СУ “Св. Климент Охридски”,
Факултет по математика и информатика,
Катедра Информационни Технологии

Разпределена система за управление на курсове по програмиране с автоматично оценяване на решения

Здравко Иванов Гюров, РСМТ, ФН: 26379
Научен ръководител: Стоян Велев

София, 2022

Съдържание

I. Списък на съкращенията

API - Application Programming Interface
WYSIWYG - What You See Is What You Get

II. Речник на термините

Application Programming Interface - проложно-програмен интерфейс
Online Judge - онлайн система за автоматично оценяване на решения на задачи по програмиране
Java Servlet - програми, които работят върху уеб сървър, те са междинното ниво между заявките, идващи от уеб браузъра и базите от данни или други приложения на уеб-сървъра
plug-in - софтуерент компонент, който представлява “приставка”, която се инсталира в допълнение към съществуващо софтуерно приложение, за да предостави допълнителна функционалност
What You See Is What You Get - системи, при които съдържанието по време на редактиране изглежда като крайния резултат

III. Списък на фигурите

Asd

IV. Списък на таблиците

Asd

1. Увод

Обратната връзка за функционалната коректност, ефективност и придържане към конвенции за стил и добри практики е ключова за изграждането на знания и умения в курс по програмиране. Регулярната обратна връзка и особено възможността студентите сами да проверяват и подобряват итеративно решенията си, са възможни единствено чрез автоматизиране на процеса по тестване и оценяване. С увеличаването на броя практически курсове по програмиране и броя студенти през изминалите години, тези практики стават все по-наложителни за подпомагане на преподавателския състав да бъде по-полезен и да предоставя персонална помощ за разрешаване на по-сложни казуси със спечеленото време.

Целта на дипломната работа е да се проектира и реализира разпределена система за управление на курсове по програмиране с автоматично оценяване на решения.

Структура на дипломната работа

В глава **2. Обзор на подобни системи** ще разгледаме няколко системи, които покриват изискванията ни до някаква степен. След това ще ги сравним и ще съпостави техните плюсове и минуси и най-накрая ще обобщим защо те няма да ни свършват работа.

В глава **3. Проектиране на системата** ще разгледаме функционалните и нефункционалните изискванията на системата и как ще бъдат изпълнени и предоставени.

В глава **4. Реализиране на системата** ще разгледаме конкретните технологии, с които ще бъде имплементирана системата и защо точно те са избрани. Ще

видим също и основните API-та и как може един потребител да работи със системата

В глава **5. Експерименти и анализ на резултатите** ще разгледаме различни тестове, които са били приложени на системата за да се симулира реална работна среда и ще направим разбор на резултатите.

В последната глава **6. Заключение** ще направим обобщение, ще видим как системата ще влезе в употреба и ще разгледаме възможни подобрения и бъдещо развитие на системата.

Дефиниции

X наричаме У

2. Обзор на подобни системи

В тази глава ще разгледаме едни от най-популярните приложения и уеб услуги, които покриват възможно най-много от изискванията ни. Първо, нека определим кои са тези най-важни за нас функционални и нефункционални изисквания, които ще действат като критерии за сравнение.

1. Системата трябва да ни предоставя възможността да управляваме много на брой курсове (стотици, хиляди, а може и повече, за момента не се интересуваме от конкретен брой).
2. Всеки курс трябва да може да съдържа голямо количество задания (десетки, това отново е приблизително число).
3. Системата трябва да предоставя ясен и лесен начин за работа с предадените решения.
4. Системата трябва да дава възможност на преподавателския състав да проверява и оценява решенията по интуитивен начин.
5. Системата трябва да може да проверява решенията на студентите автоматично.

6. Системата трябва да може да улавя признаци на плагиатство у решенията на студентите.
7. Системата трябва да е гъвкава в отношение на технологиите, които се преподават в курса. Целта е да намерим една система, която може да се използва за всички ситуации, която ще е удобна за всички преподаватели и ще стане позната на студентите.
8. Системата трябва да дава свобода на лекторите сами да управляват курсовете си без намеса на администратор.
9. Системата трябва да е възможно най-достъпна финансово както за преподавателите, така и за студентите.
10. Системата трябва да е достъпна 24/7.
11. Системата трябва да е устойчива.
12. Системата трябва да е лесна за използване.

