Jaipur Engineers

Your One-Stop IT & Staffing Ecosystem

FULL-STACK WEB & MOBILE DEVELOPMENT

MERN Stack + React Native

Build a React web dashboard and an Android or iOS app backed by the same API.

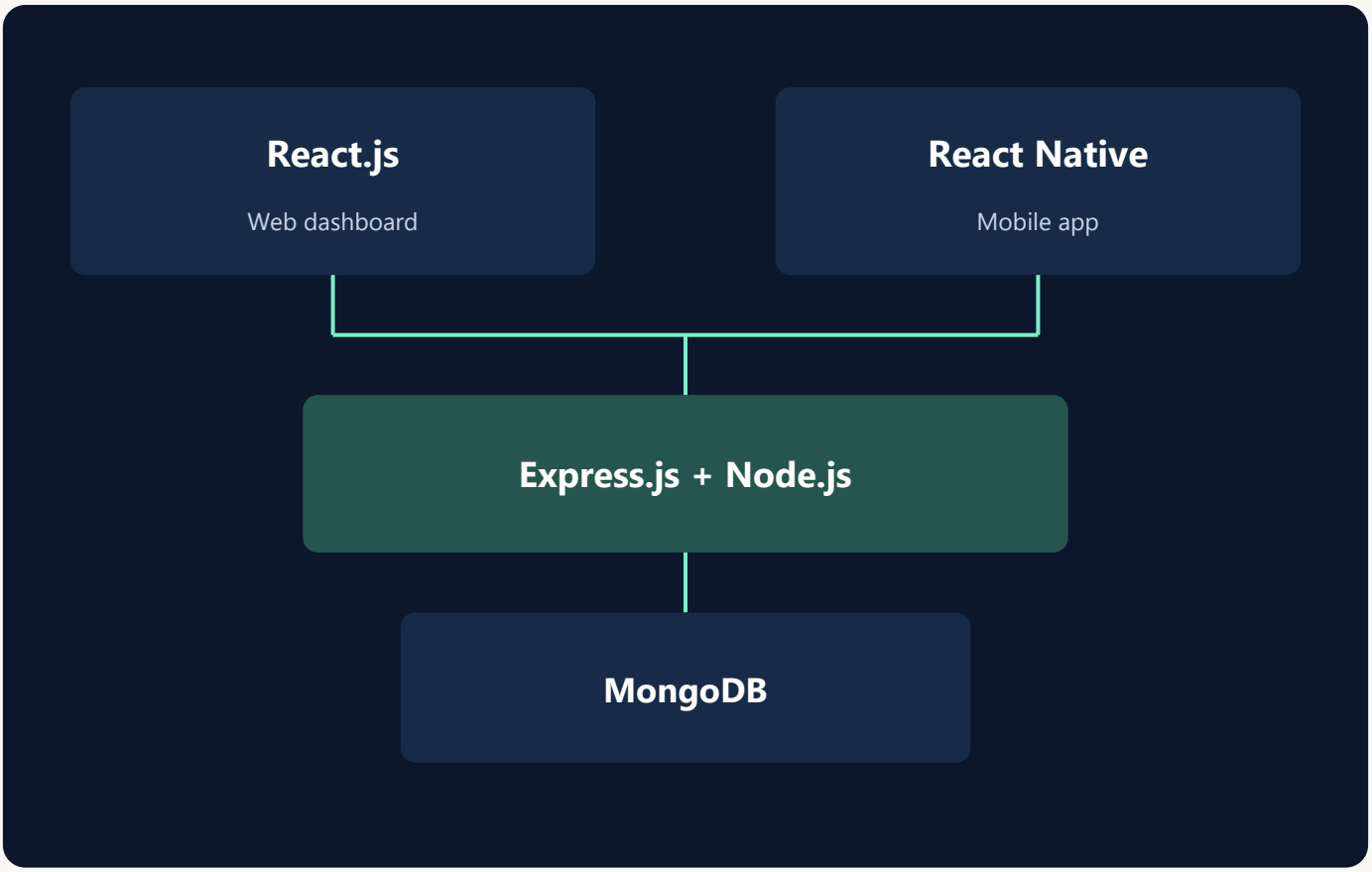


MANSAROVAR, JAIPUR

jaipurengineers.com

Two apps, one backend

See how the React dashboard and React Native app connect to the same Express API and MongoDB database.



Web

Build the browser interface.

Server

Validate requests and persist data.

Mobile

Bring the same records to a phone.

WHO IT'S FOR

Students and developers who want to build both web and mobile apps. Basic coding familiarity helps.

