Activity 7: Future Cloud Computing and Cloud Research

Ryan Coon

CST-323

Professor Dan Neal

April 30, 2023

**Features**

GitHub CodeSpaces is a new service offering from GitHub. There is no charge for use on personal repositories. It runs a copy of VSCode on a Docker container service, with some additives like port forwarding to fill out its feature set as a fully usable IDE. VSCode extensions can be installed just like on the desktop version. These CodeSpaces are repo-specific and shutdown after 30 minutes of inactivity.

**Workflow Use Cases**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case** | **Supported (Y/N)** | **Observations on Usability and Productivity** | **Other Notes** |
| Opening local folders to grab snippets of code | N | The editor is scoped to just the repository contents. |  |
| Editing editor settings | Y |  |  |
| Installing external extensions | Y | Same extensions are allowed as in desktop VSCode. |  |
| Saving code to version control | Y | Using Git cli tools or the GUI editor tab are options. |  |
| Running development processes/servers | Y | CodeSpaces handles port forwarding automatically for easy testing. |  |
| Use keyboard shortcuts | Y | Supports many of the exact same shortcuts as VSCode. |  |
| Install and manage app dependencies | Y | Full use of a bash terminal on 40GB of space. |  |
| Connect to Local Database instances | N | Port forwarding seems limited for this purpose. | For testing, would need externally hosted database. |
| Manage different versions of lanaguages/tools/CLIs | Y | In the case of the test app, CodeSpaces supports NVM (Node Version Manager) out of the box |  |
| Run Builds and Initiate Deployments | Y | Through use of VSCode extensions and the terminal, these are both possible. |  |

Cloud IDE Analysis

Read the assigned textbook required readings for this topic. Answer the following questions:

What is a feature flag or feature toggle?

A feature flag is a method used by developers and DevOps administrators to set which features should or should not be included in an application build or deployment. Using feature flags allows developers to push out new features to a specific group of users or none at all, should the need arise.

ii. What is A/B testing?

A/B testing is a process of sorting out the best design option for a new feature in an application by presenting (traditionally) two options to end users or testers for evaluation. The option which receives the most positive reception is then pushed forward or iterated upon.

iii. What is continuous delivery?

Continuous Delivery is the process by which developers can automate their deployment pipeline to transform their source code into a deployable application or service set.

iv. What is continuous integration?

Continuous Integration is the process by which developers can push changes to their version control system on a project and see how those changes interact with testing suites, quality assurance automation, and build pipelines.

b. One trend in cloud computing is the ability to develop code in a cloud- and browser-based IDE. Research two viable existing cloud- and browser-based IDEs on the market. How might these cloud-based IDEs conceptually be used to lower the cost of developing code for a company? What are some disadvantages or features that are missing in the IDEs you researched when comparing them to a desktop-based IDE such as Eclipse or Visual Studio?

Theia-ide: Theia is a browser-and-desktop compatible, extendable code editor that was built to be vendor-neutral and open-source. It contains many of the design features of VSCode, going so far as to support VSCode extensions. However, it's far more customizable for a company's individual needs. This editor would be a strong option for larger development firms where they have the resources to define their own editor needs and set up a standard cloud or desktop executable. The costs associated with Theia would be heavily dependent on a company's needs and how they host the system.

A clear downside to these IDE options is that they require internet access to work constantly. If the teams could ensure they have strong and reliable connections to the Internet, then there's little reason they couldn't replace their local development environments with one of these options (given their projects suit the medium). More intensive projects like game development likely won't move to cloud-based IDE's anytime soon.

c. Complete a Scrum retrospective. Document the following:

For one I feel that it was unnecessary to have a CLC for this class. Each CLC for the weeks basically were the same as the activities, but with partners. As far as the curriculum is concerned, I think it is a little outdated. I understand that it cannot be changed each and every class, but every year or two could suffice. A lot of the services have vastly improved, and the deployment methods have changed a little as well. The guides were nice and somewhat accurate to how cloud services work.

I feel that we jumped too much. We spent more on making an application and getting it deployed to a cloud service(which most of us have done in a previous class). I wish we spent a little more time with Docker, Kubernetes and DevOps. I have looked through a lot of IT related jobs and a lot of them focus on DevOps knowledge, which would have been great to dive a little deeper into. Other than that I feel that I learned how to deploy a web application onto the cloud with success.

References:

Start coding instantly with Codespaces. (n.d.). GitHub. https://github.com/features/codespaces

Kavis, M. J. (2014). Architecting the cloud: Design decisions for cloud computing service models (SaaS, PaaS, and IaaS). Wiley.