

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

FINAL YEAR PROJECT REPORT

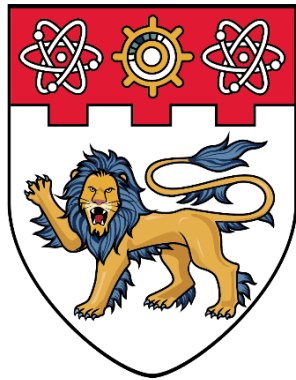
Code Testing and Grading in Learning Management Systems

Wee Jing Wen (U1821631A)

School of Computer Science and Engineering (SCSE)

Nanyang Technological University

2020



**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

SCSE20-0695

Code Testing and Grading in Learning Management Systems

Submitted in Partial Fulfilment of the Requirements
for the Degree of Bachelor of Engineering (Computer Science)
of the Nanyang Technological University

By

Wee Jing Wen (U1821631A)

School of Computer Science and Engineering (SCSE)

2021

Abstract

Learning management systems (LMSs) are increasingly being adopted for use in online learning. While LMSs offer key benefits such as providing a single platform for educators to manage course content, as well as online assessment functionalities such as automated grading, these benefits are not easily available for programming courses. Existing LMSs generally do not offer automated grading for code assessments, which may prompt some educators to use external tools. However, such external tools are often unable to be integrated into existing LMSs, which then diminishes the “single platform” benefit of LMSs if educators were to use such external tools.

This project aimed to implement an automated code testing and grading feature in an LMS, using Reactjs, Node.js, as well as MySQL. The system runs student-submitted code against inputs submitted by educators, and grades the code based on how many test cases passed. This aimed to provide two key benefits. It allows students to receive immediate feedback, thus aiding in their learning, and also saves the time and effort of educators. This thus extends the benefits of LMSs to students and educators in programming courses as well.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr Althea Liang for her patience and support. She provided me valuable advice and guidance which allowed me to understand what was required to carry out this project to the best of my potential, making it a meaningful and fulfilling one.

I would also like to express my thanks to alumni Nicholas Koh En Jian, who provided the base LMS on which my project is based.

Thank you both for explaining and taking the time to help me even during this difficult time of the ongoing pandemic.

Table of Contents

Abstract.....	1
Acknowledgements	2
List Of Figures.....	5
1 Introduction	6
1.1 Background: Increased Adoption of Digital Learning and LMSs.....	6
1.2 Benefits of LMSs.....	6
1.2.1 Single Platform.....	6
1.2.2 Online Assessment Functionalities.....	6
2 Report Organization	8
3 LMSs for Programming Courses	9
3.1 Gaps in LMSs for Programming Education	9
3.2 External Automated Code Grading Tools	10
3.2.1 Case Study of Existing External Automated Code Grading Tool: Web-CAT.....	10
4 Project Objective	11
5 Requirements Analysis	11
5.1 Use Case Diagram	13
6 Technologies Used	13
6.1 Front-end: Reactjs.....	13
6.2 Back-end.....	15
6.2.1 Node.js.....	15
6.2.2 MySQL	15
6.3 Code Testing/Analysis.....	16
6.3.1 Analysis of Code Testing/Analysis Methods	16
6.3.1.1 Static Code Testing/Analysis.....	16
6.3.1.2 Dynamic Code Testing/Analysis	17
6.3.2 Chosen Code Testing/Analysis Method and Rationale	18

7 Implementation.....	19
7.1 Create and Submit Assignments.....	19
7.1.1 Logic.....	19
7.1.2 Assignments with Code Testing Enabled.....	21
7.1.3 Assignments without Code Testing Enabled.....	27
7.2 View Assignments and Submission Scores.....	28
7.2.1 Educators' View	28
7.2.2 Students' View	29
8 Future Work.....	29
9 Conclusion.....	30
10 References	31

List Of Figures

Figure 1: An example of a Web-CAT generated report. Screenshotted from [19]	11
Figure 2: Use Case Diagram.....	13
Figure 3: Screenshot of educator's Add Assignments screen	19
Figure 4: Screenshot of educator's Add Assignments screen when automated code testing is enabled.....	21
Figure 5: Screenshot of example assignment created with code testing enabled.....	22
Figure 6: Screenshot of code used as correct code	23
Figure 7: Screenshot of student's Submit Assignment screen when automated code testing is enabled.....	24
Figure 8: Screenshot of example code submitted by student	24
Figure 9: Screenshot of student's screen after testing code	25
Figure 10: Screenshot of student's screen after testing code against own input.....	26
Figure 11: Screenshot of educator's Add Assignments screen without automated code testing ..	27
Figure 12: Screenshot of student's Submit Assignment screen without automated code testing ..	27
Figure 13: Screenshot of educator's List Assignments screen.....	28
Figure 14: Screenshot of educator's List Submitted Assignments screen	28
Figure 15: Screenshot of student's List Assignments screen.....	29

1 Introduction

1.1 Background: Increased Adoption of Digital Learning and LMSs

An increasing digital literacy and adoption of digital learning has led to a great growth in the market for Learning Management Systems (LMS). As of September 2020, the number of LMS users is estimated at 73.8 million [1], and the global LMS market size is expected to grow from USD 13.4 billion in 2020 to USD 25.7 billion by 2025 [2].

This movement to a more technologically-assisted education has been further boosted by the onset of the Coronavirus (COVID-19) outbreak. To minimize social contact and reduce spread of the virus, social distancing policies have led many educational institutions to temporarily shut down. As of 2 April 2020, with 173 country-wide school closures, 84.8% of total enrolled learners (close to 1.5 billion learners) have been affected [3]. To maintain the continuity of learning, many schools have adopted online learning in place of face-to-face lessons. For instance, countries such as China, South Korea, Italy, and Iran have already shifted to temporary home-schooling via online educational tools and platforms within the first four months since the discovery of COVID-19 in December 2019 [4].

1.2 Benefits of LMSs

With the increased use of online learning, there is a need to evaluate the benefits of using LMSs in education.

1.2.1 Single Platform

LMSs provide teachers with a single platform for managing educational coursework online. They enable educators to make course material accessible online, including lesson content and assignments.

1.2.2 Online Assessment Functionalities

LMSs also offer online assessment functionalities, such as quizzes and tests. By providing a platform for quizzes, tests, assignments, and problems, LMSs encourage active learning by encouraging students to apply their knowledge and learning frequently.

Online assessment functionalities such as quizzes and tests also often have automatic grading options where the quizzes and tests can be graded and scored automatically by the system, removing the need for teachers to grade each assessment individually. This would be especially resource-efficient and timesaving for educators. Such automated grading can also instantly provide feedback to the students on their quiz answers and assignments. Automated grading functions are especially important for online learning. In online learning, students are

unable to get immediate feedback from teachers as often as in a traditional classroom. However, this can be overcome with the use of such automated grading functions, which can provide immediate feedback on students' learning.

1.2.2.1 Benefits of Feedback in Learning

In learning, feedback can be defined as “information given to students about their performance that guides future behavior” [5]. It is a response to a student's action that evaluates their current understandings and performance and aims to direct the student's attention to areas for growth and improvement. As such, feedback is essential in reducing the gap between a student's current understandings and performance and a goal, and thus, is essential to students' learning and growth.

In their book, Ambrose et al. [5] underscore the importance of feedback, coupled with opportunities for practice: “Goal-directed practice coupled with targeted feedback are critical to learning”. They further highlight the interconnection of feedback, practice, and performance in relation to overarching course goals [6].