След като сме въвели тази основа, може да преминем към приложенията.
За този анализ са подбрани както системи, които са били в експлоатация дълго време и са се доказали като едни от най-добрите за времето си, така и новонавлезли системи, използващи модерни и иновативни подходи и технологии, а именно:

- Web-CAT
- codePost
- GitHub Classroom

Web-CAT

Web-CAT е гъвкава и приспособима система за оценяване, която е предназначена да обработва задания по програмиране.
Web-CAT е програма, която върви на сървър и предоставя възможностите си чрез уеб интерфейс. Всички дейности свързани с решенията на заданията, обратната връзка, разглеждане на резултатите и оценяване се извършват в уеб браузър. Всички административни дейности на инструкторите свързани с

курсовете, заданията и оценяването също се извършват в уеб браузър, дори системните административни дейности се извършват по този начин.
Преди създаването на Web-CAT са съществували и други системи за автоматично оценяване, но те обикновено са се фокусирали над определяне дали изходният код на студента произвежда желания резултат. Това са така наречените Online Judges. Web-CAT оригинално е била проектирана като автоматична система за оценяване с общо предназначение, но преди да се завърши първоначалната версия, нейните автори са решили, че искат по-скоро да поддържат дейности по тестване на софтуера на студентите, отколкото да оценяват работата на студенти чрез просто сравнение на резултатите.

Основни силни страни

Сигурност (Security)
На първо място, системата е проектирана да поддържа сигурна работа. Нейният потребителки модел включва подход за удостоверяване, базиран на приставки с отворен API, така че администраторът може да избере една от няколко вградени стратегии за удостоверяване или дори да предостави персонализирана(custom). Услугите на Web-CAT са предпазени едновременно от специфични права на ниво потребител и от система за контрол на достъпа на ниво роля. Разпознаването на злонамерени студентски програми и предпазването от тях идва по подразбиране. Цялостността на данните се поддържа от политики за сигурност в системата и от услугите предоставени от релационна база данни.

Преносимост (Portability)
Web-CAT е приложение, написано на Java, което дава на кода висока степен на преносимост. То може да бъде пакетирано и разпространявано като Java Servlet приложение, което ще работи под всеки съвместим сървлет контейнер като Apache Tomcat. Когато е пакетиран като сървлет, Web-CAT може да се разпространява като единичен .war файл (сервлет уеб архив), който включва всички необходими зависимости.

Разтегливост (Extensibility)
Може би най-голямата сила на Web-CAT е вродената гъвкавост и разширяемост, вградени в неговата ахитектура. Приложението е проектирано с напълно plug-in базирана архитектура. Основни функционалности могат да бъдат добавени без промяна на код в съществуващата система, а всички съществуващи възможности на Web-CAT се реализират в няколко плъгина. Приложението е напълно неутрално по отношение на езика. Web-CAT се използва за оценяване на решения написани на Java, C++, Prolog и други. В допълнение, системата е напълно неутрална по отношение на това как се оценяват заданията и каква обратна връзка се връща на студентите.

Ръчно оценяване
Web-CAT се справя с всички автоматизирани задачи, които инструкторът иска да се извършват за да се обработват студентските решения, но приложението също има вградена поддръжка за задачите за ръчно оценяване. Преподавателският състав може да пише коментари и предложения на студентите за техните решения, може да добавя или премахва точки директно в HTML изгледа на изходния код на студента по WYSIWYG начин. Студентите се уведомяват автоматично по имейл, когато ръчното оценяване на тяхното решение е завършено, за да могат да го разгледат.

~(Edwards, Stephen. “What is Web-CAT?”)

Опит със системата

След разглеждане на документацията на Web-CAT виждаме защо тя е една от най-използваните системи от рода си. Системата е доста близко до това, което търсим, но за да направим цялостен и изчерпателен анализ, трябва също да се използва, за да се потвърди, че действително се предоставят всички тези функционалности. Също трябва да я съпоставим с нашите изисквания.

