THE PRIMITIVES

SHAKEDOWN

PRODUCT AND BUSINESS CASE

BUTTERWORTH PRODUCT DEVELOPMENT COMPETITION IN ICS
&
BEALL STUDENT DESIGN COMPETITION IN ENGINEERING

THE PRIMITIVES

Product Description

Shakedown is a web application that provides an environment for computer science instructors to test student's programming assignments and automate the grading process. It modernizes and simplifies the way instructors teach programming courses and improves students' learning through instantaneous feedback.

Instructors and students register for Shakedown's subscription service which provides a cloud-based solution for managing programming courses. The system stores submitted code and automated test results. It features an intuitive user interface in mind to make it accessible to everyone.

Instructors create courses which provide a place for students to register. They create assignments with test code that is used to automatically evaluate student submissions. As students submit solutions, instructors view and download both the code and the results, simplifying administrative tasks. Additionally, instructors can analyze trends through the statistics that Shakedown collects from student submissions. This allows instructors to spend more time providing qualitative feedback for students, creating a well-rounded learning environment for students.

Students finally have a central hub where they can find all their programming assignments information. They upload their submissions to assignments and receive instantaneous feedback from instructor provided tests. Depending on how the instructor configured the assignment, students may inspect public test cases for additional guidance or resubmit work to improve their scores and understanding.

Market Demand

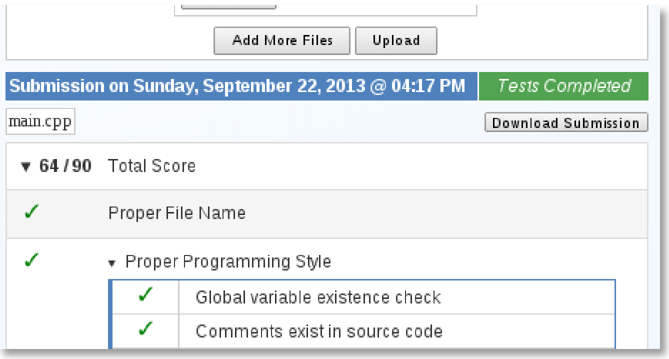
Shakedown organizes automated testing around the classroom environment and provides a centralized place for instructors to view code and results. It offers a scalable distributed system to support many users and plugin architecture to support different languages. Some competitors push instructors to evaluate particular parts of students' work. Since different instructors and courses have distinct grading methods, Shakedown allows customized plugins to accommodate each instructor's style. In addition, competitors' user interfaces are outdated and unintuitive, and lack customer support and documentation for users.

Shakedown's business model is sustainable as it is designed to be offered as a service, rather than a one-time purchase. Shakedown targets software development, a rapidly growing field with plenty of new potential customers, assisting its continued growth. Since Shakedown's services cater to the educational space, the constant supply of new students allows for consistent growth and profits.

Shakedown is designed to be used in programming classes by both instructors and students. Currently, most programming instructors at universities and high schools across the United States manually grade their programming assignments or individually execute automated tests, with delayed feedback for students. Shakedown aims to improve upon this problem by allowing programming assignments to be graded automatically, with results aggregated and reported in real time. Students will get near instantaneous feedback upon submission, allowing them to learn and improve faster than by traditional means.

Competitors

Other web based solutions exist that attempt to solve similar problems. Shakedown embodies knowledge learned from their weaknesses to create greater value for users. Below is a summary of three open source systems in use at various US universities:



