Development of Automation Testing Tool for Android Set-Top Box

Submitted By

PALAK PUROHIT

21MCED10

NIRMA UNIVERSITY

INSTITUTE OF TECHNOLOGY

NAAC ACCREDITED 'A+' GRADE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2023

Development of Automation Testing Tool for Android Set-Top Box

Major Project - II

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology in Computer Science and Engineering (Data Science)

Submitted By

PALAK PUROHIT

(21MCED10)

Guided By

Prof. Kruti Lavingia

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2023

Certificate

This is to certify that the major project entitled **”Development of Automation Testing Tool for Android Set-Top Box”** submitted by **PALAK PUROHIT (21MCED10)**, towards the partial fulfillment of the requirements for the award of degree of Master of Technology in Computer Science and Engineering (Data Science) of Nirma University, Ahmedabad, is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project part-I, to the best of my knowledge, haven’t been submitted to any other university or institution for award of any degree or diploma.

Prof. Kruti Lavingia
Guide & Assistant Professor,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Prof./Dr. Swati Jain
Associate Professor,
Coordinator M.Tech - CSE (Data Science)
Institute of Technology,
Nirma University, Ahmedabad

Dr. Madhuri Bhavsar
Professor and Head,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Dr R. N. Patel
Director,
Institute of Technology,
Nirma University, Ahmedabad

Statement of Originality

———————————————————————————————

I, **Palak Purohit**, **21MCED10**, give undertaking that the Major Project entitled ”**Development of Automation Testing Tool for Android Set-Top Box**” submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in **Computer Science & Engineering (Data Science)** of Institute of Technology, Nirma University, Ahmedabad, contains no material that has been awarded for any degree or diploma in any university or school in any territory to the best of my knowledge. It is the original work carried out by me and I give assurance that no attempt of plagiarism has been made.It contains no material that is previously published or written, except where reference has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action. As a part of my curriculum I’m working in Technicolor Connected Home India Pvt. Ltd company under Vantiva Group as Software Intern. We are working on same project as I presented in my major project. We are Developing Automation Testing Tool for Android Set-Top Box.

————————

Signature of Student

Date:23/05/2023

Place:Ahmedabad

Endorsed by

Prof. Kruti Lavingia

(Signature of Guide)

Acknowledgements

It gives me immense pleasure in expressing thanks and profound gratitude to **Prof. Kruti Lavingia**, Assistant Professor, Computer Engineering Department, Institute of Technology, Nirma University, Ahmedabad for her valuable guidance and continual encouragement throughout this work. The appreciation and continual support she has imparted has been a great motivation to me in reaching a higher goal. Her guidance has triggered and nourished my intellectual maturity that I will benefit from, for a long time to come.

Thanks to Dr. Swati Jain, Assistant Professor and Coordinator M.Tech - CSE, Institute of Technology, Nirma University, Ahmedabad for his kind support and providing knowledge.

It gives me an immense pleasure to thank Dr. Madhuri Bhavsar, Hon’ble Head of Computer Engineering, Institute of Technology, Nirma University, Ahmedabad for his kind support and providing basic infrastructure and healthy research environment.

A special thank you is expressed wholeheartedly to Dr. R. N. Patel, Hon’ble Di- rector, Institute of Technology. Nirma University, Ahmedabad for the unmentionable motivation he has extended throughout course of this work.

I would also thank the Institution, all faculty members of Computer Engineering Department, Nirma University, Ahmedabad for their special attention and suggestions towards the project work.

I convey thanks to my manager Mr. Vikram Dutta for providing encouragement, constant support and guidance which was of a great help in successfully completing the Major Project.

Finally, my deepest gratitude goes to my parents who have given me much needed comfort, support, encouragement and inspiration for completing this project.