According to Shute [7], effective feedback occurs throughout learning but not prior to learners' attempts to solve a problem, and immediate feedback is more efficient for complex content and for building long-term knowledge retention.

Giving instant feedback on an assignment serves to correct mistakes or affirm competence. Students are able to know of and learn from any mistakes they make, which makes them more likely to retain the knowledge. When learning is followed with immediate feedback, students can absorb or act on it while it is contextual to present learning and top of mind [8]. Giving feedback instantly as opposed to periodically makes learning a more active rather than passive experience as students will be more engaged in the learning.

In addition to its influence on achievement, feedback is also depicted as a significant factor in motivating learning [9]. In the workplace, Gallup estimates that employees who don't receive any feedback are 40% more likely to be disengaged [10]. This can also be applied in the context of learning, as students may become detached and uninterested without timely feedback as they may not know what areas to work on and how to proceed. On the other hand, timely and relevant feedback can trigger an intrinsic motivation to learn and engage students, thus conferring them the benefits of active learning. Furthermore, research has also suggested that students prefer immediate feedback over delayed feedback [11], thus providing such immediate feedback can help motivate students further.

1.2.2.2 Benefits of Active Learning

Active learning is defined as “any approach to instruction in which all students are asked to engage in the learning process” [12]. This is in comparison with more traditional modes of instructions such as large-scale lectures, where students passively receive knowledge from the lecturers.

Active learning fosters understanding rather than memorization. It “(shifts) the focus of instruction away from knowledge transmission to learners' knowledge construction through the creation of guided tasks, interactions, assignments, and environments that cultivate deep, meaningful learning” [13]. Through assignments and problems, active learning encourages students to apply their knowledge and learning in problem-solving. This can be especially useful in learning procedural and application-based skills such as programming.

Active learning encourages knowledge retention and increases student performance. Research has shown that “when students are actively involved in the learning task, they learn more than when they are passive recipients of instruction” [14]. Studies have also shown that “active learning leads to increases in examination performance that would raise average grades by a half a letter, and that failure rates under traditional lecturing increase by 55% over the rates observed under active learning” [15]. Active learning can also help in maintaining student concentration and deepening learning towards skills such as critical thinking.

2 Report Organization

This report aims to discuss the limitations of LMSs in the context of programming courses. The report is subdivided into nine sections. The following highlights the purpose of each section in the report:

Section 1: Introduction

This section provides an overview of the project by introducing LMSs and the benefits of LMSs.

Section 2: Report Organization

This section outlines how the report will be organized.

Section 3: LMSs for Programming Courses

This section discusses LMSs in the context of programming education and the limitations of such. This section outlines the problem and gaps in the current situation.

Section 4: Project Objective

Considering the current gaps as mentioned above, this section states the objective of the project.

Section 5: Requirements Analysis

This section discusses the requirements of the project's proposed solution.

Section 6: Technologies Used

This section focuses on the technologies used for this project.

Section 7: Implementation

This section discusses the system implemented and its functionalities.

Section 8: Future Work

This section discusses future extensions and improvements that can be carried out.

Section 9: Conclusion

This last section will thus provide a conclusion for this project.

3 LMSs for Programming Courses

3.1 Gaps in LMSs for Programming Education

Looking at the application of LMSs in the context of programming and coding education, there leaves more to be desired. In the market today, LMSs generally lack automated grading functions for programming and coding assignments. As such, one of the major benefits of using LMSs is not reaped. This results in 2 major problems:

1. Teachers need to spend time marking each assignment

Without automated grading functions, teachers are required to grade each assessment individually, which would not be efficient both timewise and energy-wise. By spending time marking each assignment, teachers are unable to use their time on more fruitful things that cannot be automated, such as coming up with lesson plans etc.

2. Students cannot get immediate feedback on their learning

This lack of automated grading functions means students must wait longer to receive feedback on their assignment. The benefits of receiving immediate feedback have been discussed in section 1.2.2.1 on the benefits of feedback in learning. Meanwhile, an automated

grading system can give students feedback on their learning through an almost immediate grade score. This will allow students to assess their own level of understanding and re-evaluate their learning without an unnecessary delay in feedback, aiding in their active learning.

3.2 External Automated Code Grading Tools

To circumvent these issues, educators can use external tools such as Web-CAT and CodeGrade which can automatically grade programming assignments.

However, such external tools cannot integrate smoothly into existing LMSs. According to I. Bouchrika [1], dissatisfaction with learning technology comes from the inability of LMSs to be integrated with other digital platforms. This means that, due to the difficulties in integration, in order to make use of such external tools, educators often have to use these tools and applications as a second platform to manage coding assignments. This diminishes a key benefit of using LMSs which is the ability for LMSs to act as a single platform to manage all coursework materials.

3.2.1 Case Study of Existing External Automated Code Grading Tool: Web-CAT

Furthermore, these external grading tools may also be limited in their grading capabilities. Each existing code grading tools have their own features and functions. This section aims to discuss the functionalities of code grading tools through a case study of Web-CAT.

Web-CAT is an advanced automated grading system that can grade students on how well they test their own code. It is free, open-source software implemented as a web application with a plug-in-style architecture [16].

Web-CAT offers multiple features, including assignment submission, automated feedback based on predefined test cases, hints to help to fix coding errors, and grade generation based on the test case report produced by JUnit-Reporter.

However, while using Web-CAT at Dickinson College in an introduction to programming course [17], the instructors often noticed undesirable difficulties from students' side. The students were unable to understand the feedback messages generated by the system about their submitted assignments (Figure 1).

Most of the errors were general and did not indicate the cause of the actual error. The same issue was also found at King Saud University, where 50.6% of the students were “unhappy using the system because the generated feedback report was unclear and does not explain the error type and also how the system generates their scores” [18].

Your Assignment Submission Results

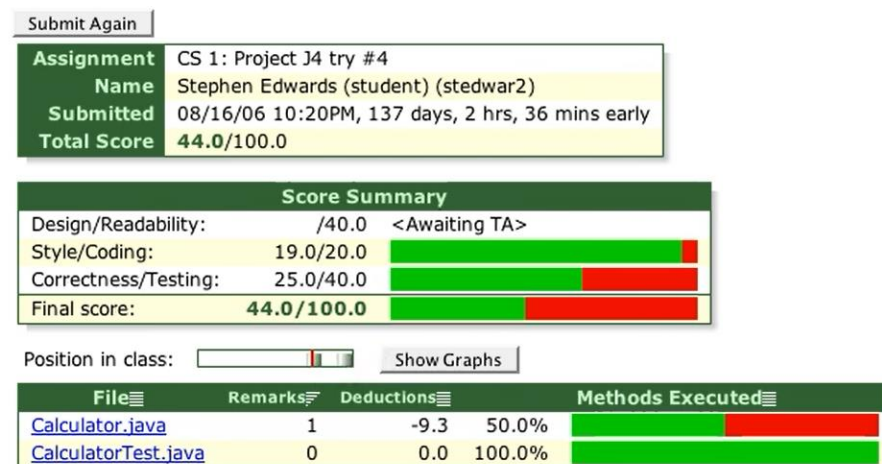


Figure 1: An example of a Web-CAT generated report. Screenshoted from [19]

4 Project Objective

This project aimed to solve the lack of an automatic grading system for coding assignments in LMSs as well as the limited capabilities of existing external automatic code-grading tools by implementing a code testing and grading feature and integrating it into an existing LMS program. This new feature will run student-submitted code against test cases to check for code correctness, to detect mastery of learning outcomes such as the correct use of coding techniques, and give students immediate feedback on their learning. This code grading tool will benefit both educators (by reducing the workload), and students (by providing immediate feedback to students in the process of learning).