Add More Files | Upload

Submission on Sunday, September 22, 2013 @ 04:17 PM — Tests Completed

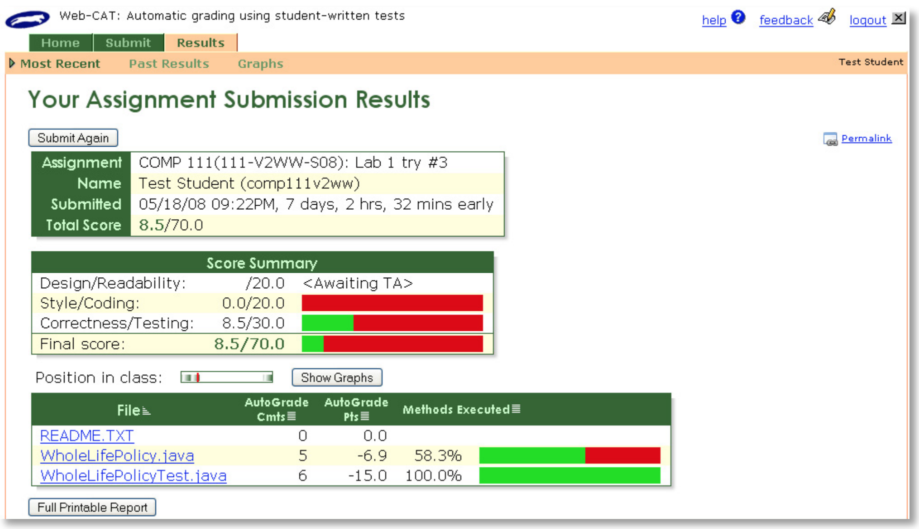
main.cpp — Download Submission

▾ 64 / 90	Total Score
✓	Proper File Name
✓	▾ Proper Programming Style
✓	Global variable existence check
✓	Comments exist in source code

Galah

Galah is used in programming courses at UC Riverside but only tests in C++. Instructors are only able to interact with the system through the command-line utility, which can only work on Linux machines. For students there is a web page to see assignments and scores on their assignments. Galah is open source and free to use, but it must be deployed by the instructor themselves.

Galah uses a plugin software architecture, but expects the plugin writer to know the Python programming language. The plugin architecture allows instructors to test code in different languages, but since the test harnesses are written in Python, the instructor can't use a programming language's own testing framework. In addition, instructors have to deploy their own server to perform tests. This means that students will not have a central location to go to where they can find all their programming courses. There is no publicly accessible information regarding Galah's turnaround time or performance under load.



Your Assignment Submission Results

Assignment	COMP 111(111-V2WW-S08): Lab 1 try #3
Name	Test Student (comp111v2ww)
Submitted	05/18/08 09:22PM, 7 days, 2 hrs, 32 mins early
Total Score	8.5/70.0

Score Summary		
Design/Readability:	/20.0	<Awaiting TA>
Style/Coding:	0.0/20.0	
Correctness/Testing:	8.5/30.0	
Final score:	8.5/70.0	

Position in class:

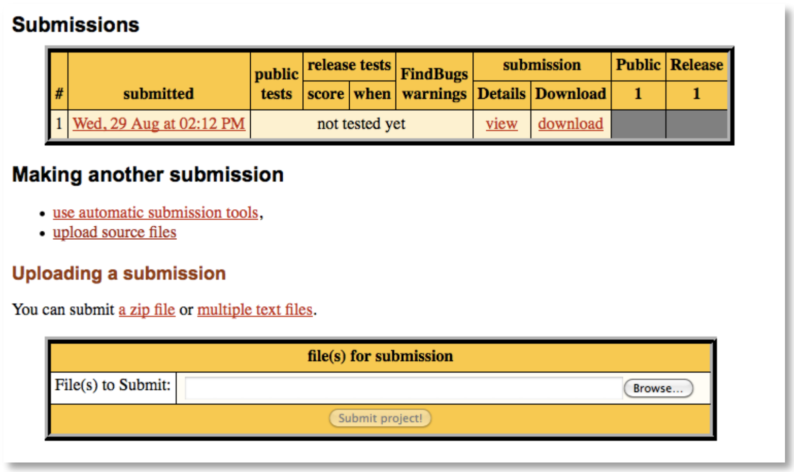
File	AutoGrade Cmts	AutoGrade Pts	Methods Executed
README.TXT	0	0.0	
WholeLifePolicy.java	5	-6.9	58.3%
WholeLifePolicyTest.java	6	-15.0	100.0%

Web-CAT

Web-CAT is maintained at Virginia Tech and is used by at least six universities. As opposed to instructors evaluating students' code through tests, students' scores are calculated based off of how well they test their own code.

