React Native monorepo

Unless explicitly specified, the work directory in Terminal defaults to `<RN_APP_ROOT>`

(1) Make workspaces
Modify `package.json` by adding entries `"workspaces"`:
```
{
  ...
  "workspaces": [
    "packages/*"
  ],
  ...
}
```
Then VERY IMPORTANTLY, in Terminal run `yarn`.

(2) Make a package for the first JS app
In Terminal,
`mkdir packages && cd packages`
`mkdir app && cd app`
`yarn init -y`
We will get a new created package.json at packages/app/
```
{
  "name": "app2",
  "version": "1.0.0",
  "main": "index.js",
  "license": "MIT"
}
```
Then add “scripts” and “dependencies” to the above package.json file, which should finally look like
```
{
  "name": "app2",
  "version": "1.0.0",
  "main": "index.js",
  "license": "MIT",
  "scripts": {
    "start": "react-native start"
  },
  "dependencies": {
    "react": "18.2.0",
    "react-native": "0.72.4"
  }
}
```

Run `yarn` at packages/app/ (A file node_modules/.bin/react-native would be created locally and I don’t know why but don’t worry about it.)

Create 3 new files:
 - packages/app/index.js
 - packages/app/app.json
 - packages/app/App.tsx
then from <RN_APP_ROOT> find the 3 corresponding files, copy and paste their contents to these 3 new created files.

Create a new file `packages/app/metro.config.js` and paste the following contents into it:

```
const {getDefaultConfig, mergeConfig} = require('@react-native/metro-config');

/**
 * Metro configuration
 * https://facebook.github.io/metro/docs/configuration
 *
 * @type {import('metro-config').MetroConfig}
 */
const config = {};

const mergedConfig = mergeConfig(getDefaultConfig(__dirname), config);

const path = require('path');

module.exports = {
  ...mergedConfig,
  watchFolders: [
    path.resolve(__dirname, '../../node_modules'),
    // path.resolve(__dirname, '../shared-components'),
  ],
};
```

Now we can delete the 4 files from <RN_APP_ROOT>:
 - <RN_APP_ROOT>/index.js
 - <RN_APP_ROOT>/app.json
 - <RN_APP_ROOT>/App.tsx
 - <RN_APP_ROOT>/metro.config.js

To verify the first JS app package,
In Terminal,
`cd <RN_APP_ROOT>/packages/app`
`yarn start`
then run the iOS app.

If everything goes well, repeat the above steps to create the package for the second JS app at `packages/app2/`.

One important step for app2,
packages/app2/app.json
```
{
  "name": "app2",
  "displayName": "app2"
}
```
ios/app2/AppDelegate.mm
```
  self.moduleName = @"app2";
```