Brandon Laubaugh

Tools for continuous integration consist of

- Bamboo - https://www.atlassian.com/software/bamboo

Bamboo is able to access any language, and other popular technologies. Very adaptable and has a big variety for both build and deployment projects.Bamboo has a dedicated agents feature, you can run hot fixes and critical builds right away, When bamboo releases its dedicated agent no other activity will be able to use it unless dedicated to that activity. Bamboo can trigger builds based on changes detected in the repo, it has Branch detection, detects new branches in git mercurial etc and applies the main lines CI scheme to them automatically. Allows you to set up CI builds as normal and feed the artifacts into deployment projects. Bamboo supports everything from continuous integration to deployment to delivery.

Bamboo is a great tool with many many features and honestly ill probably wind up using it as it seems it can do a lot. It is a paid subscription but luckily there is a free version that is limited.

The company started in 2002 by two developers straight out of college and they have many other applications, bamboo is just their number one it seems

They power
NASA
splunk>
ROAMES
Ames research center

Tool for real time error monitoring.

- Sentry - https://sentry.io/welcome/

Sentrys application monitoring platform helps every developer diagnose, fix, and optimize the performance of their code. Source code, error filters, stack locals, sentry enhances application performance monitoring with stack traces. Leaves breadcrumbs to make application development a little easier by showing you the trails of events that lead to errors, sentry essentially gives you fool control of the monitoring and allows you to really get a grasp on what's going wrong with the code by stating they pinpoint a trail of errors to lead you to the right conclusion.

Sentry even has proof that never(rarely) RUNS into errors, using this link for proof.
https://status.sentry.io/

Companies/support with sentry
-android
-apple
-django
-github
-python
-slack
ETC and i mean ETC there are alot more.

Part.2

Documenting the node runtime.js (runtime analysis)

Upon typing in node runtime.js, its followed with the results for the extraLargeArray. Taking around 660.3279 ms and appending at 2.6696 ms.
Taking a very long time being the array is a large number it seems as its written out.

```
const extraLargeArray = getSizedArray(100000);
```

Compared to the others being they are smaller numbers will have a faster and shorter run time. Double append and double insert would still make the run time for the extraLargeArray longer. Doubling the amount of time it wouldbe originally taken.

When placing the time in the new arrays the insert and append get shorter

extraLargeArray(1) = 20.9 us and append 65.2 us

Making the time wayyyyy shorter. The higher the array number the longer the run time.

When making it smallArray = getSizedArray(100) the run time is 43.5/insert and 87.8 append.