Instructors are able to manually grade code and add comments in Web-CAT. Even though Web-CAT has a plugin architecture, there is no clear or simple way to create a plugin for their system. Web-CAT is very opinionated on how it evaluates the grading of students and doesn't provide any way for instructors to automate their grading. The Web-CAT interface is very cluttered and outdated, making it harder to use.

Furthermore, Web-CAT does not scale well and instead has long submission queues. In fact, maintainers suggest that students budget wait times of up to 75 minutes before the deadline for classes with around 100 students. Shakedown's distributed system allows for student submissions to be ran in parallel to address huge usage spikes.



Submissions

#	submitted	public tests	release tests		FindBugs warnings	submission		Public	Release
			score	when		Details	Download	1	1
1	Wed, 29 Aug at 02:12 PM	not tested yet				view	download		

Making another submission

- use automatic submission tools,
- upload source files

Uploading a submission

You can submit a zip file or multiple text files.

file(s) for submission	
File(s) to Submit:	Browse...
Submit project!	

The Marmoset Project

The Marmoset Project is a web-based tool created and used at the University of Maryland. Initially, student code is executed against public tests shared by the instructor. Once students' code passes these tests (resubmissions are encouraged), they can be evaluated against graded tests. Students are allowed to submit until they used up all of their "tokens", encouraging students to submit assignments earlier and learn from their errors.

The Marmoset Project allows for manual code reviews in which instructors can provide feedback to students about the code.

Conclusion

A common flaw between all the competitors is the lack of a out of the box solution that works for professors - they are required to deploy servers and configure settings. Their user interfaces are not geared towards improving the process of grading programming assignments for instructors and students. They lack the amount of support to make it a complete service instructors can rely on.

Shakedown focuses on simplicity in plugin interfaces without sacrificing expressiveness. Only the essence of testing and analysis operations are required in the plugin interfaces. As a result, users can easily develop plugins to support new languages or frameworks.

Security is also a top priority in Shakedown that only Galah seems to address. Shakedown prevents students from writing code that accesses restricted information (solutions, application code, etc.) or “phone home” for information (such as unit test input) over the network. Each student's submission is processed in isolation with minimal permissions and all remnants are removed before the next student's code is evaluated.

Revenue

For Shakedown to penetrate the market, the first step is to partner with educational outreach programs from large software companies to increase awareness. As Shakedown reaches a larger audience, more schools and instructors will register for Shakedown’s software as a service model. A large cost of the system, the prototype, has already been developed, so Shakedown has a low time to market and can quickly begin generating revenue. Shakedown’s revenue model is a subscription based service, similar to online interactive textbook services. As such, it will be priced competitively at $40 per term per student. Since the subscription does not restrict the number of courses per quarter, multiple instructors at the same institution will be encouraged to concurrently use the product.