След използване на системата 8 месеца (4 като студент и 4 като асистент) като кратко обобщение може да се каже, че системата изпълнява добро количество от желаните функционалности, но определено може да се желае доста повече. Първо, започвайки с това как изобщо един лектор може да се сдобие с това приложение, за да го използва за своя курс по програмиране. Тъй като това е приложение, а не уеб услуга, преподавателят ще трябва сам да се погрижи за сервирането на това приложение до външния свят. Алтернативно би могло да има една инстанция на това приложение за целия факултет или университет, така може да има системен администратор, който да е посветен на тази дейност. Хостването на приложението няма да бъде безплатно, а и освен административните дейности в самата система, ще има и усилия за поддържане на самото приложение работещо 24/7 дори при голямо натоварване.

Приемайки, че вече има сервирана система за използване, следваща стъпка би била регистрация на потребители, както лектори и асистенти, така и студенти. Това става ръчно, като за това отново се грижи системният администратор. Този процес е доста досаден, времеемък и склонен към грешки.

При създаване на курсове и задания не може да се пишат дълги описания или условия, така че тази дейност трябва да се извършва в друга система. След създаване на задание, потребителите виждат само името му, което може да се кликне. Това води към друга страница, в която студентите могат да си качат решението като .zip архив. Работата със zip архиви за контрол на версиите на решенията на студентите е много неудобна. Докато решават задачата, студентите постоянно правят промени по решенията си, дооправят бъгове и качват нова версия на решението си, което е напълно нормално. За една задача те могат да имат голям брой версии, което става трудно за следене. Те трябва да знаят в коя версия в кой архив им се намира. Също какви точно промени са направени в определена версия и още много други проблеми свързани с контрол на версията на изходен код. След като си качат архива с финалната версия в системата не излиза ясно и видимо кода, който се съдържа там и за да си сигурен, че си качил правилната версия трябва да си изтеглиш решението, да го разархивираш и да го разгледаш цялостно, което е голямо неудобство. В допълнение, от гледна точка на преподавателския

състав, когато някой студент иска помощ, неговото решение отново трябва да бъде свалено и разглеждано локално.
При проверка на решения се вижда така наречения WYSIWYG интерфейс, който звучи много удобен и интуитивен, но на практика бъгове в потребителският интерфейс го правят доста неудобен и труден за ползване. Доста неприятно и произволно, но често явление е да си на един клик от страница за срив.
Срещат се и сериозни технически ограничения, примерно при задания свързани с упражняване и тестване на знанията на студентите за многонишково програмиране. Изпитват се затруднения при стартиране на множество нишки.
Накратко, системата може да свърши работа, но си личи, че не използва модерни подходи и технологии и определено звучи по-добре отколкото изглежда.

codePost

…
Да опиша всяка система, като покривам всяка една от 12-те точки отгоре и дали системата предоставя тази функционалност или не и накрая след 5те описания да направя табличка с 12те точки ясно да се вижда +/-

Подобни системи
Сравнителен анализ (+/-)
Обобщение - защо тези няма да ни свършат работа

3. Проектиране на системата

Да има диаграми, архитектурна, дб ,флоу

4. Реализиране на системата

Технологии, защо съм ги избрал

Да опиша основните апита

Снимки на ui-а

5. Експерименти и анализ на резултатите

Asd

6. Заключение

Постигнати резултати

Проучени са системи за …

Проектирана е система за …

Реализирана е система …

Приноси(научни/научно-приложни/приложни)

Научни - свравнителен анализ на съществуващи системи

Научно-приложни - проектиране

Приложни - имплементацията

Апробация - системата ще се ползва за …

Насоки за бъдеща работа, перспективи

Използвана литература

Edwards, Stephen. “What is Web-CAT?” *Web-CAT*, 1 October 2020, https://web-cat.org/projects/Web-CAT/WhatIsWebCat.html. Accessed 22 March 2022.

Google corp/org name. “Google title.” *Google website title*, 14 March 2022, https://google.com. Accessed 14 March 2022.