- Palak Purohit
21MCED10

Abstract

Android set-top boxes (STBs) have become popular devices for streaming and watching digital content. However, testing Android STBs poses a significant challenge due to their complex hardware and software configurations. Manual testing can be time-consuming, error-prone, and expensive. Automation testing has emerged as a potential solution to these challenges. This paper presents a survey of the state-of-the-art in automation testing for Android STBs. The paper discusses the challenges of testing Android STBs, the benefits of automation testing, and the tools and techniques used for automation testing. The paper also provides a review of recent research in the field, as well as an assessment of the future direction of automation testing for Android STBs.

Abbreviations

STB	Set-Top Box.
AI	Artificial Intelligent.
BDD	Behavior-Driven Development.
ML	Machine Learning.

——

–

Contents

List of Figures

Chapter 1

Introduction

In the recent years android set-top boxes (STBs) become more popular. It is allowed users to access the various content on their TVs. Due to their complex hardware as well as software configurations testing of Android Set-Top box represents unique challenges. Compare to manual testing automation testing consume less time, less expensive and error free[11]. This paper presents the survey of state-of-art in automation testing for android STBs. This paper discusses the benefits of automation testing, challenges of testing Android STBs and tools and techniques used for automation testing. As the complexity of Android STBs increases, so too does the need for effective testing. Testing Android STBs poses several unique challenges, including the wide range of hardware and software configurations available and the need to test integration with external devices. In addition, manual testing can be time-consuming and error-prone, especially for complex use cases.

These problems can be overcome via automation testing, which offers a quicker and more thorough way to test Android STBs. Automation testing can increase test coverage, decrease the time needed to complete test cases, and boost testing quality. Android Set-Top Boxes (STBs) are becoming more and more well-liked as a result of their capacity to give users access to a wide range of entertainment content[1]. To find and fix any problems or flaws that can affect the user experience, Android STBs must, nevertheless, undergo comprehensive testing. Automation testing is a desirable alternative to manual testing because it can be time-consuming, expensive, and prone to mistakes. The software development life cycle now includes automation testing, which helps engineers test programmes quickly, effectively, and consistently.

The primary objective of automation testing is to automate repetitive and time consuming manual testing processes to accelerate testing cycles and improve the quality of software. Automation testing also allows developers to test a broader range of scenarios, configurations, and environments, which can added the test coverage and improve the reliability and accuracy of testing. These benefits enable developers to release applications faster and more efficiently while ensuring that the application is in depth tested.

In the context of Android STBs, automation testing can ensure that the applications are compatible with different shapes, sizes, and configurations of devices. Automation testing can also test different features and functionalities of the STBs, such as remote control, HDMI, and Wi-Fi connectivity. This is crucial to ensure that the Android STBs meet the expectations of users, who demand a high level of performance and functionality.

Automation testing for Android STBs does present some difficulties, though. Device fragmentation is one of the main issues, which might make it difficult to ensure that the programme has been adequately tested across all devices. The limited access to hardware, which is necessary for testing various STB features and functionalities, presents another difficulty. Additionally difficult issues that developers encounter when integrating automation testing into their development process include lack of standardisation and complexity.

Despite these difficulties, automation testing for Android STBs has advanced significantly, with new tools and technologies developing to overcome these difficulties.

Chapter 2

Literature Survey

The demand for effective and efficient testing has increased as Android set-top boxes (STBs) become more famous. Numerous tools and frameworks are available for testing Android STBs, which is a popular way for testing software. An overview of the state of Android STB automation testing is given in this literature study.

In order to undertake the review of the literature, academic journals and conference proceedings were searched for articles and papers. Several search terms were used, including ”Android STB,” ”automation testing,” ”Android testing,” and ”set-top box testing,” ”test automation”.

Findings: According to the literature review, there are numerous frameworks and testing tools for Android STBs. the most widely utilised instruments. These programmes offer a variety of features, including performance testing, compatibility testing, and UI testing.

The tools/frameworks discussed in the literature study as well as the Behave framework each have advantages and drawbacks of their own. Although adaptable and customizable, the Behave framework has a high learning curve and requires programming experience. It is a sophisticated BDD implementation. Although Appium, UI Automator, and MonkeyRunner are all free and provide a variety of testing tools, they each have their own drawbacks, such as a high learning curve and restricted device compatibility.