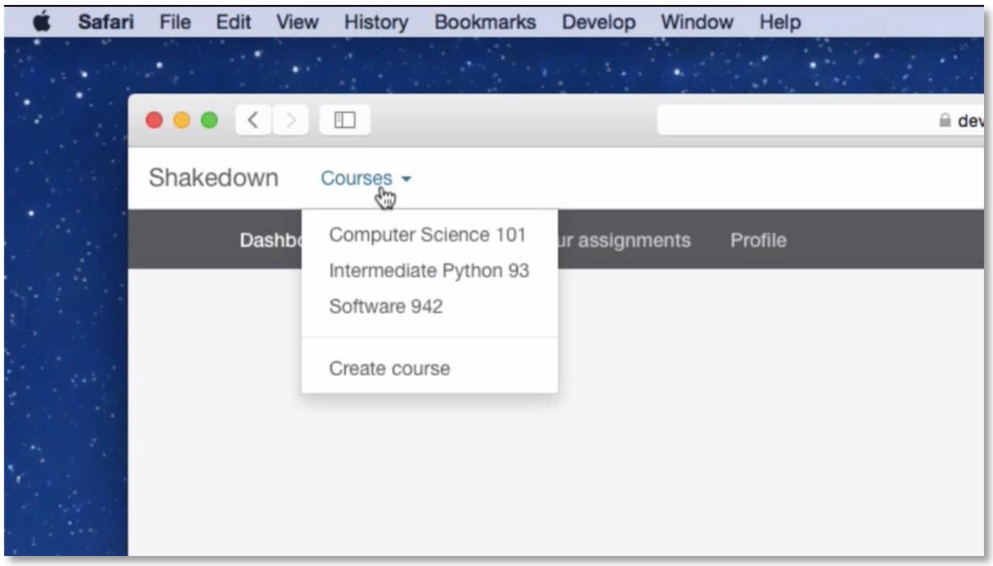
Microsoft Azure will be used for running the service. Azure will run two web servers (for redundancy), costing a total of $29.76 a month. As the business grows, it might be necessary to upgrade the capacity of these servers, but it is not expected to be a driving cost. Additionally, Linux virtual machines will run the test runners, costing $14.88 per month per machine. This system of virtual machines is designed to scale as needed, and so the cost will increase linearly as Shakedown grows. Only one machine is required to handle courses with hundreds of students. In the unlikely scenario that hundreds of students submit code near simultaneously, a second machine will be automatically provisioned for immediate use. Azure charges per minute, so such bursts in usage will inflict minimal costs. Since Microsoft extends a monthly $150 Azure credit through its BizSpark program for early stage businesses, computing costs will remain zero until a significant user base is established.

While early customer acquisition and relations will be handled by a core team of founders, as Shakedown scales these costs will drive the total cost. A typical customer acquisition will take one calendar month and 20 hours of a salesperson’s time. At least one engineer will likely attend both an initial visitation and a follow up after a deployment, totalling 6 hours. Once an instructor has been acquired as a customer, he or she will add Shakedown as a required resource for the course, and will effectively acquire the students as customers on the business’s behalf.

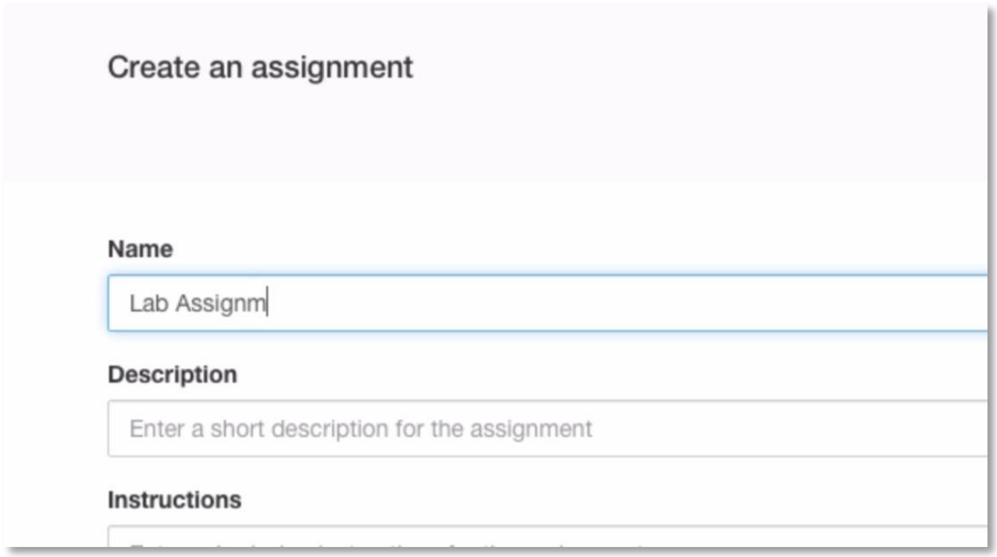
Future development for increased quality and entirely new features will also be a significant cost at the market rate for software development personnel.

