## Context\nNPM is causing confusion as to why lock files are changing in local\nenvironments when no changes have been made. We have found explanations and\nworkarounds, but it feels like the type of unexpected default behavior that will\nlead to frustration as new developers join the project.\nYarn is an alternative package manager that seems to have a more expected set\nof default behaviors while maintaining compatibility in case we need to revert.\n

We will use Yarn instead of NPM for this project.\n

## Context\nYarn and NPM can both manage the Node packages for a project. Recent updates to NPM mean that Yarn only has a negligible performance advantage over NPM.\n

We will use `yarn`, `yarn start`, `yarn add`, `yarn remove` etc. for the management of Node packages in our project.\n

## Context\nWe're getting security vulnerabilty warnings from GitHub due to transitive dependencies. Npm offers a `--depth` setting for updating dependencies that yarn doesn't seem to have. Which raises the question: why use yarn?\n

Switch to npm.\n

## Context\nWe need to make a decision on the testing framework for our project.\n

We will make use of pytest. It is a de facto standard in the Python community\nand has unrivaled power.\n

## Context\nWe want a test framework that has good support for TypeScript and Node. Jest is\na fast testing framework with good resources for mocking.\n

We will use Jest as our testing framework.\n

## Context\nWe want a test framework that has good support for React and TypeScript.\n[Jest](https://jestjs.io) is the standard, recommended test framework for React\napps.\n

We will use Jest as our testing framework.\n