THINKIFIC

Thinkific DevOps Take Home Test

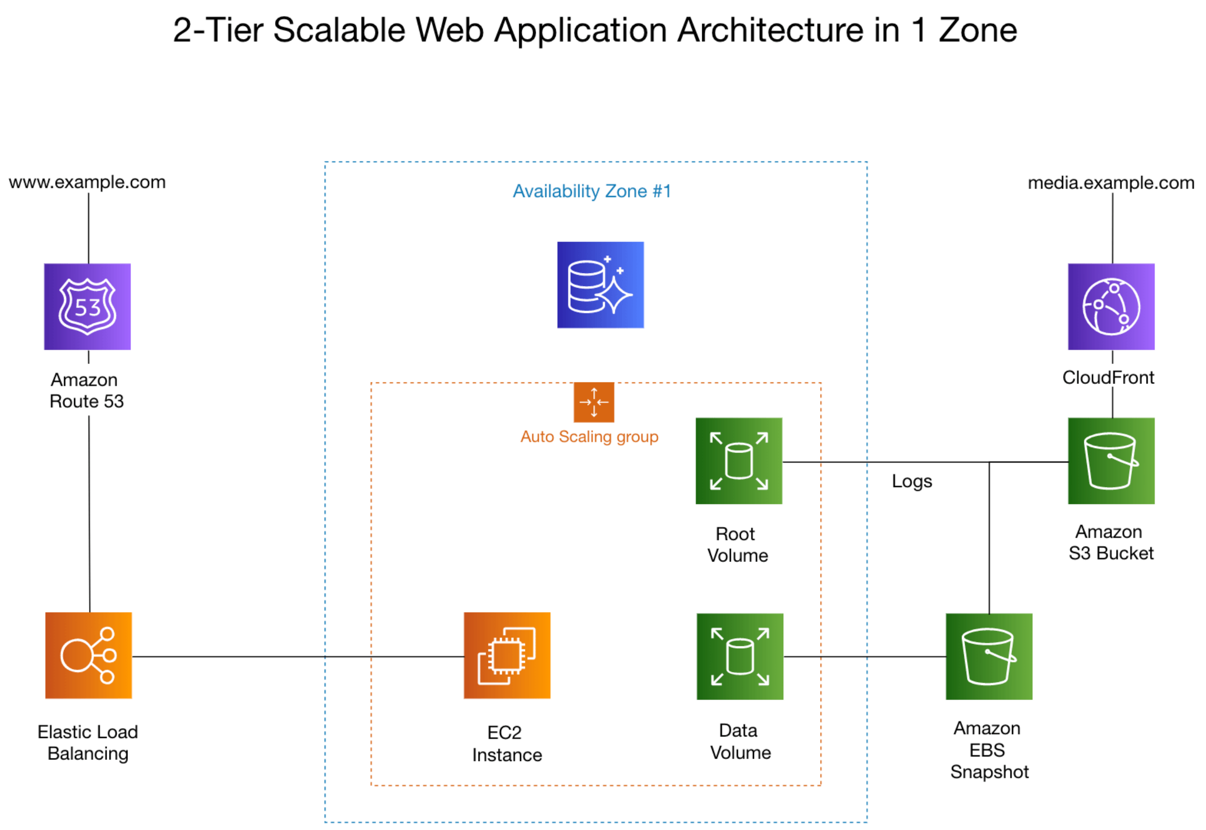
Sample Application

Use the zip file attached to tackle the following task.

Your Task

- Containerize the sample application using `docker-compose` for local development
  - Enable HTTPS for the webserver.
- Enhance the `docker-compose` stack further by choosing one of the following areas to implement:
  - Proxy Layer
    - Choose at least one of: Authentication, (or) Rate Limiting, (or) WAF, (or) Caching, (or) a feature of your choice.
      - While we prefer nginx, you may consider Kong, HAProxy, Traefik, or a proxy solution of your choice.
    - You are also required to document a minimum of two service-level indicators for the Proxy layer, as well as your methodology of defining a service-level objective for those indicators.
  - Monitoring
    - Deploy Prometheus as part of your `docker-compose` stack, and configure it to pull metrics from the instrumented Rails web application.
    - Metrics can be retrieved by requesting http://localhost:3000/metrics
      - While we prefer Prometheus, you may consider Graphite, or InfluxDB. You will then have to instrument the Rails app accordingly.
    - You are also required to document a minimum of two service-level indicators for the reference Rails web application, as well as your methodology of defining a service-level objective for those indicators.
  - Logging

        - Your goal is to implement a log forwarder that will send logs to an external end-point: https://example.com.
            - While we prefer fluent-bit or fluentd, you may consider filebeat/logstash.
- Create a production-ready Docker image for the Rails application.
- Diagram a production-grade infrastructure architecture for the sample application. Try to be as detailed as possible!
    - Note that the intention is not to run the `docker-compose` stack on production.
    - What we seek to understand is your approach to network partitioning. Also, note up any single points of failure in your design.
    - Example diagram (depicts AWS, but any provider is fine; even bare metal):



Stretch Goals

- What changes would you make to the architecture diagram in order to accommodate multi-tenancy SaaS requirements?
- Create a "roll-out plan" and document the necessary steps to go live in the new environment. Try to be as detailed as possible!

Delivery

- The assignment should be delivered as a git repository.
- Submit the project as a ZIP file including a README using the template provided below.
- Once completed, upload your submission to the link provided in our email.

Notes

- While we prescribe technologies based on our own technical alignment, you should choose implementations with which you have familiarity.
- Your primary focus is to present a clear solution, backed up by the steps you took to get there. Documenting your thought-processes helps us to understand how you reached your solution. We want to see what parts of the challenge excite you and where you would prefer to spend your time when approaching a challenge like this.
- Please work independently without code review by others—state any assumptions you made or questions you had along the way. When complete, let us know how much time you ended up spending from beginning to end. There's no limit, but we don't want you to overdo it for us. *We think that 4-6 hours is reasonable.*
- If you get stuck or need more information, please reach out for clarity. We are more than happy to help.

Note: Our intent is definitely not to solicit free work—it's to evaluate your approach to solving real challenges you would be faced with if you entered into this role. If you'd prefer to demonstrate your skills with a completely unrelated company and industry, or would rather share past examples, we're cool with that.

THINKIFIC

README Template

(Copy and paste the following template to create your README file)

## Notes

Please do not supply your name or email address in this document. We're doing our best to remain unbiased.

### Date

The date you're submitting this.

### Location of deployed application

If applicable, please provide the url where we can find and interact with your running application.

### Time spent

How much time did you spend on the assignment? Normally, this is expressed in hours.

### Assumptions made

Use this section to tell us about any assumptions that you made when creating your solution.

### Shortcuts/Compromises made

If applicable. Did you do something that you feel could have been done better in a real-world application? Please let us know.

### Stretch goals attempted

If applicable, use this area to tell us what stretch goals you attempted. What went well? What do you wish you could have done better? If you didn't attempt any of the stretch goals, feel free to let us know why.

### Instructions to run assignment locally

If applicable, please provide us with the necessary instructions to run your solution.

### What did you not include in your solution that you want us to know about?

Were you short on time and not able to include something that you want us to know about? Please list it here so that we know that you considered it.

### Other information about your submission that you feel it's important that we know if applicable.

### Your feedback on this technical challenge

Have feedback for how we could make this assignment better? Please let us know.