Jurnal Teknik Informatika vol 18 no 4 October-December 2023, pp. 191-200
p-ISSN : 2301-8364, e-ISSN : 2685-6131 , available at : https://ejournal.unsrat.ac.id/index.php/informatika

Selenium Based Turnitin Automation For Plagiarisme Checker Service's Efficiency In Higher Education Academic Portals

Otomatisasi Penggunaan Turnitin Berbasis Selenium Untuk Efisiensi Layanan Cek Plagiarisme Di Portal Akademik Perguruan Tinggi

Eliza J. Sumampouw, Oktavian A. Lantang, Alwin M. Sambul
Dept. of Electrical Engineering, Sam Ratulangi University Manado, Kampus Bahu St., 95115, Indonesia
e-mails : elizasumampouw08@gmail.com, oktavian_lantang@unsrat.ac.id, asambul@unsrat.ac.id
Received: 18 September 2023; revised: 24 October 2023; accepted: 24 October 2023

***Abstract* — Plagiarism is a problem generally seen in the academic world. Many members of the academic community failed to defend its integrity, and opts to take some or all the writing of other people. Prevention can be made by controlling how far one's document is similar with other publication. Turnitin service can be used to check a document's similarity with other online publications. Turnitin's usage can be automated with Selenium, a browser automation tool that can do navigation actions on a website automatically, according to a programmed action flow.A REST API can be used as the data communication gateway between users and Turnitin's automation. With Turnitin automation using Selenium, and and API that connects the automation and users, the plagiarism checker service was successfully built and integrated into the academic portal and be easily used by users. Users can check their document through INSPIRE academic portal, and INSPIRE itself utilizes Turnitin automation to check the document for any plagiarism indication. The performance was found to be able to handle a large number of requests, so that service access will not be significantly disrupted**
***Key words*—Automation; Academic Portal; Plagiarism; Selenium**

***Abstrak* — Plagiarisme merupakan masalah yang umum ditemukan dalam dunia akademik. Banyak civitas akademika yang tidak mampu mempertahankan integritasnya, sehingga memutuskan untuk mengambil, sebagian atau seluruh hasil karya orang lain sebagai miliknya sendiri. Pencegahan dapat dilakukan dengan mengontrol kemiripan dokumen penulis dengan publikasi yang ada. Layanan Turnitin dapat digunakan untuk mengecek kemiripan dokumen dengan publikasi online. Pemanfaatan layanan Turnitin dapat dilakukan dengan menggunakan Selenium, sebuah alat otomatisasi browser yang bisa melakukan aksi-aksi navigasi sebuah website secara otomatis, berdasarkan alur perintah yang diprogramkan. Sebuah REST API bisa digunakan sebagai gerbang komunikasi data antara pengguna dan otomatisasi Turnitin. Dengan berbasis otomatisasi Turnitin menggunakan Selenium, dan API yang menghubungkan otomatisasi dan pengguna, layanan cek plagiarisme berhasil dibangun dan diintegrasikan ke dalam portal akademik dan digunakan dengan mudah oleh pengguna. Pengguna bisa melakukan pengecekan plagiarisme melalui portal akademik INSPIRE, dan INSPIRE sendiri memanfaatkan otomatisasi Turnitin untuk melakukan pengecekan dokumen. Performa dari layanan didapati mampu menangani jumlah permintaan yang besar, sehingga layanan tidak akan terganggu secara signifikan**
***Kata kunci* — Otomatisasi; Plagiarisme; Portal Akademik; Selenium**

I. INTRODUCTION

A. Background

Before technology developed as rapidly as it does now, plagiarism was a problem that was difficult to handle. In proving whether a work is plagiarized, extensive research is required on the related work which must be carried out manually. However, with intensive efforts to digitize media, especially research documents, data sets that must be processed one by one can be processed quickly and easily by computers. By comparing one document with millions of documents on the internet, it is possible to detect and prevent plagiarism efficiently [1].

Plagiarisme is a problem that has haunted the academic world for a long time. For various reasons, an academic decides to take part or all of someone else's work without providing appropriate citation. Inappropriate citations could be due to academic negligence [2]. Therefore, education and supervision are needed, both from academics and the academic community to monitor if there are indications of plagiarism from someone. [3], [4].

The easiest indicator of plagiarism is the similarity of the writing to existing publications. Similarities will be easily seen when the writer takes someone else's writing without any further modifications. The percentage of similarity of a document can be an indication of plagiarism. However, the results of similarities in writing detected by a program must be accompanied by further assessment from experts to conclude whether the writing being checked is the result of plagiarism or not. [3].

Various efforts have been made to combat plagiarism in higher education. Several universities are making outreach and educational efforts regarding how a written work should be structured, there are also exploring the development of a plagiarism detection application using existing algorithms to check the similarity of a document based on existing repositories, as well as the use of third-party document similarity checking services, such as Turnitin [2], [3].

Turnitin accounts are owned by each faculty's study program at Sam Ratulangi University. However, Turnitin's account

operator limitations meant that these accounts were not used effectively. One solution so that students can access Turnitin services more effectively is to connect Turnitin services to the academic portal. Thus, a plagiarism checking service using Turnitin is proposed to be integrated into the Sam Ratulangi University academic portal, namely the INSPIRE Portal. With this feature, all parties, both students and lecturers, can freely check the similarity of their thesis or other written work with previously existing publications.

To integrate Turnitin services into a platform, Turnitin actually provides their API access to be accessed by other platforms. However, this access is charged by Turnitin based on the number of existing platform accesses, which in the case of the INSPIRE Portal will be very expensive. An alternative that can be taken to overcome this problem is to automate the use of the Turnitin website and empower existing accounts. This automation will act as a Turnitin user and check document similarity and retrieve the results of the check.

A website can be automated using a browser automation tool called Selenium. Selenium is a browser automation tool that is generally used for testing a website, but can also be used for user-desired needs. With Selenium, a website that should be navigated by a human user can be replaced with a program that runs automatically.