5 Requirements Analysis

This section discusses the functional requirements of the system features.

Functional Requirements

- Educators:
 - System must allow educators to create assignments.
 - System must allow educators to enable code testing and grading functionality.
 - Where code testing functionality is enabled,
 - System must allow educators to enter visible test case inputs.
 - System must allow educators to enter hidden test case inputs.
 - System must allow educators to upload a correctly functioning code.

- System must allow educators to view student assignment submission.
- System must allow educators to download student assignments.
- System must show educators students' grade scores for submitted assignments.
 - Where code grading functionality has been enabled, system should show educators students' grade scores for submitted code assignments based on number of test cases passed.
- System must allow educators to manually grade assignment submissions.
- Students:
 - System must allow students to view/download assignments.
 - System must hide correctly functioning code from students' assignment view.
 - System must allow students to submit assignments.
 - Where code testing functionality has been enabled for students,
 - System must allow students to run test cases of inputs specified by the educator against their submitted code.
 - System must allow students to run test cases with their own inputs against their submitted code.
 - System must show students results for visible test cases.
 - System must not show details of hidden test cases to students.
 - System must show students their grade score for their submitted assignments.
 - Where code grading functionality has been enabled, system should immediately show students their grade scores for submitted code assignments based on number of test cases passed.

5.1 Use Case Diagram

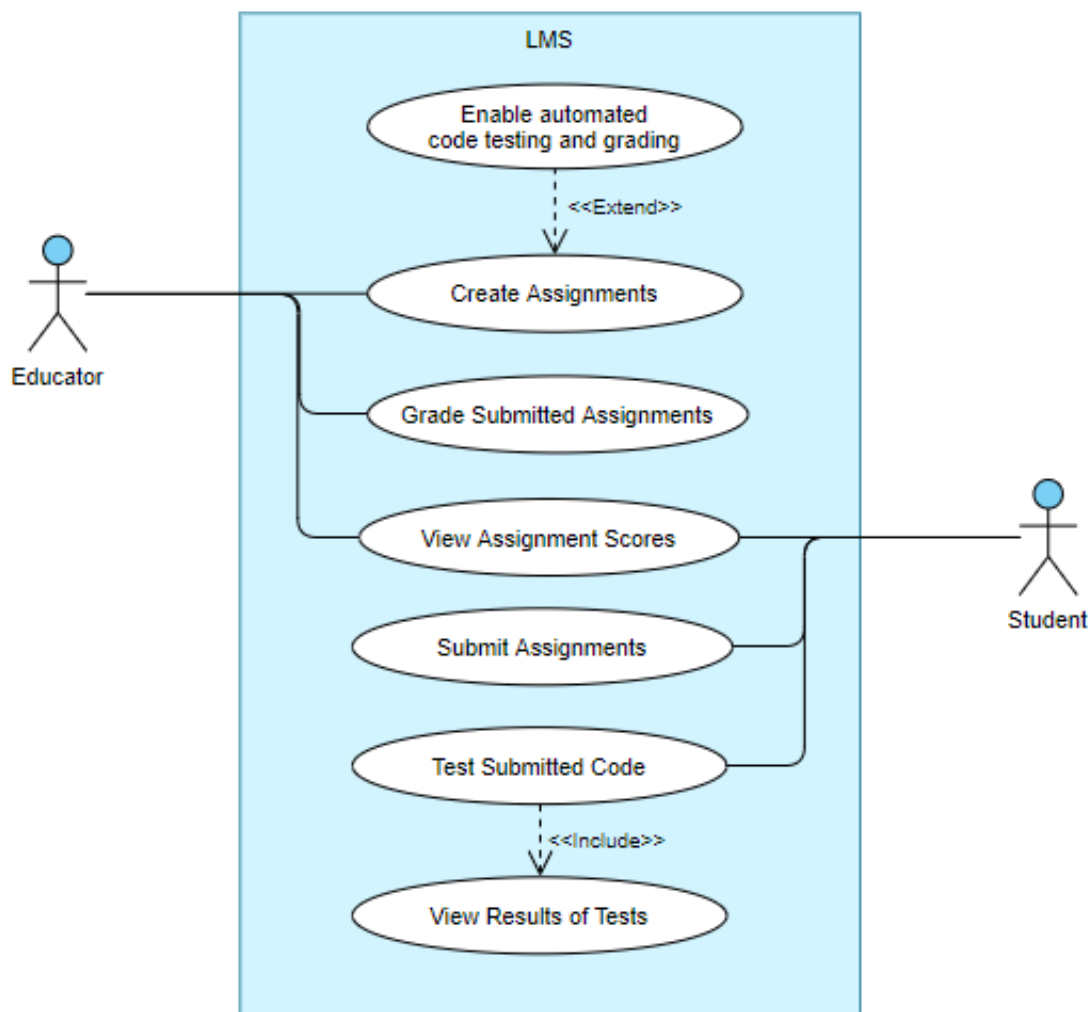


Figure 2: Use Case Diagram

6 Technologies Used

This section focuses on the technologies used for this project.

6.1 Front-end: Reactjs

Reactjs is an open-source front-end JavaScript library maintained by Facebook for building interactive and dynamic user interfaces (UIs) and UI components. In the case of this project, Reactjs has been chosen for front-end as there has been existing code in Reactjs, and it would have been counterintuitive to build on the existing project using other frameworks instead.

To further justify the choice of Reactjs, it is necessary to evaluate the benefits of using Reactjs. These include:

Component-based

React is component-based and allows developers to build encapsulated components which manage their own state, then compose them to make complex UIs. This allows for the building of high-quality, rich UIs.

ReactJS also provides reusable components that developers have the authority to reuse and create a new application with. As each component has its own logic and controls its own rendering, they can be reused wherever necessary. This isolation of functionality into components also improves code maintenance by enforcing low coupling. This ensures that modifications in one component would not dramatically affect other components. Such code re-use helps reduce development effort and makes the application maintenance easier.

Virtual Document Object Model (DOM)

React uses a virtual DOM, such that for every DOM object, there is a corresponding “virtual DOM object.” A virtual DOM object is a virtual representation of a DOM object and has the same properties as a real DOM object.

Any new view changes are first performed on the virtual DOM. From the changes made to the virtual DOM, the changes that need to be made to the real DOM is then identified and only these essential changes are applied to the real DOM [20]. This minimises update time to the real DOM, providing better user experience and higher performance by allowing faster rendering. This then enables developers to build highly dynamic UI, work with UI-objects faster, and apply changes in real-time.

Downward data flow

React follows downward data flow to ensure that the parent structure will not be affected by any modifications in its child structure [21]. To make changes in an object, the developer simply modifies its states and makes suitable changes, and only a specific component will be updated.

This data flow and structure makes code maintenance easier as changes made to child elements will not impact its parent structures. It also provides better code stability and smooth performance of the application.

6.2 Back-end

6.2.1 Node.js

Node.js is a back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. Node.js offers developers event-driven I/O APIs and asynchronous. It operates on a single threaded event-based loop to make all executions non-blocking.

In the case of this project, Node.js has been chosen for back-end as there has been existing code in Node.js, and it would have been counterintuitive to build on the existing project using other frameworks instead.

