Step 1:

1. Jenkins (https://www.jenkins.io/) for continuous integration and Sentry (https://sentry.io/welcome/) for real time error monitoring
2. Unique value for Jenkins:
    a. Jenkins provides hundreds of plugins to support building, deploying and automating any project. I really like that Jenkins is a community driven project that helps with many types of contributions.
    b. Jenkins can be used as a simple CI server or turned into the continuous delivery hub for any project.
    c. It's suppose to be easy installation and configuration. It's open source and user-friendly...does not require additional installations or components. Also, offers easy support!
    d. Free of cost! Developers and DevOps teams don't want to worry about procurement costs for code pipelines.
    e. Has over thousands of plugins to ease your work.

    Unique value for Sentry:
    a. Some major points we talked about in class: Sentry is all about context and being able to trace things easily and in detail. Tracks multiple projects and errors. Gives lots of info about logs about what they appear and occur.
    b. Supports wide array of languages and platforms.
    c. Sentry provides real time updates, where developers can fix code-level issues anywhere in the stack before users encounter errors. Makes resolution easy.
    d. Complete context where spending time where it actually matters instead of in low impact issues.
    e. You can integrate everywhere
    f. You get email notifications, it's open source.
    g. Sentry dashboard is very well laid out, organized, clean looking.
    h. Security and privacy settings is more enhances, ensures, identifiable info and other sensitive info are scrubbed from logs.
3. Getting started with Jenkins is easy:
    a. There is a lot of instructions with Jenkins and an easy to read tutorial. To get started with Jenkins Pipeline go through the tutorial/tour. This tour uses the "standalone" Jenkins distribution, which runs locally on your own machine. You do have to download Jenkins to your computer, but reviews alone are worth the download because of the ease of use. The process is very easy, the instructions are straightforward and easy to follow.
    b. On the home page of Jenkins simply click on the big button in the middle of the page that either says documentation or download. The process is step by step with instructions how to get started. Press either button to get started. It's mentioned multiple times on the site that you won't get lost. Either page will tell you how to download for your computer and even has a guided tour to help you get up and running with Jenkins.
    c. https://www.jenkins.io/doc/pipeline/tour/getting-started/, https://www.jenkins.io/download/

Sentry:

a. Sentry is extremely easy to get started with and has a big get started button at the top of the home page. You simply sign up for free online and can either connect with your google, github, or azure account.
b. On the get started page, you can also request a demo if you want to do that before signing up. Its extremely straightforward and easy to use for the instructions.
c. There is a big tab at the top of the page of documentation to help you with any questions you have, best practices, workshops, project guides and so on. Very easy and user friendly and well documented instructions on how to get started.
d. https://docs.sentry.io/, https://sentry.io/signup/, https://sentry.io/demo/

4. Jenkins was started in 2004, very popular, one of the oldest players in the industry and commands a market share of 71%. They have over a million users, and the community support is really great with Jenkins. Jenkins lists cloudbees, open source lab, cd foundation, github, jfrog, red hate, and aws for their support in jenkins. They also have github repos with commits from 3 days ago as the earliest.

Sentry was founded in 2012. They are extremely popular and have impressive list of customers and clients. They are considered the market leader in application monitoring today. Sentry's growing customer base includes many global brands, such as Disney, Peloton, Cloudflare, Eventbrite, Slack, Supercell, and Rockstar Games. There are commits on github for sentry about 20 min ago.

Step 2:

Timing result for extraLargeArray doublerAppend: 27.75197 ms
doublerInsert: 2.054204964 s

The doublerInsert takes much more runtime to complete because it is doing so much more work and has many more steps to complete the runtime of this function mainly because of .unshift has to make it take so many more steps, because it is shifting the number to the beginning of the array and has to push all of the numbers one step afterwards to push all of the numbers to the beginning of the array.

Table:

```
const tinyArray = getSizedArray(10);
const smallArray = getSizedArray(100);
const mediumArray = getSizedArray(1000);
const largeArray = getSizedArray(10000);
const extraLargeArray = getSizedArray(100000);
```

	Append	Insert
tinyArray	162.881 µs	114.981 µs
smallArray	222.141 µs	126.292 µs
mediumArray	255.558 µs	322.482 µs
largeArray	1.219863 ms	15.675271 ms
extraLargeArray	27.75197 ms	2.054204964 s

It’s very interesting to compare all of these different arrays and see the runtime of each and see how they scale. I definitely can see how the runtime of each of the functions affect how the computer reacts to each and all of the different steps it takes for each one to run. You can definitely see the effectiveness of how you put a function together can affect the runtime. .unshift makes the computer have to do a lot more work and take so many more steps. You can see with the runtime of the doublerInsert function in the insert column how the runtime is significantly longer and slower because of the .unshift in the function. It can takes sometimes twice as long as the append function where.push is in the function. It is cool to see how making your function more efficient with less steps can make a significant impact on the run time of the function. It’s also a lot easier to read the runtime the larger the array. With largeArray and extraLargeArray, it’s simple to see that the larger you scale over time, the more significant you can see the difference of the runtime of the functions over time and how it gets longer and longer and slower and slower with the insert function. The append function is simply better!