Work with JavaScript

Use modern JavaScript to prepare data for the web interface, the API and the mobile app.

booking-summary.js

```
01  const confirmed = bookings
02    .filter(({ status }) => status === 'confirmed')
03    .map(({ service, time }) => ({ service, time }));
04
05  console.log(confirmed);
```

Language

ES6+ syntax, arrow functions, destructuring, modules and the spread operator.

Data

Arrays, objects, JSON, array methods, promises and async/await.

Workflow

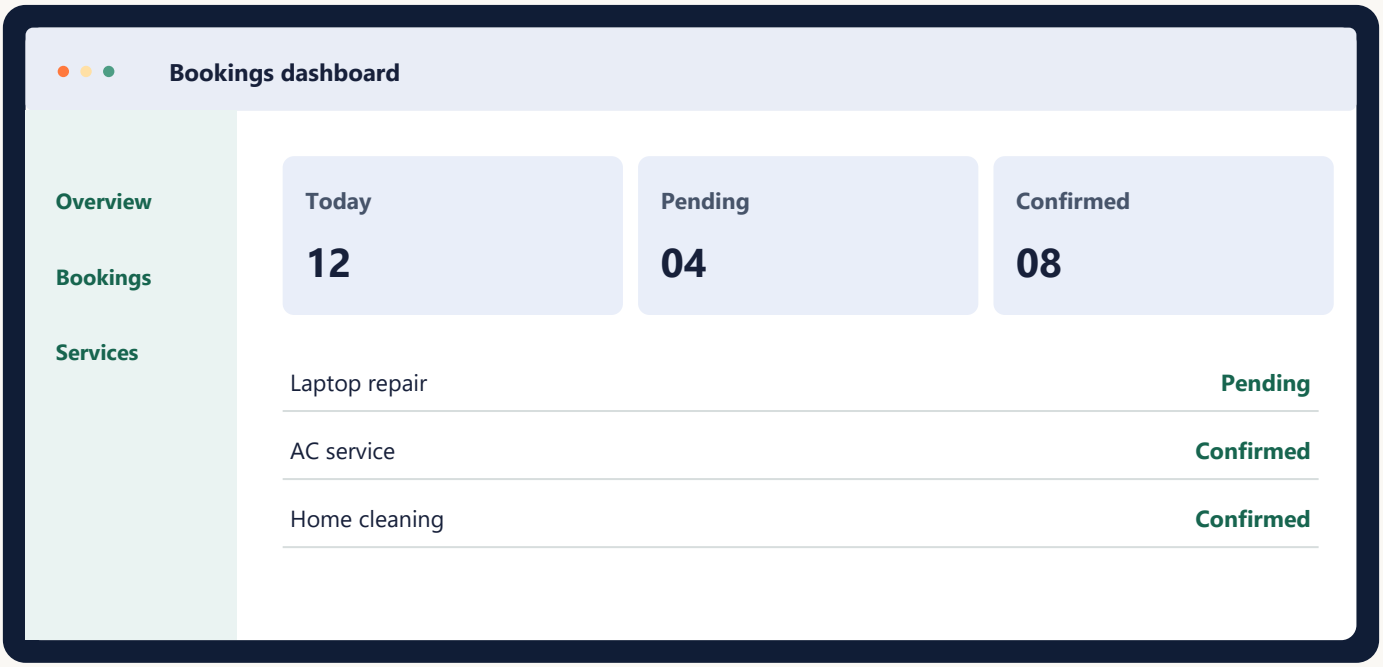
Set up Node.js and npm. Track changes with Git and document work in GitHub.

PRACTICE

Filter a list of bookings, format the results and save the exercise in a Git repository.

Build the web dashboard

Create a browser interface with reusable components, clear routing and responsive layouts.



ILLUSTRATIVE PROJECT SCREEN

Components

Build dashboard cards, tables and reusable form controls with JSX and props.

Routing

Use React Router to connect dashboards, lists and detail views.

Layout

Apply responsive CSS and review styling workflows with Tailwind CSS or Bootstrap.

PRACTICE

Build a booking list that adapts to a small screen and links each record to a details page.

Manage app state

Handle changes from forms, filters and API requests so users see the right information at each step.

01

Local state and effects

Keep input values and filters in the component. Use React hooks to handle state and coordinate side effects.

02

Shared state with Redux Toolkit

Create slices and selectors for features that multiple screens need. Track async request status alongside the results.

03

Forms and feedback

Validate inputs, explain field errors and provide clear loading, empty and retry states.

PRACTICE

Filter bookings by status and display a clear message when the filter has no matches.

Build a REST API

Create a service that responds to web and mobile requests with consistent JSON and HTTP status codes.

METHOD	ROUTE	PURPOSE
GET	/services	Browse services
GET	/bookings	List visible bookings
POST	/bookings	Create a booking
PATCH	/bookings/:id/status	Update status

Structure the server

Use routes, middleware and handlers. Validate input, map errors to HTTP status codes and check requests with Postman.