To further justify the choice of Node.js, it is necessary to evaluate the benefits of using Node.js. These include:

High Scalability

Node.js allows developers to leverage microservices that further allows segregation of the application into smaller parts. By breaking the application logic into smaller modules (microservices), instead of creating a single, large monolithic core, it enables better flexibility and increases ease of future growth [22]. As a result, in the case of additional features in the future, it would be easier to add more microservices than to integrate the additional features. Therefore, in planning for future extensions, Node.js would be a good choice as it enables high scalability.

Seamless JSON communication support

Node.js uses JavaScript and can use JavaScript Object Notation (JSON) format for communication without converting between binary models [22]. This is especially useful for this project where application programming interfaces (APIs) need to be built to communicate with the database.

6.2.2 MySQL

MySQL is an open-source relational database management system. In the case of this project, MySQL has been chosen for database management as the existing database models and tables are based in MySQL, and it would have been counterintuitive to not make use of the existing system and set up a new system.

To further justify the choice of MySQL, it is necessary to evaluate the benefits of using MySQL. These include:

Portability

MySQL is a cross-platform database server and can run on different platforms such as Linux, Windows etc. MySQL supports many platforms, with different languages like C, C++, PHP, PERL, JAVA, Python etc [23]. This would be useful for a web application such as that of this project.

Data Security

MySQL provides strong and robust data security to protect data, “including secure connections, authentication services, fine-grained authorization and controls, and data encryption” [24]. This makes MySQL a secure and reliable database management system, which would be beneficial in ensuring information such as assignments, quizzes, and their scores are secure.

On-Demand Scalability

MySQL offers “scalability to facilitate the management of deeply embedded apps” [25]. This would be useful in a LMS to allow the management of large amounts of information based on the large number of courses and students in a typical school.

High Performance

MySQL features a distinct storage-engine framework that allows system administrators to configure the MySQL database server. MySQL is designed to ensure optimum speed, using full-text indexes and unique memory caches for enhanced performance [25].

Round-the-Clock Uptime

MySQL also comes with the “assurance of 24×7 uptime and offers a wide range of high-availability solutions” [25]. This would be useful in this project to allow educators and students to create or submit assignments at any time convenient to them.

6.3 Code Testing/Analysis

6.3.1 Analysis of Code Testing/Analysis Methods

There are two types of software testing methods—static testing and dynamic testing. Static testing is software testing where the software application is tested without code execution, while under dynamic testing, the code is executed.

6.3.1.1 Static Code Testing/Analysis

Static code analysis examines an application’s source code before the program is run. This is usually done by analyzing the code against a given set of rules or coding standards. The main

objective of static testing is to “improve the quality of the software application by finding errors in early stages of software development process” [26].

Benefits

Static code analysis helps identify weaknesses in the code and ensure adherence to strict development standards, which will help reduce potential production issues. Static analysis may also be relatively cost-efficient as it is able to detect bugs at an early phase of the software development life cycle, compared to detecting such bugs later in the life cycle.

Limitations

Static code analysis shows little understanding of developer intent and is unable to expose overly complicated and subtle flaws or vulnerabilities. It does not find vulnerabilities introduced in the runtime environment. Automated static code analysis tools may not support all programming languages. It is also limited by the underlying rules and guidelines and are only as good as the rules it is using to scan with.

6.3.1.2 Dynamic Code Testing/Analysis

Dynamic code analysis examines an application during or after a program is run. Dynamic testing executes the software and validates the output with the expected outcome. It checks for the functional behaviour of the software system, memory and central processing unit (CPU) usage, and overall performance of the system. The main objective of dynamic testing is “to confirm that the software product works in conformance with the business requirements” [26].

Benefits

Dynamic code analysis “can (expose) a subtle flaw or vulnerability too complicated for static analysis alone to reveal” [27]. It identifies vulnerabilities in a runtime environment, and can be conducted against any application.

Limitations

Dynamic code analysis can only find defects in the part of the code that is actually executed. This means that if the code does not run, it does not get analyzed. Paths in the code that are not taken are also not analyzed.

6.3.2 Chosen Code Testing/Analysis Method and Rationale

Dynamic Code Testing was chosen as the method of code testing for this project.

The benefits of static code analysis are less applicable and thus less significant due to the context of this project.

For this project, the code analysis will be used in testing for the functionality of the code, and is not meant to check for adherence to coding standards. The functionality of the code can be easily tested by running the software and comparing the actual and expected outputs, as done in dynamic code testing. This can ensure that the code does what it is intended to.

As the assignments for which code testing is enabled are likely smaller-scale code and simple functions instead of large applications, dynamic code testing may be more suitable for the purpose of this project since cost of fixing the bug is not likely to be a big concern.

Furthermore, the limitation of static code analysis might be amplified in the context of this project.

Static code analysis is only as good as the underlying rules it uses, and generally does not support all programming languages. While it may be possible to update and enhance rules to support many programming languages for static code analysis, it may be more feasible to use dynamic code testing, which can be conducted against any application. This is important as the code assignments may require the use of different programming languages.

Additionally, the limitations of dynamic code analysis are less severe in the context of this project.

The most likely use case for the code testing tool is for students to test their completed and executable code assignments. Even if the code is unable to run, in the context of the assignment grading, it is a non-issue since it would not meet the requirements and the expected functionalities. The result of the dynamic code testing will be a failure (due to a failed validation of the output) and hence fails the assignment as expected.

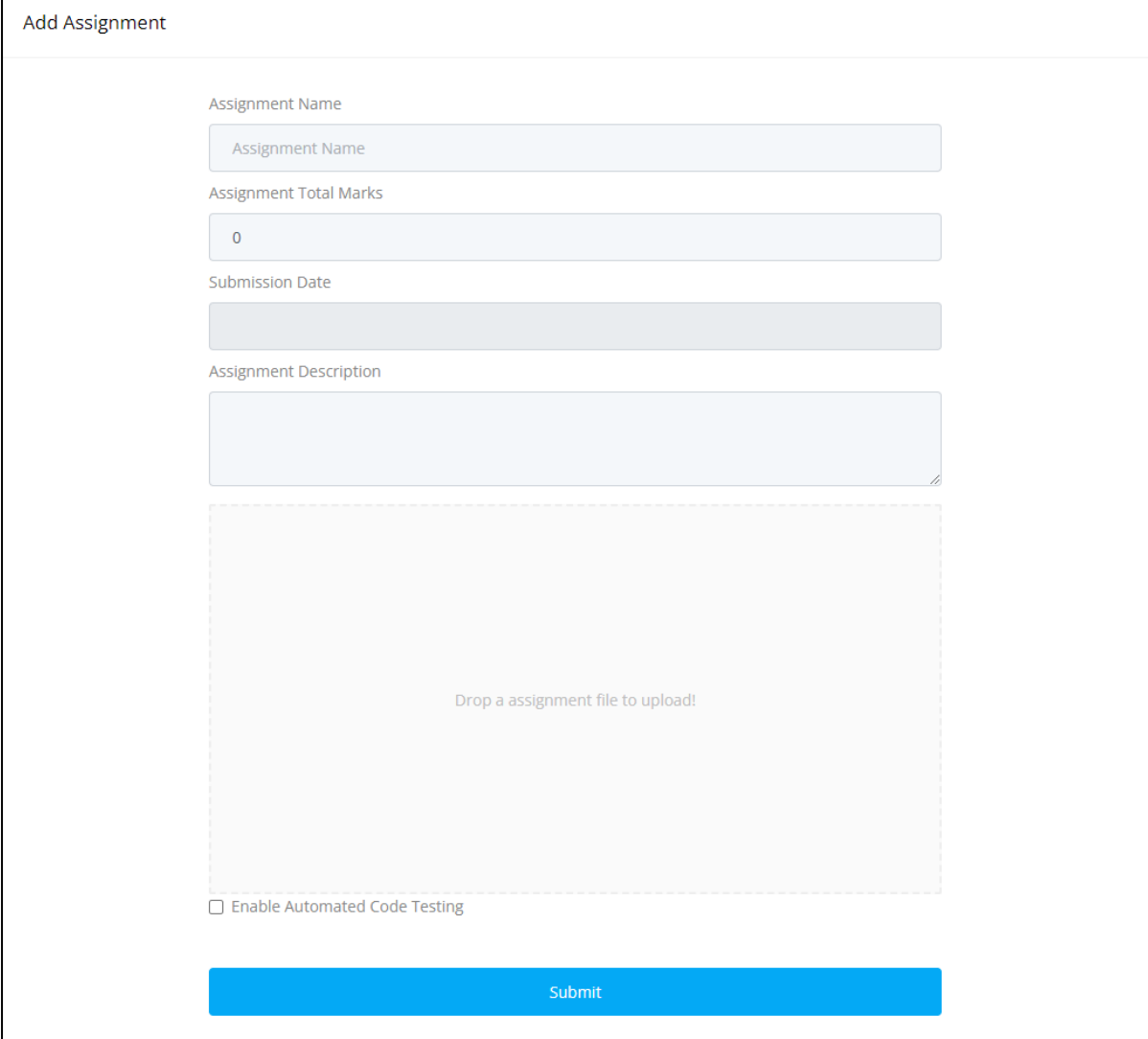
While limitations persist in that only the paths in the code that are executed are analysed in dynamic code analysis, such limitations can be overcome using more test cases to cover all the possible paths.