B. Related Works

Research by Warouw et al [5] discusses the integration of a third-party plagiarism checking service, namely Unicheck, into the INSPIRE academic portal by utilizing the API provided by Unicheck.

Noor Kamala Sari built an e-journal plagiarism checking website for the medical faculty at Palangkaraya Raya University by utilizing a text similarity algorithm called Rabin-Karp [6].

Mcphahlele and McKenna conducted a study on the use of Turnitin services in higher education, starting from general perceptions regarding handling plagiarism, to how Turnitin services should be used to prevent plagiarism [7].

Cahyo Manunggal and Christiani raised a case study of using Turnitin to detect plagiarism [2]. This research discusses how Turnitin is done manually by campus librarians to process student submissions.

Umar and Zhanfang conducted a study on automated software testing [8]. This study discusses a number of tools and frameworks that are generally used to test software automatically, with one of the tools discussed being Selenium.

Supriyadi and Adhari Adiguna designed an Android-based Key Performance Indicator application using React Native and Backend API [9]. The application is designed using the Rapid Application Development (RAD) method to shorten application processing time.

C. Automation

According to KBBI, automation is "the replacement of human power with machine power which automatically carries out and organizes work so that it no longer requires human supervision"[10]. According to Dorf, automation is a technology in which a process or procedure can be achieved through programmed instructions, and is usually combined with automatic reciprocal control (feedback) to ensure the appropriate execution of instructions [11]. Automation is usually used in the context of work in factories, which requires massive material processing and is difficult to do by humans. However, based on the existing definition, automation can include all work or processes that run by themselves without human intervention, based on predetermined instructions.

In the context of software engineering, automation is usually carried out for the purposes of testing software [8]. Automated testing is a process that uses proprietary software, separate from the software to be tested, to control test execution and compare expected and emergent results. Automated software testing generally saves time, and testers can also run a large number of tests in a low time span, so that repetitive and time-consuming tasks can be automated. Although more commonly used for testing, software automation tools can also be used to automate repetitive activities of other applications. [12]. Processes that can take a long time can be done by a computer in seconds, and can be repeated over and over again.

D. Selenium

Selenium is an open-source project that includes a set of tools and libraries aimed at supporting web browser automation [13].

The main component of Selenium is Selenium WebDriver, an interface for writing automation instructions that can be used for any browser. Selenium WebDriver accepts existing commands and sends them to the browser. This process is implemented through a driver specific to a particular browser, which will send commands to the browser and receive the output results of these commands. Most browser drivers access and run browser applications directly during the automation process (such as Firefox, Google Chrome, Internet Explorer, Safari, or Microsoft Edge), but there are also browser drivers that only simulate a browser. Selenium WebDriver is fully implemented and can be used in JavaScript, Ruby, Java, Kotlin, C#, and Python [14].

E. Plagiarism

According to the KBBI, plagiarism is "taking someone else's writing (opinions and so on) and making it appear as if it were your own writing (opinions and so on), for example publishing someone else's written work in one's own name; plagiarize." Meanwhile, quoted in the Regulation of the Minister of Education of the Republic of Indonesia no. 17 of 2010, plagiarism is "the act of intentionally or unintentionally in obtaining or trying to obtain credit or value for a scientific work, by quoting part or all of another party's work and/or scientific work which is recognized as his/her scientific work, without properly stating the source and adequate" [14]. It can be said that plagiarism is an act of stealing intellectual results that are not one's own, either intentionally or unintentionally, and presenting these results as one's own to the public.

F. INSPIRE Portal

The INSPIRE portal is an integrated application owned by Sam Ratulangi University which functions to accommodate

p-ISSN : 2301-8364, e-ISSN : 2685-6131 , available at : https://ejournal.unsrat.ac.id/index.php/informatika

Unsrat lecture activities. The INSPIRE portal provides various services to support ongoing academic implementation and lecture administration. The contents of the services provided by the INSPIRE portal are divided based on the user's position, whether as a student, lecturer or other party [15], [16].

II. Development Method

In developing plagiarism checking services, the development method chosen was Rapid Application Development (RAD), an incremental software development method to shorten processing time [17]. This method emphasizes speed in developing an application, where users or stakeholders provide feedback during the development process and it can be implemented immediately. The 5 stages of RAD are Analysis & and Quick Design, Prototype Cycles, Testing, and Deployment.

A. Analysis & Quick Design

This stage is carried out to study the business processes related to the plagiarism checking service that will be created. From the existing business processes, we then see what functionality can be developed based on the available business processes.

First of all, the Turnitin business process is described to find out how Turnitin works, then an automation flow is built which is carried out by Selenium. These processes are the process of uploading documents to Turnitin, the process of downloading documents from Turnitin, as well as the process of creating new tasks to accommodate a certain format. These processes are described in the activity diagram in Figure 2-4

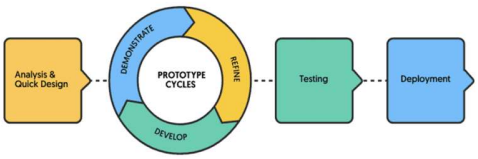

Figure 1. RAD stages

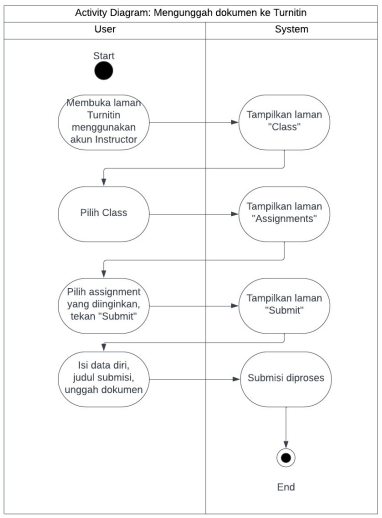

Figure 2. Activity diagram of the document upload process in Turnitin

After analyzing the INSPIRE portal business process, it was then determined how the plagiarism checking service could be integrated into the INSPIRE portal. The features that will be integrated are the Check Plagiarism feature itself so that users can make independent submissions, integration of the plagiarism check service into other INSPIRE features such as thesis guidance, as well as creating formats as templates for checking formats. These processes are also described in the activity diagram in Figure 5-7.