Although Calabash and Espresso are also free, they have fewer supported devices and

Table 2.1: Literature Review

Paper	Research Objective	Methods	Tools	Results
[3]	To develop an automated testing tool for Android set-top boxes	Implemented a test automation tool using Java, Android SDK, and Apache Maven	Java, Android SDK, Apache Maven	The developed tool was able to automate testing of the user interface and user input on an Android set-top box
[5]	To compare the performance of UI Automator and Appium for Android set-top box automation testing	Conducted a series of tests using both tools on an Android set-top box	UI Automator, Appium	UI Automator had a faster response time and was more stable than Appium. However, Appium had more features and better documentation
[9]	To evaluate the effectiveness of different automation testing tools for Android set-top boxes	Conducted a study using different automation testing tools on an Android set-top box	Espresso, UI Automator, Robotium, Calabash	Espresso and UI Automator were found to be the most effective tools for automation testing on an Android set-top box
[12]	To evaluate the usability and effectiveness of a BDD approach for Android set-top box automation testing	Conducted a study using Behave, a BDD framework, for automation testing on an Android set-top box	Behave	The BDD approach improved communication between technical and non-technical stakeholders and Behave was found to be an effective tool for automation testing on an Android set-top box

provide special capabilities like BDD support and sophisticated UI interactions. The Robot Framework interfaces with Appium and Selenium and provides keyword-driven testing, although it only partially supports Android and therefore requires programming experience. The final decision on the tool or framework to use is determined by the project's unique needs as well as the testing team's level of experience.

Chapter 3

Benefits of Automation Testing

Automation testing provides several benefits for testing Android STBs. These benefits include increased testing efficiency, improved test coverage, and reduced testing costs and time. Automation testing can also improve the accuracy and repeatability of test results, making it easier to identify and fix bugs. These benefits enable developers to release applications faster and more efficiently while ensuring that the application is thoroughly tested.

3.0.1 Repeatable Tests

Automation testing enables developers to execute the similar tests repeatedly, to ensuring that the application is consistently tested and reducing the risk of issues and defects going unnoticed. This is particularly important for Android STBs, which require a high level of consistency to ensure a good user experience.

3.0.2 Reduced Time and Cost

Automation testing can shrink the amount of money and time needed for testing, enabling developers to release applications more faster and effectively. This is because of the fact that automation testing restore manual testing, which can be resource and time consuming..

3.0.3 Increased Test Coverage

Automation testing can help ensure that the application is in depth tested across all devices. Automation testing enables developers to test a broader range of scenarios, environments, and configurations, increasing the application’s test coverage. This is specifically important for Android STBs, which come in different shapes, configurations and sizes.

3.0.4 Improved Accuracy and Reliability

Automation testing remove human errors and ensures that test cases are executed consistently, improving the accuracy and reliability of the testing process. This is specially important for Android STBs, which require a high level of accuracy and reliability to ensure a good user experience.

Chapter 4

Benefits of Automation Testing

The complexity of the hardware and software setups, the requirement for intensive manual testing, and the difficulty in simulating real-world usage scenarios make it challenging to test Android STBs. Manual testing takes a lot of time and is prone to error. Furthermore, it is not possible to manually verify each configuration because to the enormous variety of STB models and setups. By automating the execution of test cases and lowering the amount of manual labour necessary for testing, automation testing can assist in addressing these issues. Despite the advantages of automation testing, applying it in Android STBs presents a number of difficulties for developers[8]. Device fragmentation, restricted hardware availability, a lack of standards, and complexity are some of these difficulties.

4.0.1 Limited Access to Hardware

This can make it challenging for developers to in depth test the application because automation testing requires access to the hardware to test different features and functionalities, which may not possible with Android STBs due to restricted hardware access.

4.0.2 Device Fragmentation

it can be difficult to guarantee that the programme has been extensively tested on all of them, Because Android STBs come in a variety of dimensions, styles, and configurations. Because of this, it may be challenging for developers to warranty that their application works with all Android STBs.