7 Implementation

This section discusses the system implemented and its functionalities.

7.1 Create and Submit Assignments

Educators are able to create new assignments (Figure 3) and choose whether to enable the automated code testing and grading for this new assignment.



The screenshot shows a web form titled "Add Assignment". It contains the following fields and elements:

- Assignment Name:** A text input field with the placeholder text "Assignment Name".
- Assignment Total Marks:** A text input field with the value "0".
- Submission Date:** A date picker field.
- Assignment Description:** A large text area for the description.
- File Upload:** A dashed border box containing the text "Drop a assignment file to upload!".
- Enable Automated Code Testing:** A checkbox.
- Submit:** A blue button at the bottom.

Figure 3: Screenshot of educator's Add Assignments screen

7.1.1 Logic

7.1.1.1 Create Assignment

To create an assignment, a form is used to accept inputs from the user/educator and files are accepted from the user through a drop zone. This information is then posted to the backend controller through an Application Programming Interface (API) post. The file is uploaded using Multer, a node.js middleware for handling form data and uploading files. The backend controller then builds a model instance and saves it to persist this instance in the database using Sequelize.

7.1.1.2 Submit Assignment

To submit an assignment, a drop zone is used to accept a file from the user/student. When students submit their assignment, information such as their user ID and the assignment ID is posted to the backend controller through an API post, while the submitted file is uploaded using Multer. The backend controller then builds a model instance and saves it to persist this instance in the database using Sequelize.

Where code testing is enabled for the assignment, students are able to test their code as well as receive an immediate score based on the number of test cases passed. Students are not required to test their code before submission and may also opt to directly submit their code. The system will detect if test cases have already been run and run the test cases to obtain the score if they have not been run. Where code testing is enabled, the score will be posted together with their user ID etc. during submission to the backend controller. The score is calculated as follows:

$$\text{Obtained Marks} = \frac{\text{Number of test cases passed}}{\text{Total number of test cases}} \times \text{Total marks for assignment}$$

Students are able to test their code with inputs specified by the educators that are saved into the state. The file is then uploaded with Multer and the list of inputs is posted to the backend controller. The backend controller then spawns the specified executable file directly as a new process, passing to the file the input. The stdout output of the child process is then collected and returned. The backend controller also writes the previously uploaded the correct code into a file and spawns this correct code executable file directly as another new child process, passing to this file the input. The output of this child process is also collected. The outputs of these two files are then compared and the result of this comparison and the two collected outputs are returned to the frontend, which then dynamically displays the data in a table format.

The same process occurs when students test their code with their own specified input. These two testings are separate and can be done individually, or together in any order.

7.1.2 Assignments with Code Testing Enabled

7.1.2.1 Educators' View

If educators choose to enable the automated code testing and grading, new fields will appear (Figure 4) to prompt them to enter inputs for test cases, as well as indicate whether each input should be visible to or hidden from students. Educators may add more than one input, as well as remove extra inputs (Figure 5). Educators will also be required to upload a file as a correctly functioning code.

The screenshot shows the 'Add Assignment' form. It includes fields for 'Assignment Name', 'Assignment Total Marks' (set to 0), 'Submission Date', and 'Assignment Description'. Below these is a large dashed box for uploading an assignment file. The 'Enable Automated Code Testing' checkbox is checked. Under the 'Input' section, there is a text input for 'Enter input test value', radio buttons for 'Visible' (selected) and 'Hidden', and an 'Add More Inputs' button. Another large dashed box is provided for uploading a 'correctly functioning code file'. A 'Submit' button is at the bottom.

Add Assignment

Assignment Name

Assignment Name

Assignment Total Marks

0

Submission Date

Assignment Description

Drop a assignment file to upload!

☒ Enable Automated Code Testing

Input

Enter input test value

☒ Visible ☐ Hidden

Add More Inputs

Drop a correctly functioning code file to upload!

Submit

Figure 4: Screenshot of educator's Add Assignments screen when automated code testing is enabled

To illustrate the functionality of the system, an assignment with the following fields (Figure 5) will be used as an example.

Add Assignment

Assignment Name

Assignment 1

Assignment Total Marks

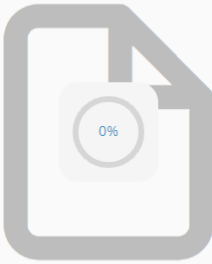
50

Submission Date

11/12/2021 12:00 AM

Assignment Description

Assignment 1, code testing enabled



☒ Enable Automated Code Testing

Input

1

☒ Visible ☐ Hidden

Remove

Input

2

☒ Visible ☐ Hidden

Remove

Input

3

☒ Visible ☐ Hidden

Remove

Input

4

☐ Visible ☒ Hidden

Remove

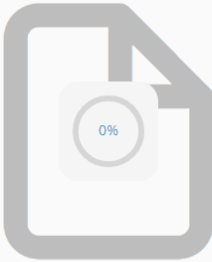
Input

5

☐ Visible ☒ Hidden

Remove

Add More Inputs



Submit

Figure 5: Screenshot of example assignment created with code testing enabled

For the correctly functioning code file, the following code (Figure 6), which demonstrates a simple function to add one to the input, was used.

```
import sys

def plusOne(x):
    return(x+1)

if __name__ == '__main__':
    try:
        inp = int(sys.argv[1])
        result = plusOne(inp)
        print(result)
    except Exception as e:
        print(e)
```

Figure 6: Screenshot of code used as correct code

7.1.2.2 Students' View

When submitting for an assignment with code testing enabled, students given the option to test their code against inputs submitted by the teacher as well as the option to test their code against their own input (Figure 7).