Based on these processes, the plagiarism checking process in INSPIRE can be described in the flowchart in Figure 8.

For data communication between users and plagiarism check automation, an API will be built to control user input with automation, so the API here acts as a “gateway” between the user and the automation. The API can also be used to integrate the plagiarism check feature with other features in INSPIRE. All business processes from Turnitin are controlled by automation, so users don't need to know the behind-the-scenes details of this feature. An illustration for this scheme is depicted in figure 9.

B. Prototype Cycles

This development stage has the character of a cycle, where this stage will continue to be repeated until the client is satisfied with the program being developed. In general, this stage requires the developer to create a prototype that will be validated by the client. However, because this research is a back-end development in nature, what will be discussed here is what functionality has been completed, and whether the compiled data is appropriate or still needs to be adjusted to the data in INSPIRE. This process continues until all functionality has been implemented

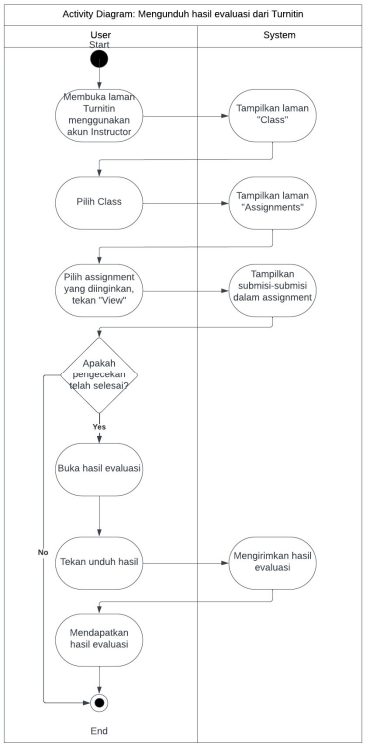

Figure 3. Activity diagram of the document download process in Turnitin

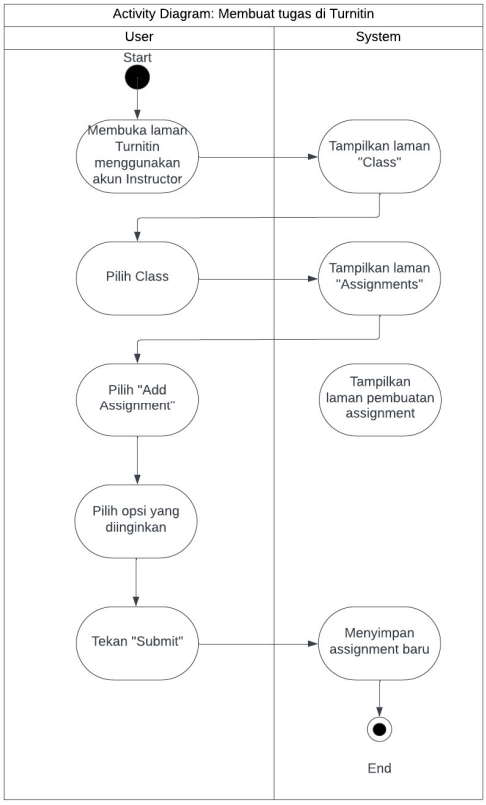

Figure 4. Activity diagram of the task creation process in Turnitin

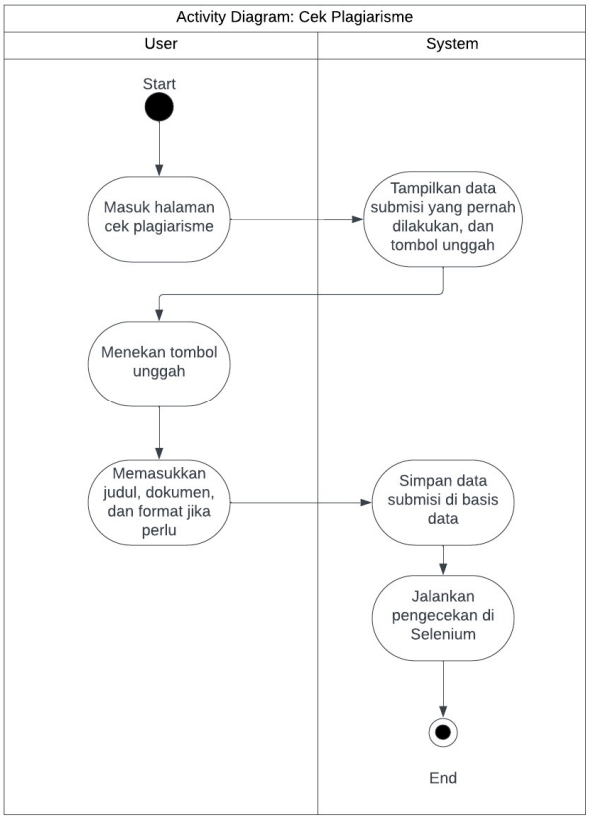

Figure 5. Activity diagram for plagiarism checkers

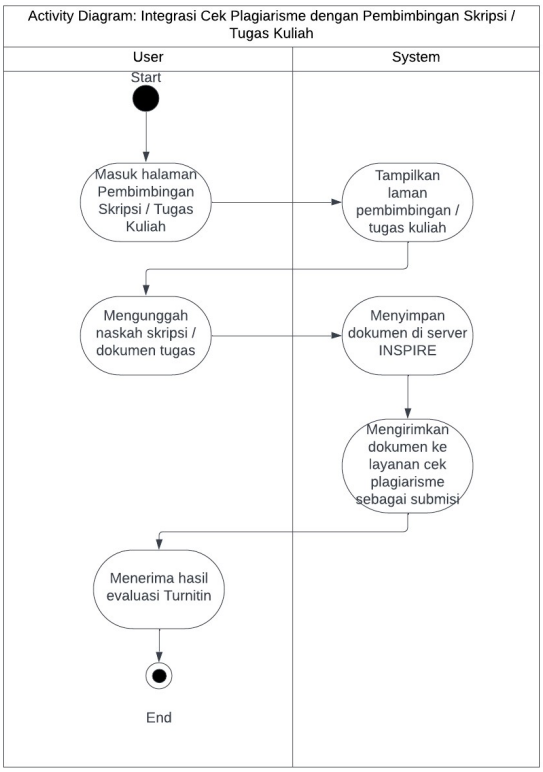

Figure 6. Activity diagram integration of plagiarism checking into the thesis and course assignment guidance feature