4.0.3 Lack of UI Standardization

There is no standardization for Android STBs, which means that testing frameworks and methodologies may not be compatible across different devices. This can make it challenging for developers to ensure that the application is in depth tested over all Android STBs.

4.0.4 Complexity

The process of automating testing for Android STBs is complicated and time- and resource-intensive to set up and maintain. Developers may find it difficult to integrate automation testing into their development process as a result.

Chapter 5

Tools and Techniques for Automation Testing

Several tools and methodologies for automation testing in Android STBs have been developed. Frameworks such as Appium, Behave Framework, and UI Automator, perfecto, among others, enable the automation of user interactions with the STB. They can also simulate human input and record the ensuing STB output[7]. In addition to these frameworks, numerous programming languages and testing frameworks, like as Python and the Robot Framework, have been utilised for automation testing in Android STBs. Here are some of the most popular tools and strategies for android STB automated testing:

5.0.1 Appium:

Appium is a free and open-source automation tool for testing native, and hybrid, mobile web apps. It provides a framework for testing android STBs and allows testers to develop automated tests in a collection of programming languages like Java, Ruby, Python, and others. Cross-platform testing for Android and iOS devices is also supported by Appium.

5.0.2 Behave Framework:

It is built on the Gherkin programming language and provides a framework for developing and running tests in a human-readable style. Behave is a popular tool for testing Python program using Behavior-Driven Development (BDD). Behave enables testers and

Table 5.1: Tools and Techniques in Automation

Tool Framework	Behave Framework	Appium	UI Automator	Monkey Runner	Calabash	Espresso	Robot Framework
Language	Python	Multiple languages supported	Java	Python	Ruby	Java/Kotlin	Python
Cross Platform	Yes	Yes	No	Yes	Yes	No	Yes
Learning Curve	Steep	Steep	Moderate	Easy	Moderate	Moderate	Moderate
Device Support	Limited	Wide Range	Limited	Limited	Limited	Limited	Limited
Test Features	UI Testing, BDD	UI Testing, Compatibility Testing	UI Testing, Integration Testing	UI Testing, Regression Testing	UI Testing, Regression Testing	UI Testing, Performance Testing	UI Testing, Regression Testing
Advantages	Powerful BDD implementation, flexible and customizable	Cross-platform support	Built-in tool for Android devices	Easy to use and script in Python	Easy to use, supports BDD	Fast and reliable	Supports keyword-driven testing
Disadvantages	Steep learning curve, limited device support, requires programming	Steep learning curve, complex setup	Limited functionality	Limited device support	Limited device support, lacks integration testing	Requires programming knowledge	Limited Android support, requires programming knowledge
Cost	Free	Free	Free	Free	Free	Free	Free

developers to cooperate and produce tests in a common language that non-technical stakeholders can comprehend.

Behave defines an application’s behavior using feature files that comprise scenarios, steps, and step definitions. The feature files are produced in plain text format, allowing the stakeholders and development team to happily exchange and review them. Behave provides a set of pre-defined steps that may be used to rapidly and easily write tests.

Behave is frequently used to test various sorts of applications, such as online apps, APIs, and desktop applications. It provides UI testing, user interaction testing, and cross-device testing, among other automated features. Parallel testing and interaction with other testing frameworks such as Selenium and Appium are also supported by Be- have.

To summarise, Behave is an effective solution for BDD testing of Python applications. It allows developers and stakeholders to cooperate and understand the application’s behaviour by providing a human-readable language for authoring tests. Behave also offers a variety of automation tools as well as connection with other testing frameworks, making it a popular choice for testing a wide range of applications.

5.0.3 UI Automator:

UI Automator is meant to test applications at the UI level and includes a comprehensive range of UI testing capabilities. UI Automator is a testing framework for testing Android applications. UI Automator provides a set of APIs for developing automated tests that link with an application’s user interface.