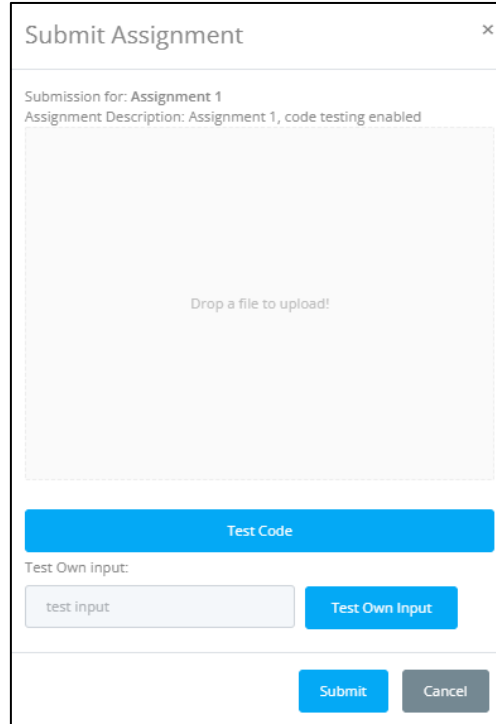


Figure 7: Screenshot of student's Submit Assignment screen when automated code testing is enabled

As an example, a code file with the following code (Figure 8) will be used as the student's uploaded assignment attempt. It should be noted that for inputs of "2" and "4", the student's submitted code will return the wrong answers. It can also be noted that the only part of the code that changes is the body of the function `plusOne(x)`, meaning that educators may provide students with a boilerplate code which handles any inputs, and students can focus on the functionality of the code.

```
import sys

def plusOne(x):
    if x==2:
        return(22)
    if x==4:
        return(100)
    return(x+1)

if __name__ == '__main__':
    try:
        inp = int(sys.argv[1])
        result = plusOne(inp)
        print(result)
    except Exception as e:
        print(e)
```


Figure 8: Screenshot of example code submitted by student

After uploading the above code in a file, students may click on the “Test Code” button to test their submitted code against inputs submitted by the educator. The system will then display a table of results of the test cases (Figure 9). The table states the input, expected output, actual output, and results of the test case. The table row would appear in light red for failed test cases, and in light green for passed test cases. For hidden test cases, the input, expected output, and actual output will be hidden from students.

Submit Assignment

Submission for: Assignment 1

Assignment Description: Assignment 1, code testing enabled



Test Code

Input	Visibility	Expected Output	Actual Output	Results
1	visible	2	2	Passed
2	visible	3	22	Failed
3	visible	4	4	Passed
-hidden-	hidden	-hidden-	-hidden-	Failed
-hidden-	hidden	-hidden-	-hidden-	Passed

Test Own Input:

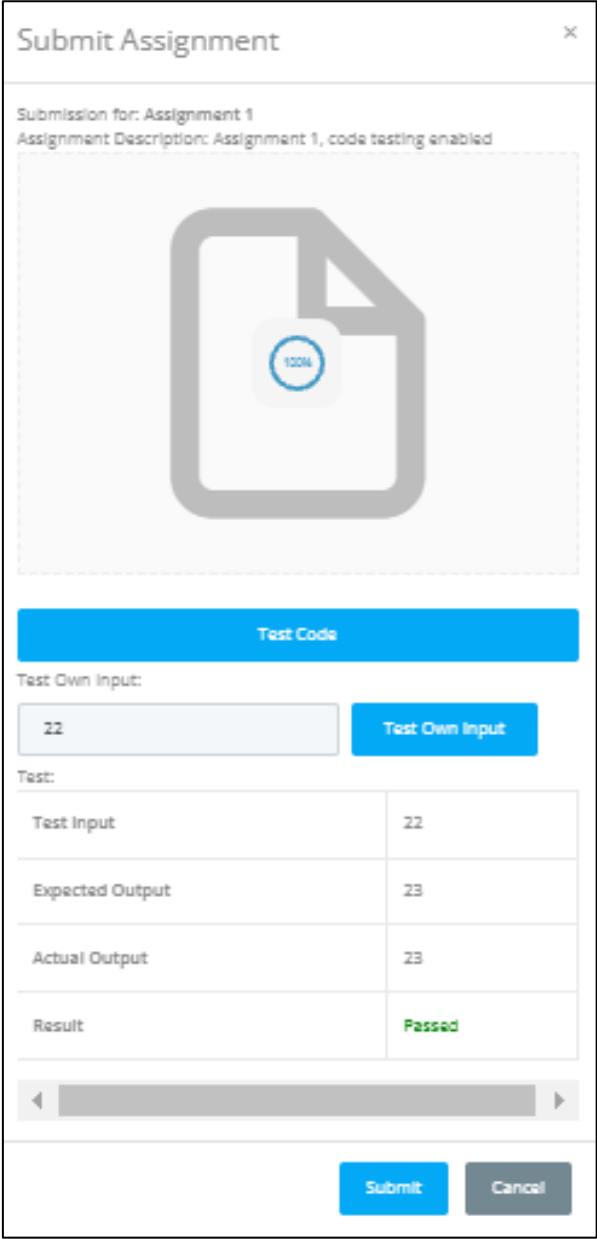
Test Own Input

Submit

Cancel

Figure 9: Screenshot of student's screen after testing code

Students may also test the submitted code against their own input using the “Test Own Input” button. Figure 10 shows the view if a student tests their own input without running test cases specified by the teacher. Similar information (input, expected output, actual output, results) are displayed.



Submit Assignment

Submission for: Assignment 1
Assignment Description: Assignment 1, code testing enabled

100%

Test Code

Test Own Input:

22 Test Own Input

Test:

Test Input	22
Expected Output	23
Actual Output	23
Result	Passed

Submit Cancel

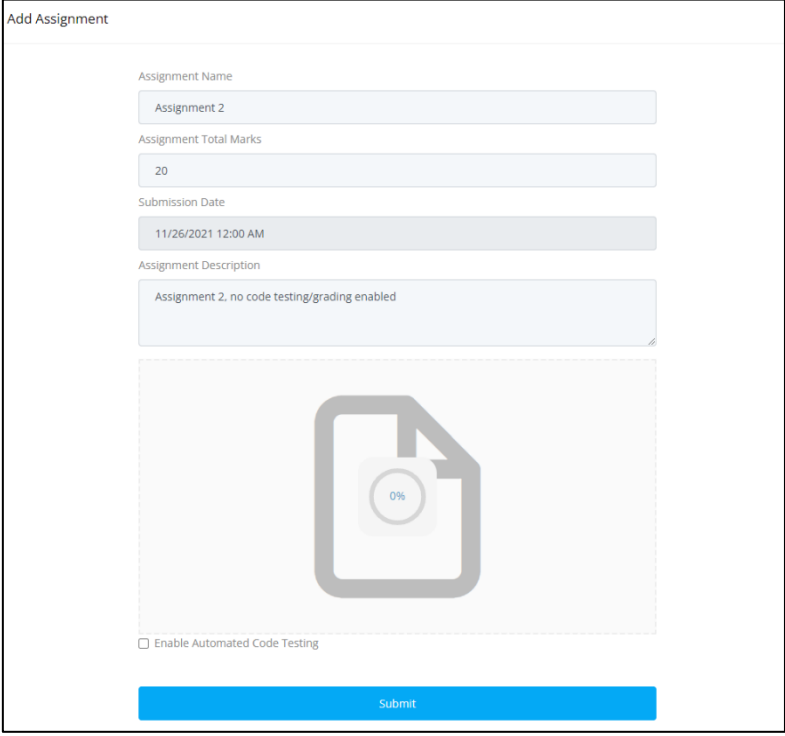
Figure 10: Screenshot of student's screen after testing code against own input

Students can click on the “Submit” button to submit their assignment. This can be done with or without first testing the code. The system will calculate the obtained marks based on the number of test cases passed and the obtained marks may be viewed immediately after submission.