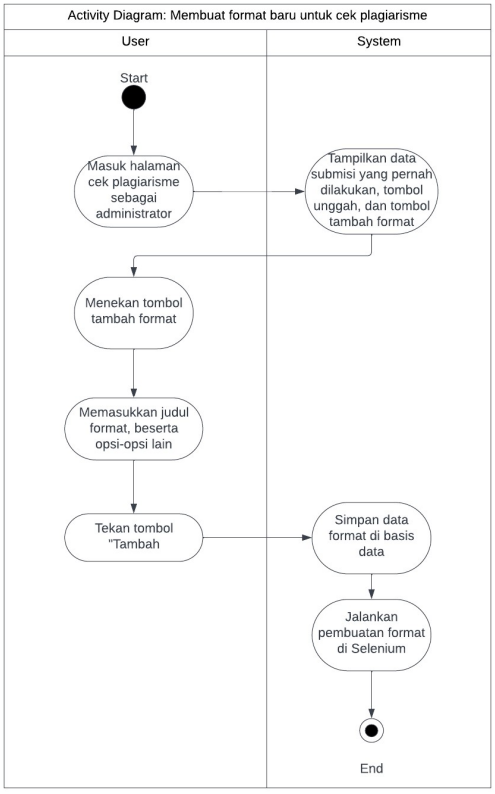

Figure 7. Activity diagram creating a new format for checking plagiarism

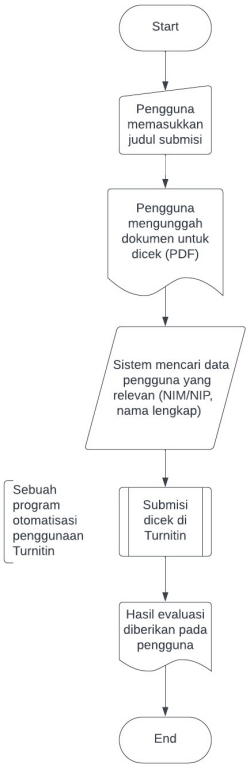

Figure 8. Design of the plagiarism checking process in INSPIRE

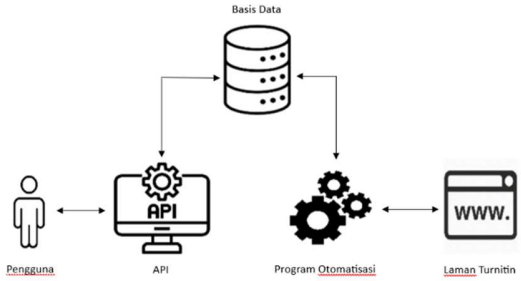

Figure 9. Plagiarism check service architecture

C. Testing

Before the plagiarism check feature can be launched, final testing is required to see whether the feature will work properly. If it is found that there are still bugs that were not discovered previously, then these bugs are fixed at this stage. Apart from bug fixes, optimization can also be carried out if it is found that the performance of this feature does not reach the expected performance.

D. Deployment

At this stage, all the functionality of the plagiarism check feature is running and ready to be launched. The developer collaborates with UPT TIK, in this case the frontend developer from UPT TIK, to discuss what user interface components will be displayed to users, based on the API documentation that has been created. After the display has been developed, the developer discusses with the INSPIRE administrator regarding the connection of the plagiarism check feature with the INSPIRE portal. If you can connect, the API program can be run and is ready to receive requests from users. The plagiarism check feature is ready to be launched live and can be accessed by INSPIRE users.

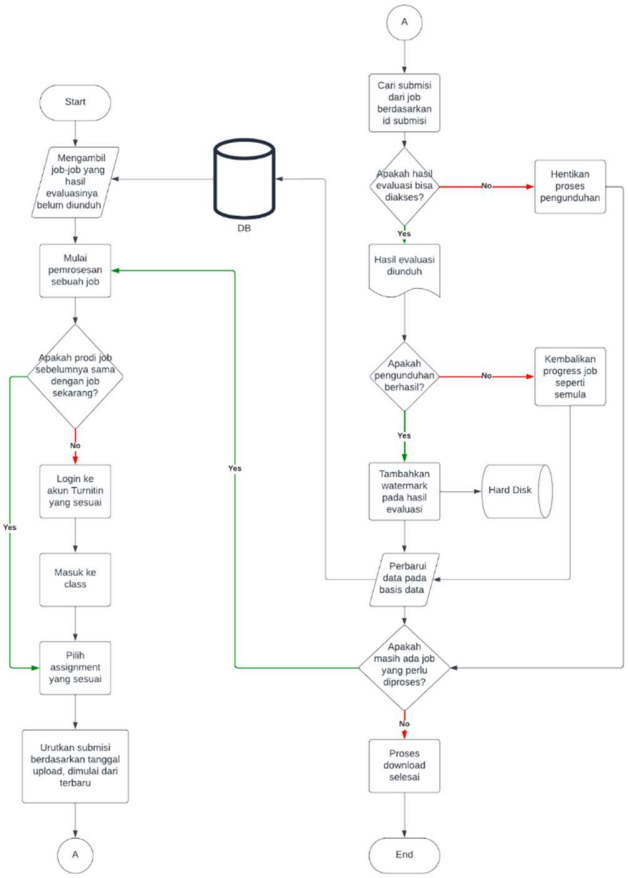

Figure 10. Upload process automation flow

```
def login(self, credential):
  username = self.driver.find_element(By.NAME, "email")
  password = self.driver.find_element(By.NAME, "user_password")
  username.clear()
  username.send_keys(credential[0])
  password.clear()
  password.send_keys(credential[1])
  randSleep()
  self.driver.find_element(By.NAME, "submit").click()
  print(getCurrentTimestamp() + " Proses: " + str(os.getpid()) + " Log: Selesai melakukan login")
```

Figure 11. Example of using Selenium for website navigation

III. Results And Discussions

A. Selenium's Usages

Selenium WebDriver is used to automate the use and navigation of a website in a browser. In this automation, WebDriver provides a number of program functions to be able to search for elements in HTML and perform certain actions on those elements. The code is then sent to the browser as input as if a human were using it. An example of the login process carried out by Selenium can be seen in Figure 10.