5.0.4 stb-tester:

An STB tester is a professional charged with testing set-top boxes (STBs) to ensure that they meet the required performance, functionality and usability standards. STB testers frequently work for software development companies or STB manufacturers in the software development industry.

5.0.5 Perfecto:

This cloud-based testing tool allows you to test OTT applications on a different different devices, including Android set-top boxes. Perfecto supports both manual and automation testing and includes capabilities like parallel testing, AI-powered testing, and real-time test result monitoring.

5.0.6 Stb-tester v/s Behave

STB and behave testers have various responsibilities in the software development process. Behave is a testing framework that aids in the automation of software application testing, whereas an STB tester is a specialist in charge of testing set-top boxes (STBs).

Behave is a common testing framework for behavior-driven development (BDD), which entails specifying an application's desired behavior in terms of use cases and scenarios. Behave enables developers and testers to create tests in normal language using Gherkin syntax, making communication with non-technical stakeholders easier.

An STB tester, on the other hand, charge of testing STBs to ensure that they fulfil the required criteria for performance, functionality, and usability. This includes creating and executing test plans and test cases to validate the functionality and performance of STBs, discovering and reporting software flaws or problems, and collaborating with developers to remedy them.

While Behave can be used to test software programmes running on STBs, it is not intended to test STBs themselves. STB testers, instead, have a in-depth understanding of STB software and hardware and have been specifically educated to evaluate STB functionality and performance.

In conclusion, the roles played by Behave and STB testers during the software development process differ. Behave is a testing framework for automating software application testing, in other hand STB testers are experts in charge of testing STBs to make sure

Table 5.2: Behave Framework vs Stb-tester

	Behave Framework	Stb-tester
Role	Testing framework	Professional tester
Focus	Testing software apps	Testing STBs
Responsibilities	Write tests in natural language using Gherkin syntax, automate testing process for software applications	Design and execute test plans and cases, identify and report defects or bugs in STB software, ensure STBs meet performance, functionality, and usability standards
Technical Skills	Proficiency in programming languages (Python), understanding of behavior-driven development (BDD) and testing frameworks	Understanding of software development lifecycle, testing methodologies, and various testing techniques and tools, knowledge of STB hardware and software
Communication Skills	Ability to communicate effectively with developers and non-technical stakeholders through natural language syntax	Excellent communication and collaboration skills to work closely with developers, product managers, and other stakeholders to ensure STBs meet quality standards
Testing Capabilities	Provides testing capabilities for software applications that run on STBs	Comprehensive range of testing functionalities for STBs, including video playback testing, remote control testing, HDMI testing, etc.
Integration with Other Tools	Supports integration with various testing and development tools through plugins and extensions	Supports integration with JIRA, Jenkins, and other tools
Programming Language	Python-based	Python-based
User Interface	Command line interface for running tests	Graphical user interface for setting up and running tests
Supported Platforms	Linux, macOS, Windows	Linux, macOS, Windows
Cost	STB-tester is a commercial tool and requires a license	Behave is an open-source framework and is free to use

Table 5.3: UI Automator vs Behave

Framework	License	Programming Language	Cross-Platform Testing	Device Pool	User Interface Testing	BDD Support	Integration with CI/CD Tools
Behave	Open-Source	Python	Yes	N/A	Yes	Yes	Jenkins, Travis CI, etc.
UI Automator	Open-Source	Java	No	Limited	Yes	No	Android Studio

they stick to the necessary performance, functionality, and usability criteria.

Chapter 6

Implemented Work

6.1 FRAMEWORK USED FOR AUTOMATION

- **Behave Enhancement Tool:**
  - Behave test framework based will be a part of existing framework
  - Automation solution for test validation of Android/Linux/RDK or Any OS
  - Single automation solution for technicolor or non-technicolor Box
  - Provides a subset of professional paid automation tool(WITBE)
  - Can be used on Secured/Debug boxes
  - Quick and Easy Setup
  - End to End integration

6.2 HARDWARE REQUIRED and SETUP

- Linux PC
- HDMI Video capture card
- IR – