is
`app.config.ts`	App name, scheme (deep links), bundle id / package name and the store URL. Reads `.env`.
`.env.example`	Copy to `.env`: `STORE_URL`, `APP_NAME`, `APP_SCHEME`, `BUNDLE_ID`, `BRAND_COLOR`, `EAS_PROJECT_ID`, `ENABLE_PUSH`.
`eas.json`	EAS build profiles: development, preview (Android .apk) and production.
`src/app/`	The screens (Expo Router): tabs for Home, Categories, Search, Cart, Account; product, category, checkout, order, my orders, sign in, settings, contact.
`src/lib/`	API client, data types, cart / customer / store state, theme, notifications.
`src/components/`	Product card and grid, filters, country picker, WhatsApp and live-chat buttons.
`assets/`	Placeholder icon, Android adaptive icon layers, splash and favicon. Replace with yours.
`README.md`, `LICENSE.txt`	Reference and licence.

Requirements

- Node.js 20 or 22 LTS on your computer.
- Your Northfield store running and reachable from the phone (https for a published app).
- To try it on a phone: the Expo Go app (iOS / Android) or a development build.
- To publish: a free Expo account; a Google Play developer account (one-time fee) and/or an Apple Developer Program membership (yearly fee).

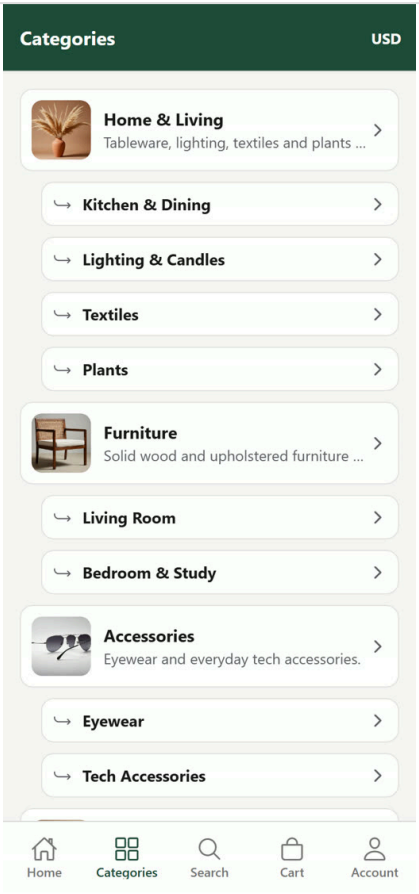
2. Run it on your computer

1. Unzip the download and open a terminal in the folder.
2. Install the packages and create your settings file:

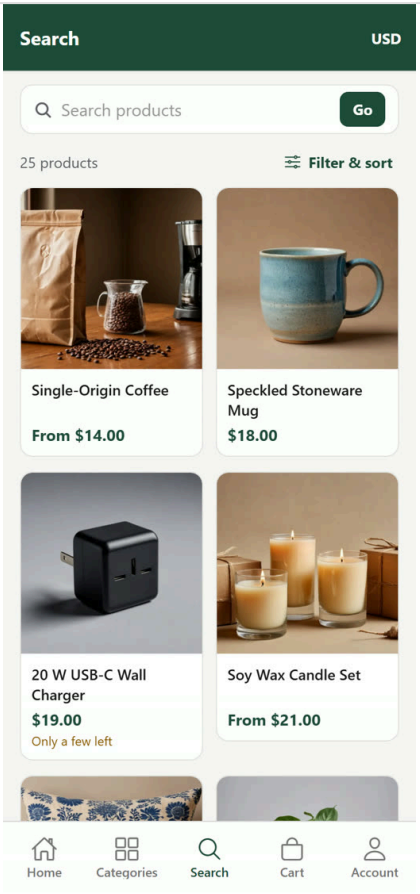
```
npm install
cp .env.example .env     # Windows: copy .env.example .env
```

3. Open `.env` and set `STORE_URL` (next section).
4. Start the app:

```
npx expo start
#  w = open in the browser    a = Android emulator    i = iOS simulator (Mac)
#  or scan the QR code with Expo Go on your phone
```



Categories: the nested category tree from the store.



Search: search box, category and price filters and five sort orders.

3. Point the app at your store

The store address is a single setting, `STORE_URL`, in `.env`:

```
STORE_URL=https://shop.your-domain.com
```

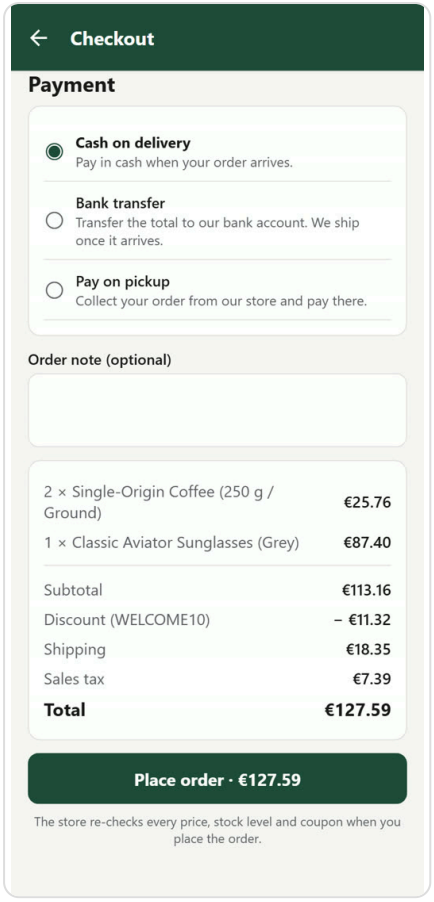
- No trailing slash, and no `/api/v1`: the app adds it.
- The published app must use **https**. Android and iOS block plain http by default.
- Android emulator with the store on your own computer: `http://10.0.2.2:3420`.
- A real phone on the same Wi-Fi: `http://192.168.x.x:3420` (your computer's address).
- Restart `npx expo start` after any change to `.env`.

What the app reads from the store

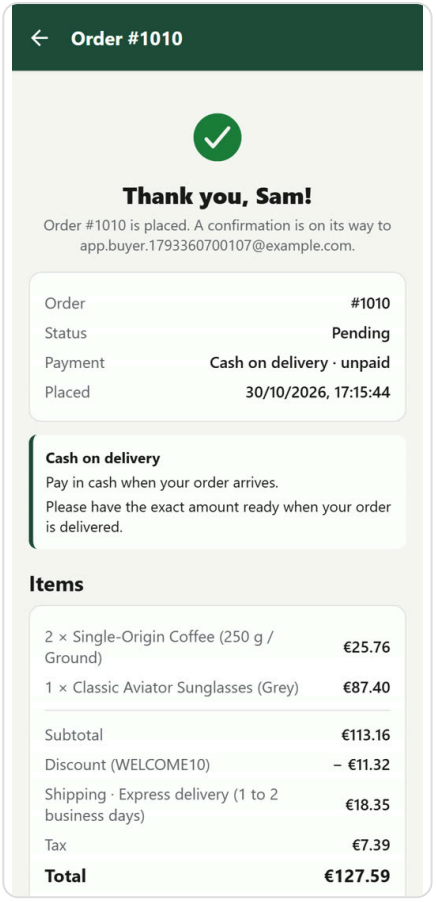
In the store admin	In the app
Settings → business name, logo, colours	Header logo or name, header and button colours (text colour picks itself for contrast)
Settings → announcement bar, tagline	Announcement strip and the home banner
Settings → WhatsApp button, live chat (Tawk.to, Crisp, Tidio)	Floating WhatsApp and chat buttons, shown only when switched on
Currencies (enabled, rates, symbols)	Currency switcher; prices converted by the store
Shipping zones and methods, taxes	Shipping choices and tax at checkout for the customer's address
Payments (enabled and order)	Payment methods at checkout
Coupons	Coupon field in the cart (expired, minimum spend and usage limits checked by the store)
Licence	Until the store licence is active on its domain, checkout in the app is switched off

> **The app never calculates a price.** The cart keeps only product, variant and quantity. Every line price, discount, shipping cost, tax and total comes from the store's `/cart/quote`, and the store re-prices again when the order is placed.