A few Selenium class and methods that usually used is find_element(), send_keys(), click(), and WebDriverWait.

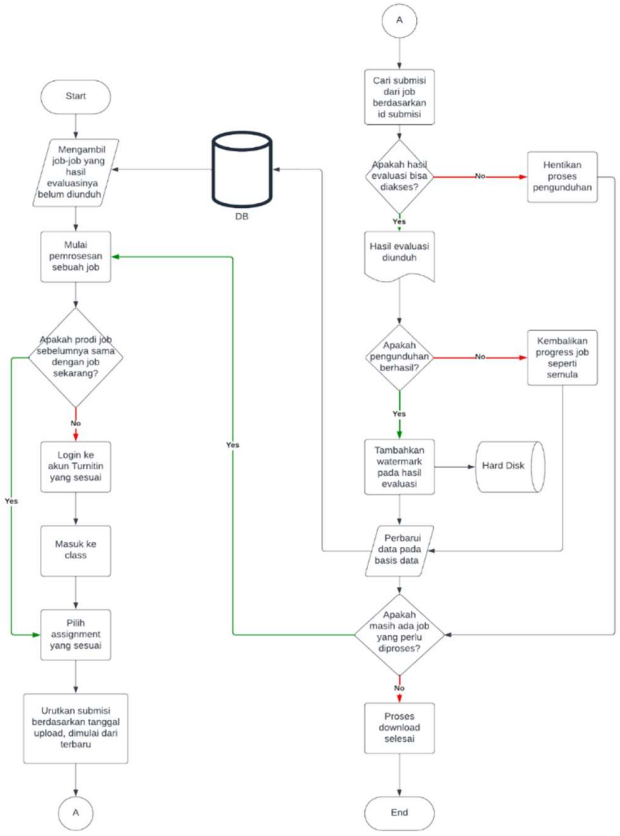

Figure 12. Download process automation flow

B. Turnitin's Automation

The plagiarism check service was built by utilizing the Turnitin service and connecting it to the INSPIRE portal. In this research, the use of Turnitin was carried out by automating Turnitin using a program. Turnitin's HTML elements that are normally navigated by humans, such as typing forms and clicking buttons, are automated using automation programs, so that these interactions are carried out by the computer according to programmed steps.

The upload process is carried out to upload documents to be checked from INSPIRE to Turnitin. The upload process flow is in the flowchart in Figure 10.

Once the similarity check is complete, Turnitin will output the evaluation results. These results are downloaded by the user as document improvement material. The download process flow is in the flowchart in Figure 11.

Evaluation of document similarity is carried out within Turnitin, which is then accessed by users via the INSPIRE portal. To differentiate Turnitin checks carried out in INSPIRE from regular Turnitin checks, a Sam Ratulangi University logo watermark is used, which is affixed to each page of the similarity evaluation results. An example of the downloaded evaluation results is in Figure 12.

TABLE I
LIST OF ALL API ENDPOINTS CREATED

No.	Endpoint	Fungsi
1.	GET /plagiarisme	Retrieve data on submissions that have been made by a user
2.	POST /plagiarisme	Sending user submissions for review on Turnitin
3.	DELETE /plagiarisme	Deletes user submissions
4.	GET /plagiarisme/format	As a user, take the format created by the admin
5.	GET /plagiarisme/result-files	Retrieves Turnitin evaluation results files from a submission
6.	GET /plagiarisme/submission-files	Retrieve user uploaded documents for a submission
7.	GET /plagiarisme/skripsi	Retrieve all submissions made by users in the thesis guidance feature
8.	GET /plagiarisme/tugas	Retrieve all submissions made by users in the coursework feature
9.	GET /plagiarisme/admin	As an admin, take all submissions made by all users from a particular study program
10.	GET /plagiarisme/admin/formats	As admin, retrieves all formats created by the admin, including the configuration used
11.	POST /plagiarisme/admin/format	As admin, add new formats to be used by users of one program

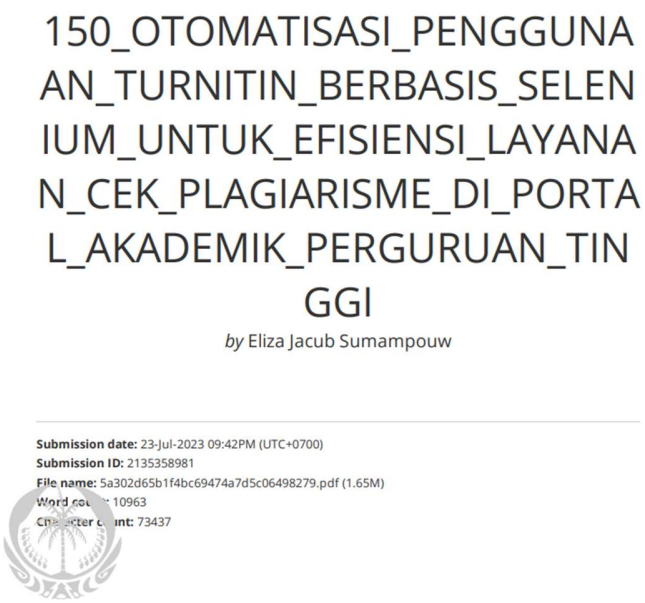

Figure 13. Turnitin evaluation results with Unsrat watermark

p-ISSN : 2301-8364, e-ISSN : 2685-6131 , available at : https://ejournal.unsrat.ac.id/index.php/informatika