PRACTICE

Send a valid booking request, then try a missing field and confirm the API returns a usable error.

Model real application data

Define the records each client needs and make common queries easy to express.

Services

```
_id
name
durationMinutes
active
```

Bookings

```
_id
serviceId
userId
status / startTime
```

Users

```
_id
name
email
role
```

Bookings reference a service and a user.

CRUD

Create, read, update and delete documents from the API.

Validation

Use schema rules and practise with a MongoDB driver or Mongoose.

Queries

Filter and paginate booking records. Learn where indexes can help.

PRACTICE

Retrieve the signed-in customer’s bookings. Sort by upcoming time and send paginated results.

Control who can do what

Protect the booking workflow with sign-in, server-side permissions and careful handling of user data.

01

Sign in and sign out

Review password hashing, login validation and session or token-based authentication.

02

Restrict access by role

Let customers see their own records and staff update booking status. Enforce these rules in the API.

03

Handle credentials carefully

Keep server secrets in environment configuration and store session credentials appropriately for each platform.

PRACTICE

Verify that a customer cannot update another customer's booking and that a staff user can approve a booking.

Build the mobile interface

Create iOS and Android screens for browsing services, choosing a time and checking booking status.

Set up a project

Start with Expo and test on a phone or an Android emulator.

Build screens

Use View, Text, Pressable, TextInput and FlatList. Style layouts with Flexbox.

Connect screens

Create stack and tab routes. Pass a selected service into its details screen.



ILLUSTRATIVE MOBILE SCREENS

PRACTICE

Build the mobile service list, detail screen and booking form. Check them at more than one screen size.

Connect mobile to MERN

Use the same REST API from both clients and handle the device conditions that affect mobile requests.

Share the API contract

Fetch the same service and booking records as the dashboard. Handle validation and session errors in the app.

Work with local data

Save preferences or cached non-sensitive data. Use a retry action for failed network requests.

Test on a device

Check sign-in, slow connections and screen layouts while running the app on real hardware.



MOBILE CLIENT AND BACKEND

PRACTICE

Submit a booking from a phone. Refresh the web dashboard and verify that it shows the same record.

Test, deploy and package

Check one complete booking across the browser, the API and a phone before preparing a release.

01

Test the complete workflow

Check valid and invalid inputs, signed-out states and interactions between the web and mobile clients.

02

Deploy the web app and API

Review build settings, environment variables, database connectivity, HTTPS and service logs.

03

Prepare mobile builds

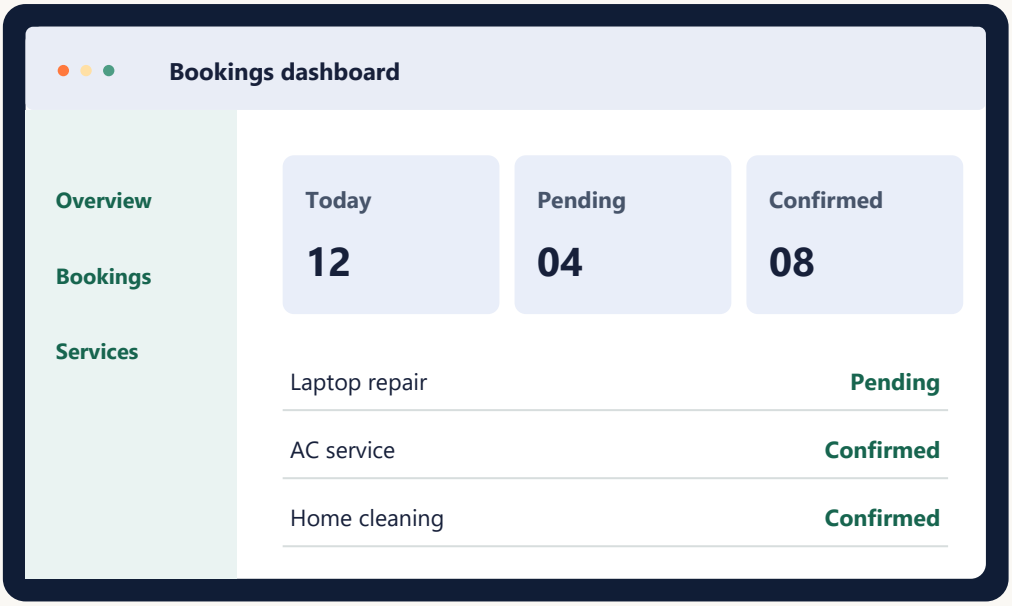
Review preview builds, app signing, Android App Bundles and the Google Play and App Store submission steps.

