Automated and Manual Grading of Web-Based Assignments

Matthew Peveler
Rensselaer Polytechnic Institute
Troy, NY, USA
pevelm@rpi.com

Evan Maicus
Rensselaer Polytechnic Institute
Troy, NY, USA
maicue@rpi.edu

Barbara Cutler
Rensselaer Polytechnic Institute
Troy, NY, USA
cutler@cs.rpi.edu

ABSTRACT

Grading web-based assignments poses many unique challenges when compared with other types of programming assignments. For introductory courses, grading involves not just validation of source code, but also performing some level of functional testing through a browser environment, where one clicks on content, and validates the browser state. In upper level courses, assignments increasingly use several different services, such as a web server and database, running concurrently, each potentially exposing ports for user access. Finally, for some assignments where students are encouraged to be creative, an instructor must then be able to view and interact with the running code, which has historically meant downloading, setting it up on their local machine, and running it – which can prove burdensome. In this work, we present a system that can perform the task of automated grading, create a web-accessible environment on the server for the instructor to manually grade the assignment, and that can scale to handle many concurrent instances of both. To evaluate the effectiveness of our system, we will demonstrate its usage within the context of several assignments ranging over several different levels and courses of our Information Technology curriculum.

CCS CONCEPTS

• **Social and professional topics → Information technology education**; **Student assessment**.

KEYWORDS

autograding, autograding platforms, information technology

ACM Reference Format:
Matthew Peveler, Evan Maicus, and Barbara Cutler. 2018. Automated and Manual Grading of Web-Based Assignments. In *SIGCSE '20, March 11–14, 2020, Portland, OR.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/1122445.1122456

1 WEB GRADING CHALLENGES

The most significant challenge is how to run and test multiple web assignments in parallel without them being able to see or affect each other. Secondly, these assignments may require multiple running services (e.g., a web server, database) to function. Thirdly, these assignments may want or need to access specific websites and external APIs for content, but should not be allowed to access the entire web. Finally, there are considerations for how to perform automated grading and facilitate manual grading. For automated grading, assignments are evaluated through analysis both of source code as well as through functional tests that are run through a browser, but that require the students follow specific criteria for creation of their content. With more complex scenarios, it may be necessary for an grader to view and interact with the student's application, ideally in a prepared environment that does not require them to download and locally configuring the student's code and its dependencies and considering the potential security implications.

2 IMPLEMENTATION OVERVIEW

Our system utilizes Docker and its concepts of containers, which are lightweight virtualized machines. Student code is built, getting any external dependencies, and then run within a container. Through the use of docker networks [1], we can effectively deal with our first two challenges. These networks can be setup to be isolated from each other, preventing students from accessing each other's assignments. Additionally, within a network, an instructor can specify the linking of concurrent services while only allowing an entry point into the network via the host machine. To handle automated grading, we utilize a setup similar to WebWolf [2], where the host machine uses the Selenium automation framework to drive a headless web browser, interacting with a student's page and running assertions about the state of a page over time. To handle both allowed web usage and manual grading, we inject a proxy layer outside the network to monitor incoming and outgoing traffic. For outgoing traffic is restricted to domains on an instructor-specified whitelist or a cache. For incoming traffic, e.g., manual grading, the proxy provides HTTP authentication to ensure only the instructor can access the content and ensure the network is created when needed and destroyed when no longer needed.

3 CONCLUSION

In this work, we present a system to handle both automated grading and assist in manual grading of web-based assignments which may require multiple running services. This system can be used to handle grading in a secure fashion of many assignments concurrently. For manual grading, the system allows graders to run and view a student's running application code without having to download and set it up on their local machine.

REFERENCES

[1] Evan Maicus, Matthew Peveler, Stacy Patterson, and Barbara Cutler. 2019. Autograding Distributed Algorithms in Networked Containers. In *Proc. of the 50th ACM Technical Symposium on Computer Science Education*. ACM Press, Minneapolis, MN, USA.

[2] Antonio Carvalho Siochi and William Randall Hardy. 2015. WebWolf: Towards a Simple Framework for Automated Assessment of Webpage Assignments in an Introductory Web Programming Class. In *Proc. of the 46th ACM Technical Symposium on Computer Science Education*. ACM Press, Kansas City, Missouri, USA.

SIGCSE 2020, March 2020, Portland, Oregon, USA

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

POSTER CONTENT

Our poster layout will contain five columns, which from left-to-right will have the following content:

(1) Our abstract (as above) and a high-level overview of the challenges facing grading of web-based assignments
(2) Column with text and images describing our system setup. The images would showcase the concept of isolation between networks, and how the proxy layer works between client and the networks.
(3) Column demonstrating how the automated grading through Selenium toolkit with images of web pages with content highlighted and overlaid with text to showcase, and the bottom images of the instructor workflow for interaction with the manual system.
(4) Column describing via text and images several different assignments of varying complexity from the courses offered at Rensselaer Polytechnic Institute, where they are:
    (a) Basic assignment that is just HTML, CSS, and JavaScript and that can be viewed and rendered without a web server
    (b) Assignment using PHP for rendering and served via a web server, and that makes calls out to the Open Movie Database API.
    (c) Assignment NodeJS with separate front and backend servers and that utilizes a database.
(5) Future work, references, and a description and overview of the larger autograding system, Submitty, this is incorporated into.

The selected assignments will be used as broad demonstrative examples to help spur discussion of various other services that students might want to use, such as a message queue, key-value store, etc., in their assignments, as well as generalization into other programming languages.

AUTHOR BACKGROUND

Barbara Cutler is a professor, while Matthew Peveler and Evan Maicus are PhD students, all in the computer science department at RPI. Collectively, they have published 12 works in the field of automated grading and course management systems.