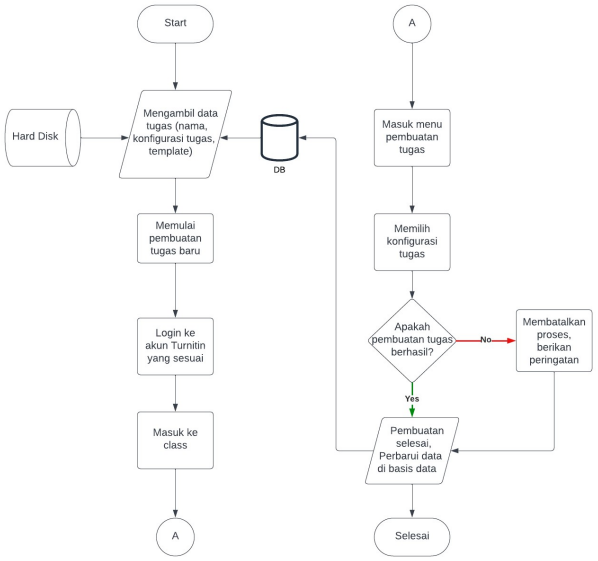

Figure 14. Task creation automation flow

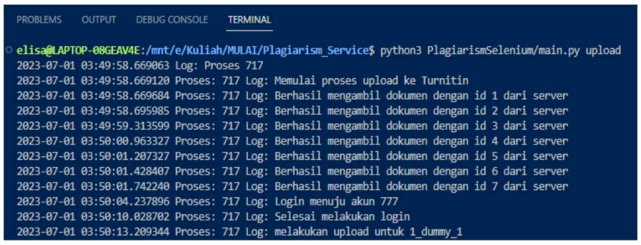

Figure 15. Selenium automation testing

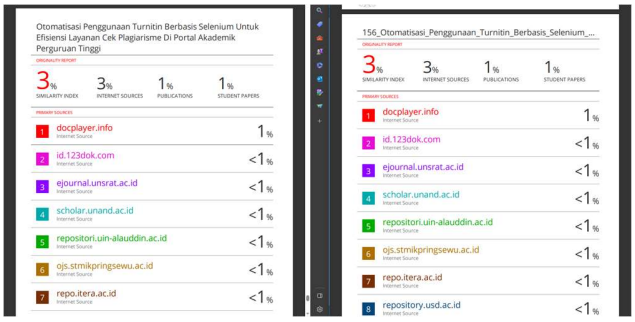

Figure 16. Comparison of manual Turnitin checking results and Turnitin automation download results

Creating a format is done by creating a new task in Turnitin with a number of configurable parameters. With the formats available in INSPIRE, users can upload documents that match the selected format, so that the similarity of documents can be reduced. The download process flow is depicted in the flowchart in Figure 13.

C. API Implementation

An automation program was created to process submissions to Turnitin based on existing data. The required data comes from users who submit submissions to INSPIRE. Data entered into INSPIRE is stored in a database, which will later be accessed by Selenium automation. This means that automation from Selenium does not receive data directly from the user, but from a database that already contains submission data.

To manage how users interact with automation, and to ensure that other INSPIRE services can communicate with the plagiarism check service, a REST API was built which acts as a gateway between users or external services and the Selenium automation program that has been created. APIs are created to ensure what data can enter and exit, how the data delivery format must be constructed, and what the data response structure will be. Users or services access the plagiarism check service via API endpoints, each of which is responsible for providing certain functionality.

The endpoints that are built are divided into four large groups, namely endpoints that manage the plagiarism check feature, endpoints that are built to manage integration with other services, which in this research are thesis and coursework guidance, and endpoints that are responsible for administering this service and only can be accessed by the head of the study program. The 11 endpoints created are listed in table 1.

The first endpoint group is responsible in handling the general flow of the plagiarism checker. Users can upload documents that Turnitin wants to check, view submissions that have been checked for plagiarism, and delete a submission. Users can also download the uploaded documents again, and when the check is complete, users can download the evaluation results of the check. When making a submission, users can use the formats available for their study program.

The second group handle plagiarism checker's integration into other service. For example, in thesis supervision, the plagiarism check service is used when the user uploads a thesis draft to be checked by the supervisor. To view thesis submissions made by a user, a special endpoint is provided to retrieve thesis submissions, via GET /plagiarism/thesis. To make a submission, the endpoint called is POST /plagiarism.

The third group handles the service's administration. To manage the plagiarism check service, functionality was created for an admin, who for this service is the head of each faculty's study program. An admin can see all submissions from all users in a particular study program. The admin can create a checking format, where there are a number of configurations that can be done, and the admin can also see the formats that have been created along with the configurations used.

D. Testing

Testing is carried out to ensure the services being developed run as planned. The aspects looked at are the suitability of the service based on what is programmed, and how optimal the processing capability is when faced with the scale of academic portal users.

The development of the Turnitin automation program was carried out in Windows Subsystem for Linux (WSL), a feature of the Windows operating system that allows users to use the Linux development environment, without the use of a Virtual Machine. WSL is used to standardize system environments, where development environment use Windows and INSPIRE production environments uses the Linux operating system. With WSL, we can adapt the operations carried out by the program to the existing operating system, such as the need to use a file system which is actually different from Windows. Developers can confidently test their programs without having to worry about interoperability to worry about interoperability between Windows and Linux.