PRACTICE

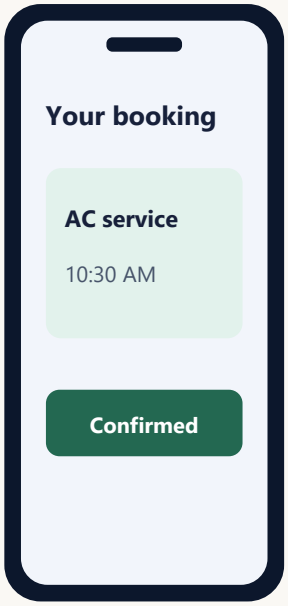
Write a test checklist and prepare a release summary with known issues and demo steps.

Service booking platform

Customers book a service on their phones. Staff review and update those bookings in the web dashboard.



STAFF WEB DASHBOARD



CUSTOMER APP

Customer app

- Browse available services
- Select a date and time
- See booking status

Staff dashboard

- Review pending requests
- Confirm or decline a booking
- Filter by date and status

PROJECT DEMO

Create a booking on the phone, approve it in the browser and show the updated status back on the phone.

Present what you built

Explain the product, how its parts work together and the trade-offs you made along the way.

Document the setup

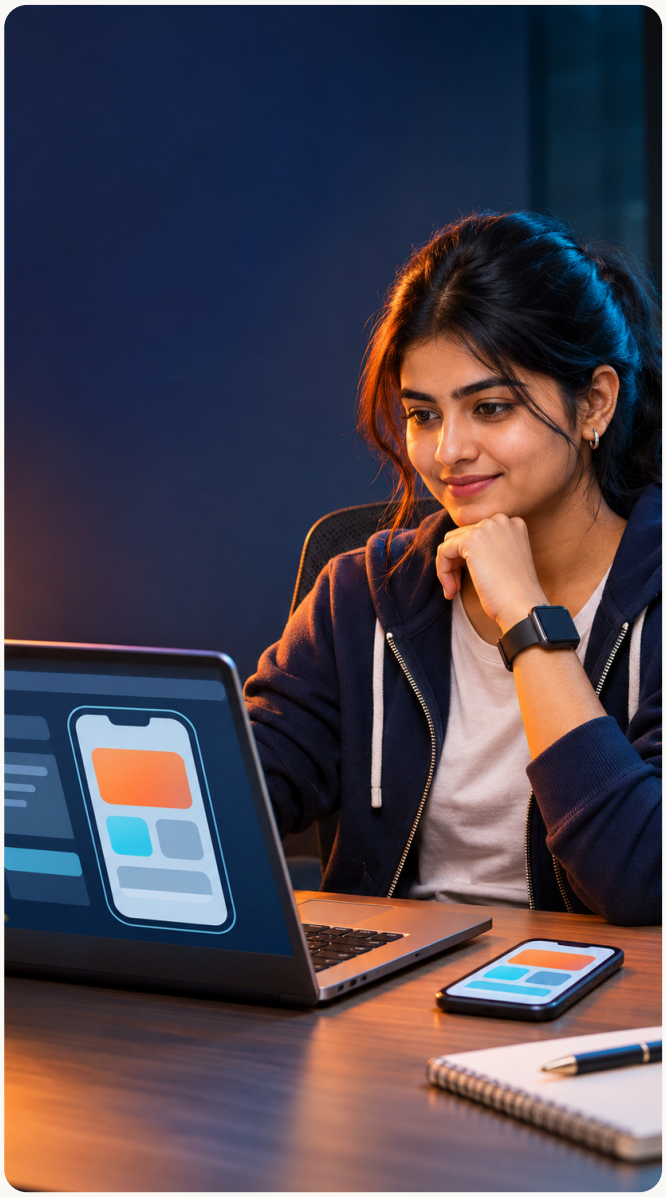
Write a README that covers the web app, API and mobile project. Include instructions and sample data.

Record a short demo

Show a booking going from the mobile app to the dashboard and back.

Practise the explanation

Trace a request from the UI to MongoDB and explain how the app handles a failure.



CAREER DIRECTIONS

Explore junior full-stack, frontend or mobile development roles and prepare to discuss your project in interviews.

Jaipur Engineers

Your One-Stop IT & Staffing Ecosystem

COURSE ENQUIRIES

Ask about the next batch

Speak with Jaipur Engineers about the combined MERN + React Native fee, course duration and available batch timings.

CALL JAIPUR ENGINEERS

+91 70146 92039

EMAIL & WEBSITE

info@jaipurengineers.com

jaipurengineers.com

VISIT THE TRAINING CENTRE

122/228, Indra Path, Thadi Market

Mansarovar, Jaipur, Rajasthan 302020

SCAN FOR MAP

MERN course details

React Native course details