7.1.3 Assignments without Code Testing Enabled

7.1.3.1 Educators' View

In the case of non-coding assignments, educators may also create assignments without enabling code testing. In this case no additional fields would appear (Figure 11).



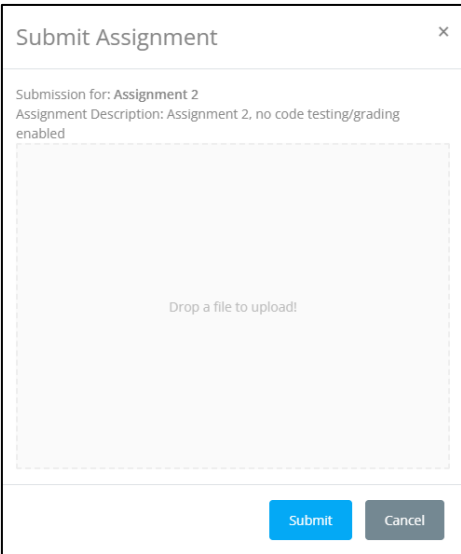
The screenshot shows the 'Add Assignment' form. It includes the following fields and controls:

- Assignment Name:** A text input field containing 'Assignment 2'.
- Assignment Total Marks:** A text input field containing '20'.
- Submission Date:** A date and time picker showing '11/26/2021 12:00 AM'.
- Assignment Description:** A text area containing 'Assignment 2, no code testing/grading enabled'.
- File Upload:** A large dashed box containing a file icon and a '0%' progress indicator.
- Enable Automated Code Testing:** A checkbox that is currently unchecked.
- Submit:** A blue button at the bottom right.

Figure 11: Screenshot of educator's Add Assignments screen without automated code testing

7.1.3.2 Students' View

When submitting for an assignment without code testing enabled, students are not given any option for testing and may only submit their assignment (Figure 12).



The screenshot shows the 'Submit Assignment' dialog. It includes the following elements:

- Submission for:** Assignment 2
- Assignment Description:** Assignment 2, no code testing/grading enabled
- File Upload:** A large dashed box containing the text 'Drop a file to upload!'.
- Submit:** A blue button.
- Cancel:** A grey button.

Figure 12: Screenshot of student's Submit Assignment screen without automated code testing

7.2 View Assignments and Submission Scores

7.2.1 Educators' View

Educators are able to view a list of created assignments (Figure 13).

Assignments List			
Title	Total Marks	Submission Date	Actions
Assignment 1	50	Fri Nov 12 2021 00:00:00 GMT+0800 (Singapore Standard Time)	<div>Download</div> <div>Check Submissions</div> <div>Delete</div>
Assignment 2	20	Fri Nov 26 2021 00:00:00 GMT+0800 (Singapore Standard Time)	<div>Download</div> <div>Check Submissions</div> <div>Delete</div>

Figure 13: Screenshot of educator's List Assignments screen

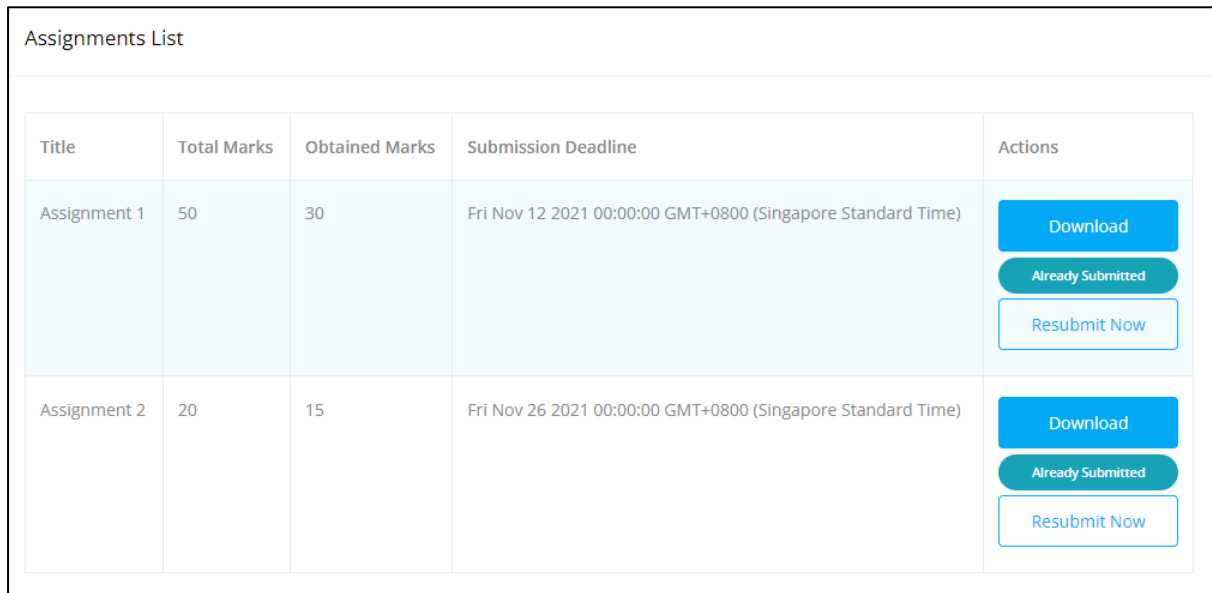
They are also able to download students' submitted assignments and obtained marks (Figure 14). Educators are also able to manually grade them. The scores manually given by an educator is able to overwrite the scores automatically given through the automated code grading.

Submitted Assignments for Assignment 1					
First Name	Last Name	U ID	Total Marks	Obtained Marks	Submission
student	soh	U1821631B	50	30	<div>Download</div> <div>Add Marks</div>

Figure 14: Screenshot of educator's List Submitted Assignments screen

7.2.2 Students' View

Students are able to view a list of assignments and the marks obtained for each (Figure 15). Where the automated code grading is enabled, the marks for that assignment would update and display immediately after submission.

The screenshot shows a web interface titled "Assignments List". It contains a table with two rows of assignment data. Each row has five columns: Title, Total Marks, Obtained Marks, Submission Deadline, and Actions. The first row is for "Assignment 1" with a total of 50 marks and 30 obtained marks, with a deadline of Fri Nov 12 2021 00:00:00 GMT+0800 (Singapore Standard Time). The second row is for "Assignment 2" with a total of 20 marks and 15 obtained marks, with a deadline of Fri Nov 26 2021 00:00:00 GMT+0800 (Singapore Standard Time). The Actions column for each row contains three buttons: "Download" (blue), "Already Submitted" (teal), and "Resubmit Now" (light blue).

Title	Total Marks	Obtained Marks	Submission Deadline	Actions
Assignment 1	50	30	Fri Nov 12 2021 00:00:00 GMT+0800 (Singapore Standard Time)	<div>Download</div> <div>Already Submitted</div> <div>Resubmit Now</div>
Assignment 2	20	15	Fri Nov 26 2021 00:00:00 GMT+0800 (Singapore Standard Time)	<div>Download</div> <div>Already Submitted</div> <div>Resubmit Now</div>

Figure 15: Screenshot of student's List Assignments screen

8 Future Work

This section discusses future extensions and improvements that can be carried out. It has to be acknowledged that this system is far from perfect and that there is room for improvements. The following are suggestions for future improvements and extensions:

Support for different programming languages

The implemented system currently only support python code submissions and testing. This can be extended in the future to cover different programming languages. The current implementation spawns a child process to run the submitted code file. To support different languages, an additional function to detect the file type and spawn the respective child process can simply be added without having to change the way the test cases' input and output values are stored and displayed.