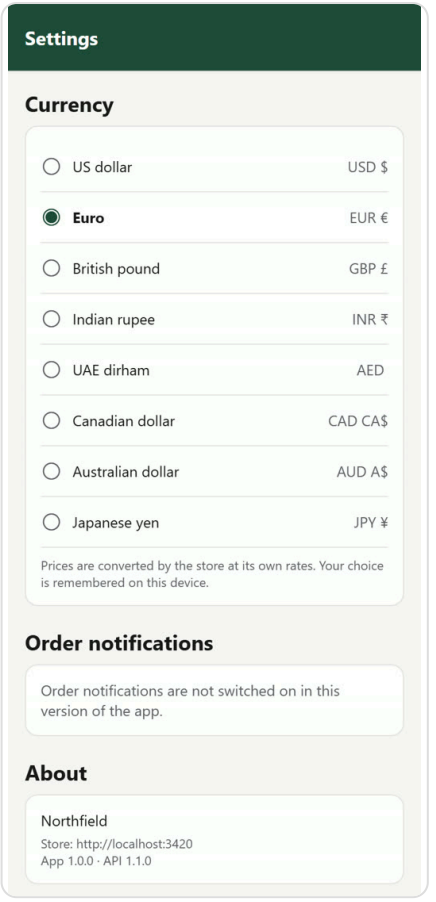
4. Shopping and checkout



Checkout: shipping methods for the address, payment methods from the store, order summary.



Order confirmation with payment instructions and the totals the store recorded.



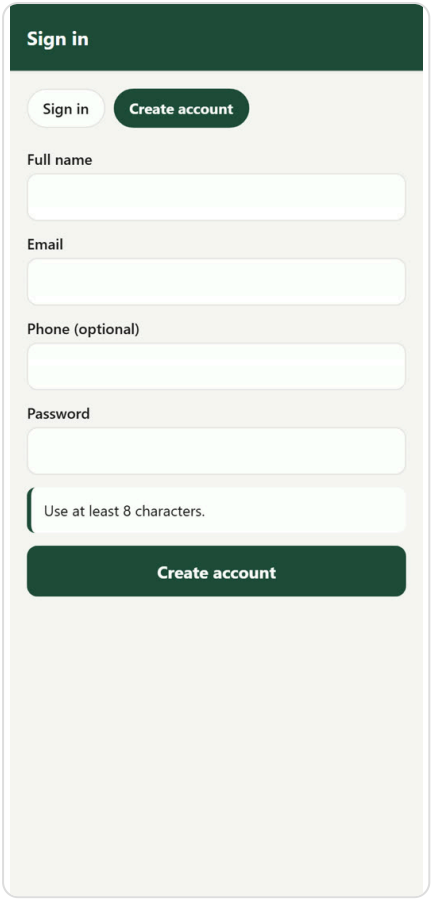
Settings: currency switcher. The choice is remembered on the device.