User Interface

Shakedown’s user interface allows exploring courses and their assignments, managing assignments, and annotating students submissions. The entire website is built on modern web standards (HTML5, JavaScript) made to be responsive and accessible from phones, tablets, and desktops.

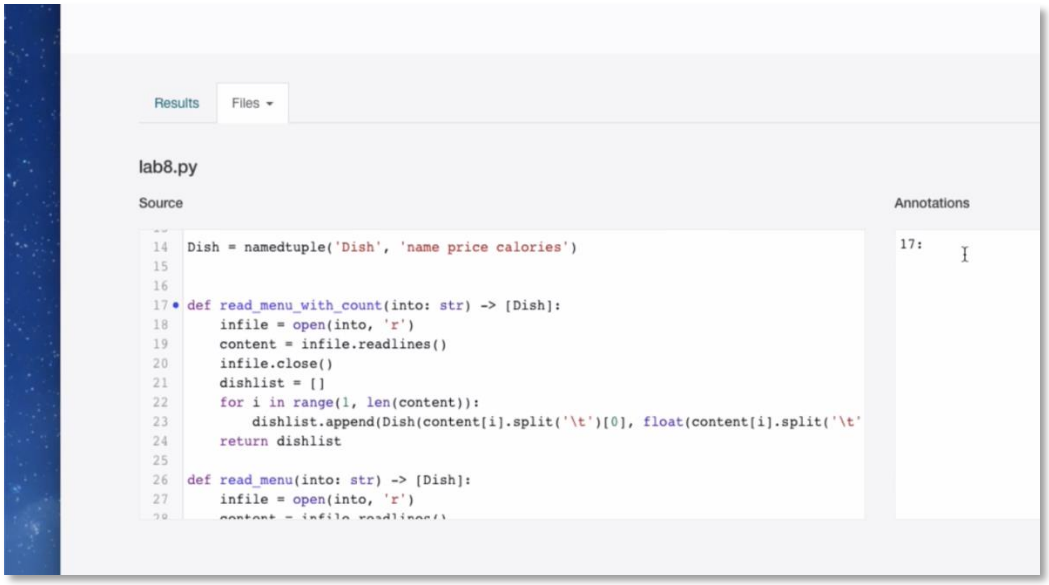


Shakedown’s interface for exploring courses and assignments behaves like a familiar file explorer. Students can easily switch between their current classes and assignments in different contexts to account for many different ways a student might expect to use Shakedown. Visually, Shakedown is colored with light whites, grays, and blues that highlight important information. This includes information like passing or failing tests and new assignments released by an instructor. Shakedown has gone through many redesigns focused on making the user interface more intuitive. A central goal is creating a unified experience for both instructors and students.



Shakedown’s interface for managing assignments presents a consistent layout for both instructors and students, with instructors given more controls and privileges. An instructor can create an assignment for a class by navigating to the class page and configuring details for the assignment. Assignment details like due dates, descriptions, and requirements can be edited at any time.

On an assignment page, students see the details for an assignment and a button to submit their files. Instructors see a list of students who have submitted code for the assignment. The instructor can view each student's code in the browser.