Integrated code editor

The implemented system currently accepts submissions and runs the test cases on them. In order to make this system more comprehensive, a code editor can be integrated such that students may directly code on the LMS instead of coding on a separate environment and only

submitting the code in the LMS. A code editor can help increase convenience for students who may have to test and edit their code multiple times.

Optimization

As the current system spawns a child process for each input value, this can cause large latency if there are many test cases, or if many students attempt to test their code at the same time. To account for the likely large number of students in each course, this system will need to be optimised to support the scale.

9 Conclusion

In conclusion, this project tackled the lack of an automatic grading system for coding assignments in LMSs as well as the limited capabilities of existing external automatic code-grading tools by implementing a code testing and grading feature and integrating it into an existing LMS program. This provides two key benefits—it allows programming students to receive immediate feedback on their online code submissions, thus aiding in their learning, and also saves the time and effort of educators. As the world increasingly adopts online learning, with this feature, programming students and educators will also be able to reap the full benefits of an LMS in their learning and teaching experience.

10 References

- [1] I. Bouchrika. "51 LMS Statistics: 2019/2020 Data, Trends & Predictions." Guide2Research. <https://www.guide2research.com/research/lms-statistics> (accessed 23 February, 2021).
- [2] "LMS Market by Component, Delivery Mode, Deployment Type, User Type, and Region - Global Forecast to 2025," in *Plus Company Updates*, ed, 2020, p. NA.
- [3] "Education: From disruption to recovery." UNESCO. <https://en.unesco.org/covid19/educationresponse> (accessed 23 February, 2021).
- [4] G. Tam and D. El-Azar. "3 ways the coronavirus pandemic could reshape education." World Economic Forum. <https://www.weforum.org/agenda/2020/03/3-ways-coronavirus-is-reshaping-education-and-what-changes-might-be-here-to-stay> (accessed 23 February, 2021).
- [5] S. Ambrose, M. Bridges, M. DiPietro, M. Lovett and M. Norman, *How Learning Works: Seven Research-Based Principles for Smart Teaching*, 1st ed. John Wiley & Sons, Incorporated, 2010, p. 125.
- [6] "Feedback for Learning", Ctl.columbia.edu. [Online]. Available: <https://ctl.columbia.edu/resources-and-technology/resources/feedback-for-learning/>. [Accessed: 16- Oct- 2021].
- [7] V. Shute, "Focus on Formative Feedback", *Review of Educational Research*, vol. 78, no. 1, pp. 153-189, 2008. Available: 10.3102/0034654307313795.
- [8] I. Markovic, "Why Giving Instant Feedback is Important for Effective Learning | EduMe", EduMe. [Online]. Available: <https://edume.com/blog/role-of-feedback-in-improving-learning>. [Accessed: 16- Oct- 2021].
- [9] S. Narciss and K. Huth, "How to design informative tutoring feedback for multimedia learning", H. M. Niegemann, D. Leutner, & R. Brunken (Ed.), *Instructional design for multimedia learning*, pp. 181–195, 2004. [Accessed 16 October 2021].
- [10] "Employee Feedback: Why It Matters [INFOGRAPHIC]", Blue Beyond Consulting. [Online]. Available: <https://www.bluebeyondconsulting.com/thought-leadership/employee-feedback-why-it-matters/>. [Accessed: 16- Oct- 2021].
- [11] M. Epstein and G. Brosvic, "Students Prefer the Immediate Feedback Assessment Technique", *Psychological Reports*, vol. 90, no. 32, pp. 1136-1138, 2002. Available: 10.1177/003329410209000315.2.

- [12] "Active Learning | Center for Educational Innovation", Cei.umn.edu. [Online]. Available: <https://cei.umn.edu/active-learning>. [Accessed: 16- Oct- 2021].
- [13] "Active Learning Activities | Centre for Teaching Excellence", Centre for Teaching Excellence. [Online]. Available: <https://uwaterloo.ca/centre-for-teaching-excellence/teaching-resources/teaching-tips/developing-assignments/assignment-design/active-learning-activities>. [Accessed: 16- Oct- 2021].
- [14] P. Cross, "Teaching for Learning", American Association for Higher Education Bulletin, 1987.
- [15] S. Freeman et al., "Active learning increases student performance in science, engineering, and mathematics", Proceedings of the National Academy of Sciences - PNAS, 2014.
- [16] "Web-CAT - Web-CAT", Web-cat.github.io. [Online]. Available: <https://web-cat.github.io/projects/Web-CAT/>. [Accessed: 16- Oct- 2021].
- [17] G. Braught and J. Midkiff, "Tool Design and Student Testing Behavior in an Introductory Java Course," in Proceedings of the 47th ACM Technical Symposium on Computing Science Education, New York, NY, USA, 2016, pp. 449–454.
- [18] H. Aldriye, A. AlKhalaf, and M. Alkhalaf, "Automated Grading Systems for Programming Assignments: A Literature Review," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 3, 2019, doi: 10.14569/IJACSA.2019.0100328.
- [19] "Web-CAT Submission Walkthrough - Web-CAT", Web-cat.github.io. [Online]. Available: <https://web-cat.github.io/projects/Web-CAT/SubmissionWalkthrough.html>. [Accessed: 16- Oct- 2021].
- [20] J. Willoughby, "The Top 5 Benefits of React that Make Life Better", Telerik Blogs, 2017. [Online]. Available: <https://www.telerik.com/blogs/5-benefits-of-reactjs-to-brighten-a-cloudy-day>. [Accessed: 16- Oct- 2021].
- [21] U. Pisuwala, "The benefits of ReactJS and reasons to choose it for your project", Peerbits. [Online]. Available: <https://www.peerbits.com/blog/reasons-to-choose-reactjs-for-your-web-development-project.html>. [Accessed: 16- Oct- 2021].
- [22] "The Good and the Bad of Node.js Web App Development", AltexSoft, 2019. [Online]. Available: <https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-node-js-web-app-development/>. [Accessed: 16- Oct- 2021].

- [23] "MySQL Advantages and Disadvantages - techstrikers.com", Techstrikers.com. [Online]. Available: <https://www.techstrikers.com/MySQL/advantages-and-disadvantages-of-mysql.php>. [Accessed: 16- Oct- 2021].
- [24] "MySQL :: MySQL Security Best Practices", Mysql.com. [Online]. Available: <https://www.mysql.com/why-mysql/presentations/mysql-security-best-practices/>. [Accessed: 16- Oct- 2021].
- [25] M. McLean, "8 Advantages of Using MySQL", DevOps.com, 2017. [Online]. Available: <https://devops.com/8-advantages-using-mysql/>. [Accessed: 16- Oct- 2021].
- [26] T. Hamilton, "Static Testing vs Dynamic Testing: What's the Difference?", Guru99, 2021. [Online]. Available: <https://www.guru99.com/static-dynamic-testing.html>. [Accessed: 16- Oct- 2021].
- [27] N. DuPaul, "Static Testing vs. Dynamic Testing | Veracode", Veracode, 2019. [Online]. Available: <https://www.veracode.com/blog/secure-development/static-testing-vs-dynamic-testing>. [Accessed: 16- Oct- 2021].