Payments

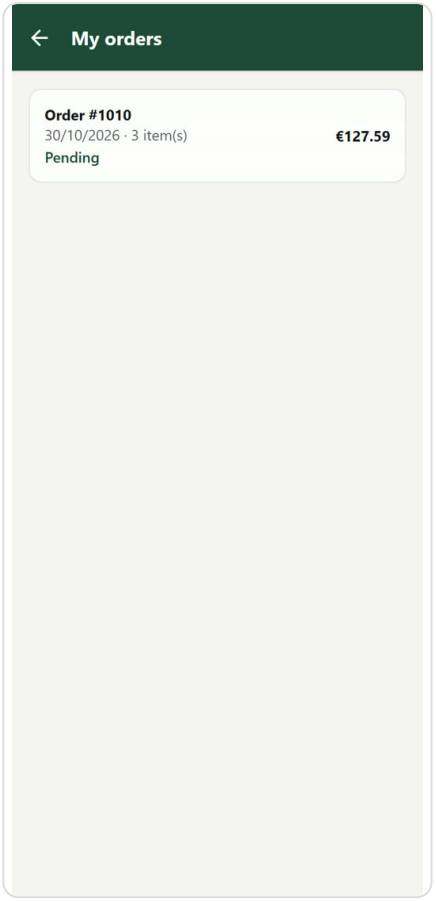
- **Cash on delivery, bank transfer, pay on pickup** (and UPI for INR) are handled in the app: after the order it shows the store's instructions and bank details.
- **Card gateways** come from the store's add-ons (Stripe, Razorpay, PayPal; sold separately). The app opens the store's secure payment page in an in-app browser and, when the customer comes back, re-reads the order to see whether it is paid. If the page redirects to `<APP_SCHEME>://order/<number>`, the browser closes by itself. Card details are never typed into the app.

> The gateway hand-off follows the store's API and was not tried with a real gateway account. Place a test-mode order with your own gateway account before you publish.

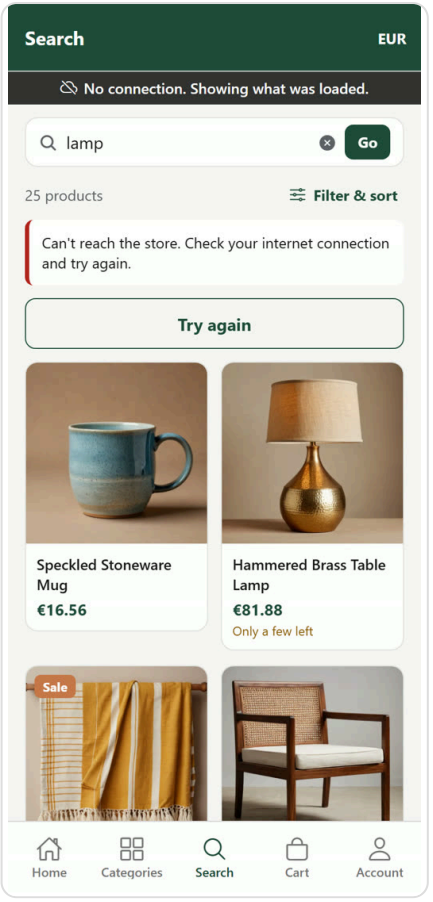
5. Accounts, orders and offline use



Sign in or create an account (tokens from the store, valid for 90 days).



My orders: the signed-in customer's orders with status and total.



No connection: a banner, the last results stay on screen, and a Try again button.

- Guests can check out if the store allows it. Their order opens with its private access key, kept on the device.
- Signing out also revokes the token on the store.
- Contact screen: phone, email, WhatsApp, live chat and a contact form that lands in the store's enquiry inbox.

6. Your branding

1. **Colours and logo:** change them in the store admin (Settings). The app follows on its next start; no rebuild needed.
2. **App name and ids** in `.env`: `APP_NAME` (shown under the icon), `APP_SLUG`, `APP_SCHEME` (deep links, lowercase, no spaces), `BUNDLE_ID` (e.g. `com.yourcompany.shop`; it cannot change after the first store upload).
3. **Icon and splash:** replace the files in `assets/`:

File	Size
`icon.png`	1024 × 1024, no transparency (App Store)
`android-icon-foreground.png`, `android-icon-background.png`, `android-icon-monochrome.png`	Android adaptive icon layers; keep the artwork in the middle 66%
`splash-icon.png`	Shown centred on white while the app starts
`favicon.png`	48 × 48, web build only

4. `BRAND_COLOR` sets the Android adaptive-icon background and notification colour.

Push notifications (optional)

Switched off by default. Set `ENABLE_PUSH=true` and `EAS_PROJECT_ID`, then Settings shows "Turn on notifications": the app asks permission and gets an Expo push token on a real phone (development or store build). The app also shows a local notice when it sees an order status change on refresh. The store does not send push messages by itself: to notify customers while the app is closed, save the token on your server and call the Expo Push API from a store add-on on the `order.statusChanged` hook (example in README.md).

7. Build with EAS

> **Please note:** we have not run native Android or iOS builds of this app ourselves. We tested it through its web build against a running store (section 9). The steps below follow Expo's EAS documentation.

```
npm install -g eas-cli            # or: npx eas-cli@latest <command>
eas login                          # your free Expo account
eas init                           # creates the project; put the id in EAS_PROJECT_ID

# set STORE_URL for each profile in eas.json ("build" > "production" > "env")

eas build -p android --profile preview      # .apk to install on a phone
eas build -p android --profile production   # .aab for Google Play
eas build -p ios --profile production       # needs an Apple Developer account
```

- Builds run on Expo's servers; you do not need Android Studio or a Mac for them.
- `.env` is not uploaded to the build servers, so put `STORE_URL` in `eas.json` or in EAS environment variables.
- Raise `version` in `app.config.ts` (and `android.versionCode` / `ios.buildNumber`) for every new upload.
- Later JavaScript-only changes can be shipped with `eas update` (optional; see Expo's EAS Update docs).

Submit

```
eas submit -p android     # needs a Google Play service-account JSON key
eas submit -p ios         # uploads to App Store Connect / TestFlight
```

8. Store submission checklist

☐ `STORE_URL` is your live **https** store; the store licence is active and a test order works.

☐ App name, bundle id / package name, scheme, icon, adaptive icon and splash replaced with yours.

☐ Version and build numbers raised for this upload.

☐ Privacy policy URL (the store has `/page/privacy-policy`; edit it for your business) and a support email or URL.

☐ Google "Data safety" and Apple "App Privacy": the app sends name, email, phone and delivery address to your store to fulfil orders. It contains no analytics or advertising SDKs.

☐ Account deletion: both stores require a way for customers to ask for their account to be deleted. Add a deletion request page or contact link and mention it in the listing.

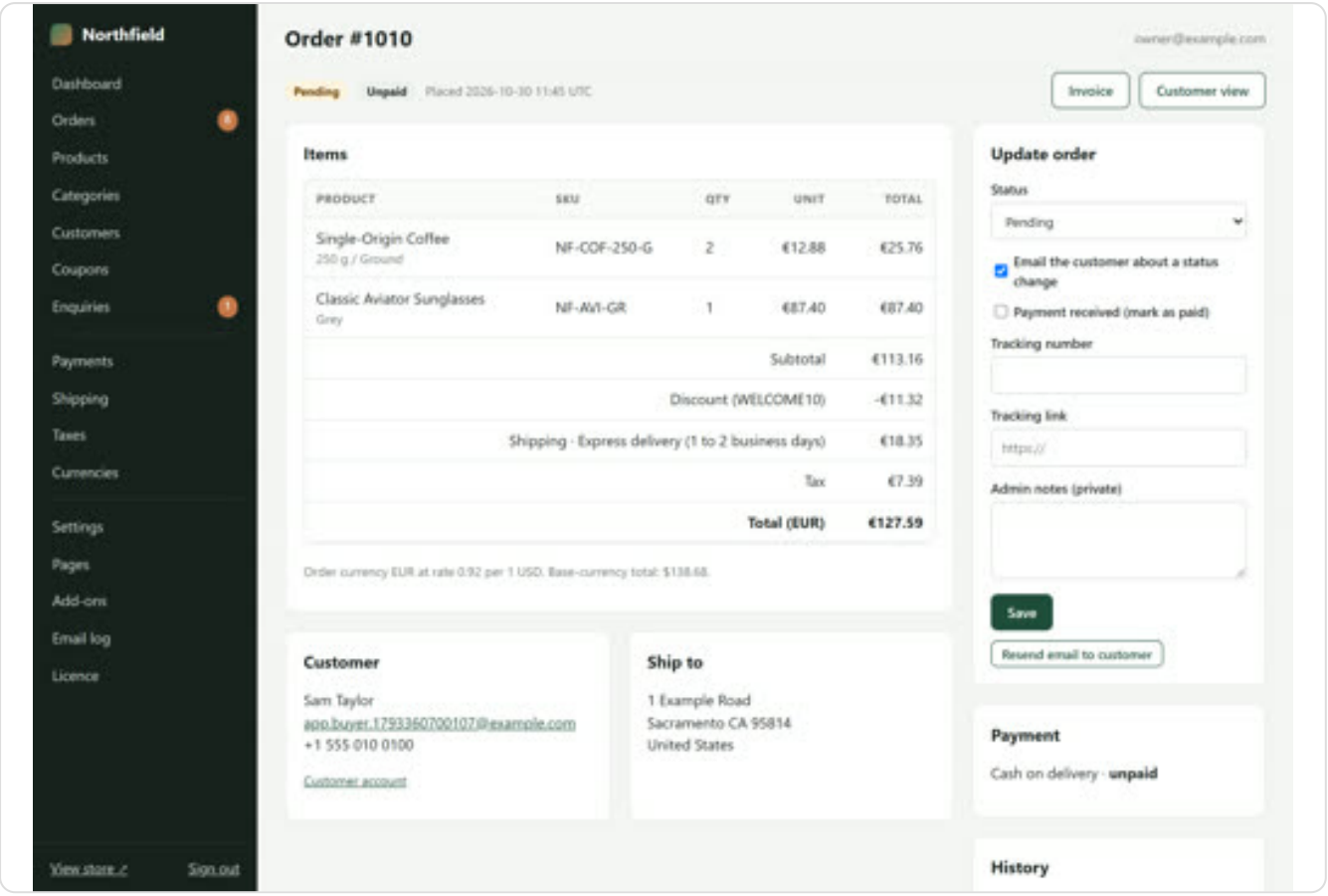
☐ Screenshots for the required device sizes, short and full descriptions, category Shopping, content rating questionnaire.

☐ Apple review notes: a test customer login, and a note that the app sells physical goods (Apple in-app purchase does not apply).

☐ Test on a real phone with every payment method you switched on, including a test-mode card payment if you use a gateway add-on.

9. What we tested

- `npx tsc --noEmit`: no errors. `npx expo-doctor`: 20 of 20 checks passed.
- The app's web build (`npx expo export -p web`) against the Northfield Node.js store, driven by Playwright at a 390 × 844 phone-sized window. 33 of 33 checks passed: home, banner, categories, WhatsApp button shown because it is on in the admin, search with sort and a price filter, variant picker (sold-out option greyed out, variant price equal to the API), currency switch to EUR remembered after reload, cart, expired coupon refused and WELCOME10 accepted, cart total equal to the store's quote, checkout signed in with a US address, Express shipping and cash on delivery, order total equal to the checkout total, the order in the store admin with the same total, sign out and in again, order history and detail, offline banner and retry.



The app's test order in the Northfield store admin, with the same EUR total.

> **Not run by us:** native Android and iOS builds, Expo Go on a phone, push notifications on a device, and an online card payment through a gateway add-on.

Support

Questions about setup: email us from the address in your order email. Code Craft · coodecraft.com