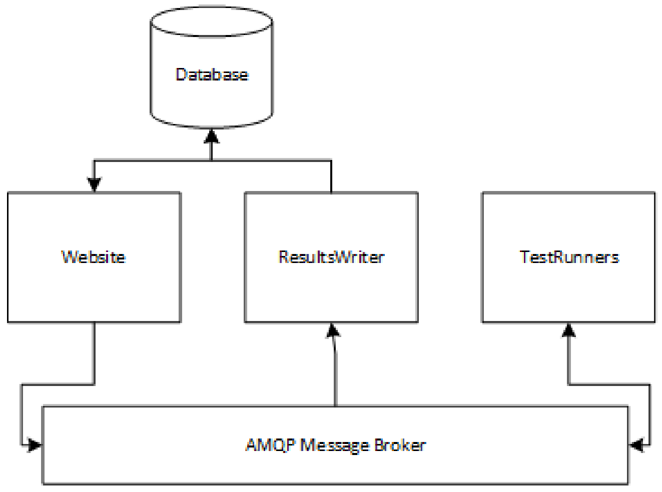
```python
Dish = namedtuple('Dish', 'name price calories')


def read_menu_with_count(into: str) -> [Dish]:
    infile = open(into, 'r')
    content = infile.readlines()
    infile.close()
    dishlist = []
    for i in range(1, len(content)):
        dishlist.append(Dish(content[i].split('\t')[0], float(content[i].split('\t'
    return dishlist

def read_menu(into: str) -> [Dish]:
    infile = open(into, 'r')
```

Before, instructors would have to download student code before viewing it. Shakedown's in-browser code view allows instructors to leave comments visible to the students, allowing personalized feedback.

Product Architecture

Shakedown's architecture was designed to solve problems present in competing systems. Namely, a plugin architecture and distributed system for running the tests were incorporated to deal with spikes in submission volume and support for any programming language. As a web application, Shakedown uses a client-server architecture to handle interactions between the users and the system.

THE PRIMITIVES



Every module in the block diagram can exist on separate computers. In fact, each module itself can be split across multiple machines. In such a configuration, there is no single point of failure. For example, if one TestRunner fails or requires an upgrade, other TestRunners and the system can still accept jobs. Modules are executed within virtual machines and the TestRunners, which execute arbitrary student code, do not have direct access to the database, providing last-resort security to our system.

The Website module encapsulates the user interface. Users make a request to the server for the website when they type in the URL. Once the user is authenticated and enrolled in a class, a user can then upload their code from the website to get their results. Once their results are acquired, it is then displayed on the website for the individual student to see.

The message broker delivers all messages between modules using the Advanced Message Queuing Protocol (AMQP). For example, the Website module sends student test code to the broker, which delivers student test code to a single TestRunner. The TestRunner will then send the results to the broker, which relays the results to the ResultsWriter. If the TestRunner fails for any reason (such as a machine failure), the broker will simply resend the message to a different TestRunner.

The TestRunners receive the student code from the broker and execute tests. A plugin architecture allows any language to be tested with Shakedown. Currently, supported plugins include support for Java, Python, and Haskell. The TestRunner chooses the appropriate plugin depending on the language and test framework. The plugin (usually itself using an existing third party testing framework) actually executes students code and evaluates the raw results. The TestRunner then packages the results and sends them back to the broker.

The ResultsWriter, receiving the test results from the broker, writes the results to the database and notifies the website, which sends the user a notification that results are viewable.

Since Shakedown is a web application, it can be run from any modern web browser. It has been tested with Google Chrome, Safari, and Internet Explorer 10 and 11. As a cloud application, there is little to no set-up or installation required.

Prototype Development

The team created a user-facing website for both students and instructors to interact with the system. A scalable, distributed system was created to handle executing tests for code submitted by students. Additionally, a process for reliably receiving and storing results was created. All this functionality is present in the modules pictured in the block diagram.

Shakedown uses Microsoft ASP.NET MVC Framework, HTML, CSS, Javascript, Bootstrap, AJAX, SignalR, RabbitMQ, and Microsoft SQL Server among other lightly used libraries.

Each module has its own unit testing suite. The web application is tested live on a special development domain with integration tests that mimic a user's actions on the site. Every time a change is made, integration tests are executed for every action a user could perform on Shakedown's site, including those that involve multiple modules.

Shakedown's prototype has already begun confirming market need. During a one week deployment of Shakedown in a UC Irvine introductory course, over 50 pairs of students used the website to gather information about the correctness of their code. Students improved their code based on the feedback that Shakedown provided. As well as confirming product need, the deployment also confirmed assumptions the tests made about user load. The entire operation was comfortably handled with one TestRunner running.