Tech stack findings

Component Libraries

- Chakra
- Ant Design
- MUI
- Semantic UI
- React Bootstrap

Deployment

https://www.invonto.com/insights/mobile-app-development-process/

- Android sdk
- Deploying an app to the android or ios store will take time.
- Android takes upto 7 days.
- AWS - little time to spend on the deployment of a new version of your web or mobile app.
- AWS offers easy deployment process for an app
- Heroku Allows the developer to focus on code instead of infrastructure
- Releasing a native mobile app requires submitting your app to the app stores, Apple App Store for iOS apps and Google Play for Android apps. However, you will need a developer account with Apple App Store and Google Play Store before launching your mobile app.
-

Data

https://www.simform.com/blog/mobile-app-developers-database-selection/

- MongoDB is faster than MySQL when it comes to handling a large volume of unstructured data. On the other hand, MySQL would be faster for structured data.
- Data Modelling is very helpful if your app contains features like search queries, reporting, location-based features, etc. Such apps require multiple databases to manage different types of data. For example, Uber uses MySQL, MongoDB and lot of other databases. They use MongoDB for their CDN and MySQL for business logic. Using MongoDB in their case made it very simple to store high volume incoming data.
- Using an unstructured database like MongoDB would bring in flexibility to change without going through the painful process that you would find yourself in with structured

databases. MongoDB allows you to work with unstructured data in multiple layers, adding new app features with flexibility

- SQL databases are ideal for situations where -
- Data Is transactional
- The structure doesn't alter frequently or at all
- Data integrity is vast
- You need Analytical queries

- NOSQL databases are suitable for situations where -
- Scalability and flexibility are a priority
- Rapid development and repetition is needed
  If you need structure, set schemas, and multi-row transactions, go with SQL. It is ideal for requirements like accounting systems. On the other hand, if your business grows rapidly and you don't require schema definitions, NOSQL will get your project up and running quicker.
  NOSQL has the best flexibility and adaptability skills to do so if you have huge chunks of data that need to be churned through data in different structures.

Backend
https://os-system.com/blog/what-is-backend-and-how-to-choose-the-best-mobile-app-backend-for-your-application/

- Axios seems like the better option compared to a standard Fetch request.
- Axios has the ability to cancel request which gives us more control.

React native:

The main differences between React and (React)Native syntaxes are the use of different tags, for example; text, view, etc

Native has lots of premade components too and the possibility to create your own with tools such as

TouchableNativeFeedback
https://reactnative.dev/docs/touchablenativefeedback
TouchableOpacity https://reactnative.dev/docs/touchableopacity

Ex:

iOS – ActionSheetIOS, AlertIOS, DatePickerIOS, ImagePickerIOS, ProgressViewIOS, PushNotificatoinIOS, SegmentedControlIOS, etc.

Android – DatePickerAndroid, DrawerLayoutAndroid, PermissionsAndroid, ProgressBarAndroid, TimePickerAndroid, ToastAndroid, ToolbarAndroid, ViewPageAndroid, etc.

React Native dev tools and libraries:
Redux http://redux.js.org/ (for handling your app's state)
Awesome React Native http://www.awesome-react-native.com/ (a list of components and demos)
Yoga https://yogalayout.com/ for building layouts
React Developer Tools
https://reactnative.dev/docs/debugging#react-developer-tools
for debugging

Besides this React Native has comprehensive docs https://reactnative.dev/docs/getting-started a great support on GitHub https://github.com/facebook/react-native and many tutorial videos https://www.youtube.com/watch?v=qSRrxpdMpVc

Example apps
https://github.com/ReactNativeNews/React-Native-Apps

Next JS