**Summary:**

I started by setting up the project using Angular 20 and standalone components. I kept the root AppComponent minimal with just the router outlet, while all feature-level code was placed under *features/dashboard/pages*. By default, the app opens on the Dashboard page, and I also included a simple 404 page to handle invalid URLs.

Inside the *features/dashboard/components* folder, I created individual components for the Header, Sub-Header, Accounts Panel, Quick Pay & Transfers Panel, and the Footer.

Header & Sub-Header:

At the top, I built a header bar with navigation links like Dashboard, Payments & Transfers, Suppliers, and Help Centre. On the right side, I added the company name “N and F solutions” along with a gear icon to represent settings.  
Just below the header, I created a sub-header that shows key information such as Pending Approvals, Attention Required, Scheduled Payments (CAD and USD), the Next Scheduled Payment Date, and Completed Payments for the month. I used **Angular Flex-Layout** to arrange these items in rows.

Accounts Panel:

I added an Accounts section that displays a list of accounts in a card style. Each account shows the type of bank or wallet, a masked account number, the balance, and the last updated time. Icons are shown for wallets, debit accounts, and inactive accounts. The data for these accounts comes from a mock array inside the component.

Quick Pay & Transfers Panel:

On the right side of the dashboard, I built a Quick Pay panel using **Angular Material form controls**. It includes fields for supplier, amount, payment method, and remittance notes. I also added Clear and Pay Now buttons. There’s also a placeholder for a Transfers tab that can be built out in the future.

Payment Orders Section:

Below the Quick Pay area, I created a Payment Orders section in the dashboard page only for simplicity. At the top, it has tabs for In Progress, Scheduled, Pending Approval, and Draft orders, with counts for each. Under that, a table lists the orders with details like Order Number, Processing Date, Status, and Amount. The rows use alternating styles for readability. This data also comes from a mock array.

Footer:

At the bottom, I built a footer with contact information, Terms of Service, and Privacy links. It also includes the RBC PayEdge branding and support details.

**How I Built It:**

I built this project using Angular 20 with standalone components as the foundation. To speed up the UI work and stay close to the mockups, I relied on Angular Material for elements like buttons, form fields, tabs, and icons. For the layout, I made use of Angular Flex-Layout, which gave me flexibility in arranging sections and keeping the design responsive. Created some common reusable classes with prefix as acl (ANGULAR component library), so that we can reuse them. All custom styling was handled with SCSS.

The application is organized into small, focused components. At the top level there is the AppComponent which serves as the shell. Inside it, the Dashboard page brings together the main building blocks: the Header, SubHeader, Accounts section, Quick Pay form, Orders Table, and the Footer. Each piece is separated out as its own component so that the structure remains clean and easy to maintain.

For data, I kept things simple by using TypeScript arrays to define both accounts and payment orders. These arrays supply the information that is displayed on the screen. Wherever formatting was needed, such as for dates and currency values, I used Angular’s built-in DatePipe and CurrencyPipe.

On the styling side, I combined Angular Material’s theme with my own SCSS classes. Flexbox was used to align items and manage spacing and also created some common reusable classes like acl-margin-left for page formatting. I also created small utility styles to handle things like font sizes, muted text, padding, and alternating row striping in the tables.

One of the main trade-offs I made was to keep the data in memory rather than connect the application to any APIs. This allowed me to focus on reproducing the layout quickly. Similarly, the tabs in the Payment Orders section are visual only and don’t yet filter the data.