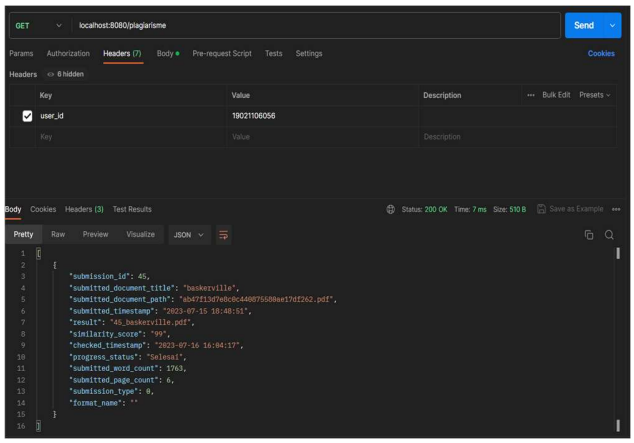

Figure 17. Functional testing of the plagiarism check submission API

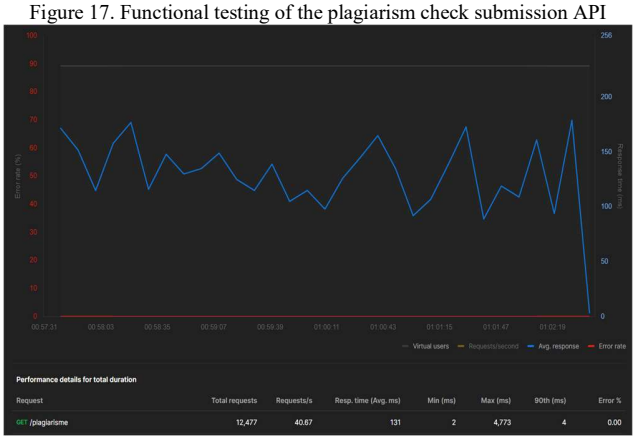

Figure 18. Stress testing result for GET /plagiarisme

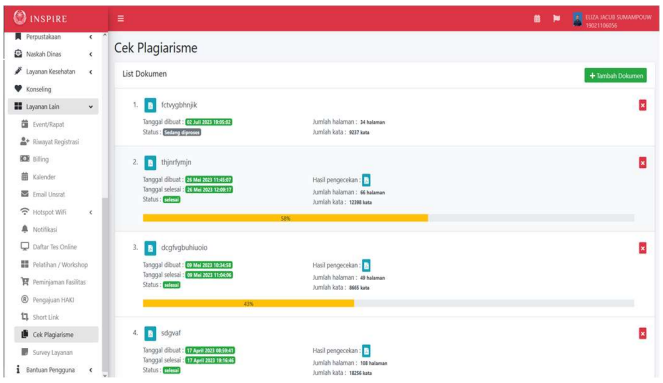

Figure 19. The "Plagiarism Check" display on INSPIRE

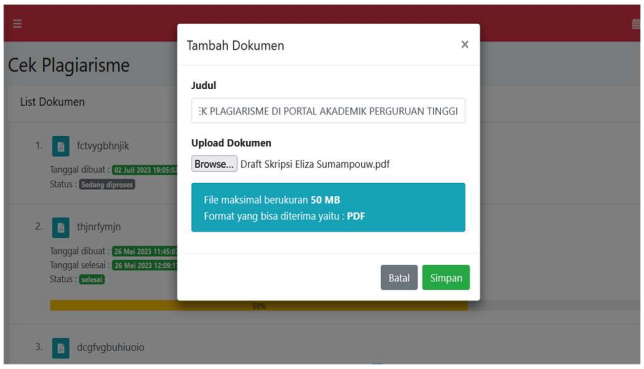

Figure 20. Document upload display in "Plagiarism Check"

WSL's biggest impact in this development is the certainty and simplicity of the file system used, especially when uploading and downloading operations are required that are closely related to what file system is used.

To ensure that Turnitin automation produces the same output as typical account usage, submission testing was carried out for 5 different documents. Testing was carried out on two different platforms, namely the Turnitin page and the automation program that was built. Similarities and differences are evaluated to ensure that the results of checks performed by Turnitin automation through Selenium are valid, and exactly match the results of checks performed when manually navigating Turnitin pages as an active user.

The differences described previously are differences resulting from platform differences. Meanwhile, for the checking results themselves, it can be seen in Figure 4.12 that the evaluation obtained is the same. Both downloaded results show a 3% similarity value with existing publications. The order of reference sources obtained is not displayed exactly the same, but it can be ensured that the reference sources remain the same for both submissions so that the evaluation results are also the same.

For API testing, the most popular application used is Postman, a multi-purpose application designed to help developers build and test APIs that have been created. With Postman, we can construct a request to the server by specifying a particular endpoint, what HTTP method to use, the contents of the header, body, and authorization used. The response from the server can also be seen here, where we can analyze the form of response given by the server, whether the message is clear and easy to understand, whether the response code used is appropriate, what data is provided, etc.

Apart from testing API functionality, Postman also provides a performance testing feature for APIs that have been created, in the form of stress tests. In the 12,477 requests given during the stress test, it was discovered that this endpoint could handle around 40.6 requests with an average time of 131 ms, and the fastest reply was 2 ms and the longest reply was 4733 ms or 4.7 seconds. Although there are variations in response times, in general these endpoints can handle user requests without significant delays.

E. Deployment

In deployment phase, the developer collaborates with the client, which in this case is UPT TIK, to discuss what user interface components will be displayed to users, based on the API documentation that has been created. The API document that is built is sent to the INSPIRE administrator so that data communication between INSPIRE and the plagiarism checking service can run well, and the service can be accessed by all INSPIRE users. INSPIRE's front-end display must also be adjusted to meet the data needs of the plagiarism checking service so that users can submit appropriate data.

After the plagiarism check service has been built, the next step is to launch the service on the INSPIRE portal. The INSPIRE developers created a service check page, a form for uploading document, you want to check, and an option to delete

submissions that have been made. To implement these functions, the endpoints used are GET /plagiarism, POST /plagiarism, and DELETE /plagiarism. The server behind the scene will operate a cron job that will periodically run the Selenium script, and each submission from INSPIRE's user will be processed accordingly. So, the front end and plagiarism checker can be connected and utilized without any human intervention whatsoever. The plagiarism checker is fully functional, and can be accessed from any member of the Universitas Sam Ratulangi, provided that they have access to the INSPIRE portal.

IV. Conclusions And Recommendations

A. Conclusions

Based on the research conducted, it can be concluded that Selenium can be used to create Turnitin automation which can be used as a plagiarism checking service to be integrated into the INSPIRE portal. Turnitin's automation-based plagiarism checking service using Selenium has been integrated into a university academic portal so that it can be used by students and lecturers to increase the efficiency of checking plagiarism. The results of checking via Turnitin automation do not have a significant difference with the results of checking directly from Turnitin, so these results are valid to be used as evaluation material. The performance of the plagiarism checking service is also able to handle a large number of requests.

B. Recommendations

Not all Turnitin features could be implemented in this research, such as the feature for uploading a number of files simultaneously and uploading zip files. Therefore, future researches can implement these features into automation. This research also uses Selenium to automate the use of Turnitin services, so future researches can explore the use of other automation tools, and see if other online services can be similarly automated using an automation tools, therefore cutting some expenses that would otherwise be incurred.

V. References

[1] A. Risparyanto, "Turnitin Sebagai Alat Deteksi Plagiarisme," *UNILIB : Jurnal Perpustakaan*, vol. 11, no. 2, Aug. 2020, doi: 10.20885/unilib.vol11.iss2.art5.

[2] Y. Cahyo Manunggal and L. Christiani, "Pemanfaatan Sistem Deteksi Plagiarisme Menggunakan Turnitin ® Pada Jurnal Mahasiswa Universitas Dian Nuswantoro," *Jurnal Ilmu Perpustakaan*, vol. 7, no. 2, pp. 231–240, 2018.

[3] S. Y. Sinaga, "Penggunaan Aplikasi Turnitin Sebagai Sarana Cek Plagiarisme Dalam Layanan Perpustakaan Universitas Ukrida," *Jurnal Kajian Perpustakaan dan Informasi BIBLIOTIKA : Jurnal Kajian Perpustakaan dan Informasi*, vol. 2, 2018.

[4] A. W. Suryani and B. Sugeng, "Can you find it on the web? Assessing university websites on academic integrity policy," in *2019 International Conference on Electrical, Electronics and Information Engineering (ICEEIE)*, 2019, pp. 309–313.

[5] R. W. Warouw, A. M. Sambul, and A. S. M. Lumenta, "Integrasi Layanan Cek Plagiarisme Pada Portal Akademik Di Perguruan Tinggi," *Jurnal Teknik Informatika*, vol. 17, no. 1, pp. 597–604, 2022.

[6] N. Noor Kamala Sari, "Rancang Bangun Website untuk Memeriksa Plagiat E-Journal Fakultas Kedokteran Universitas Palangka Raya," *Jurnal CoreIT*, vol. 5, no. 2, 2019.

[7] A. Mphahlele and S. McKenna, "The use of turnitin in the higher education sector: Decoding the myth," *Assess Eval High Educ*, vol. 44, no. 7, pp. 1079–1089, Oct. 2019, doi: 10.1080/02602938.2019.1573971.

[8] M. A. Umar and C. Zhanfang, "A study of automated software testing: Automation tools and frameworks," *International Journal of Computer Science Engineering (IJCSE)*, vol. 6, pp. 217–225, 2019.

[9] Supriyadi and M. Adhari Adiguna, "Perancangan Aplikasi Key Performance Indicator (KPI) Dashboard Berbasis Android Menggunakan React Native dan Backend API Untuk Memantau Kinerja Gudang dan Penjualan Barang Dengan Metode Rapid Application Development (RAD) (Studi Kasus : PT. Mega Dagang Internasional)," *Jurnal Penelitian Ilmu Komputer*, vol. 1, no. 2, pp. 20–28, May 2023, [Online]. Available: https://mypublikasi.com/index.php/JUPIK/20

[10] Kemdikbud, "Arti kata otomatisasi - Kamus Besar Bahasa Indonesia (KBBI) Online," Jul. 08, 2023. https://kbbi.web.id/otomatisasi (accessed Jul. 14, 2023).

[11] R. C. Dorf and A. Kusiak, *Handbook of Design, Manufacturing and Automation*. Wiley-Interscience, 1994.

[12] Computer Hope, "What is Automation?," Jul. 02, 2022. https://www.computerhope.com/jargon/a/automati.htm (accessed Jul. 14, 2023).

[13] Software Freedom Conservancy, "About Selenium | Selenium," Feb. 19, 2020. https://www.selenium.dev/about/ (accessed Jan. 09, 2023).

[14] Software Freedom Conservancy, "Downloads | Selenium," 2020. https://www.selenium.dev/downloads/ (accessed Jan. 09, 2023).

[15] M. F. Blonteng, A. M. Sambul, and S. D. E. Paturusi, "Analysis of user experience in University Academic Portal using system usability scale (a case study in INSPIRE Portal of Sam Ratulangi University)," *Jurnal Teknik Informatika*, vol. 17, no. 3, pp. 213–218, 2022.

[16] C. Rawis, S. D. S. Karouw, and S. R. U. A. Sompie, "Software Requirement Specification Sistem Informasi Akademik Universitas Sam Ratulangi," *Jurnal Teknik Elektro dan Komputer*, vol. 10, no. 2, pp. 107–118, 2021.

[17] R. A. Sukamto and M. Shalahudin, *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek*. Bandung: Informatika Bandung, 2016.

Eliza Jacub Sumampouw is the third child of three. The author was born in Manado, August 8 2001. In 2006, the author started his elementary school education at SD 09 Hati Kudus Manado. The author moved to SD Don Bosco Manado when he was in second grade, 2008. In 2013, the author continued his education at SMP Negeri 1 Manado. Furthermore, in 2016, the author studied upper secondary education at SMA Negeri 9 Manado. In 2019, the author continued his undergraduate studies at Sam Ratulangi University, Informatics Engineering study program in the Department of Electrical Engineering, Faculty of Engineering. During his studies, the author joined organizations such as Himpunan Mahasiswa Elektro (HME) and Unsrat IT Community (UNITY).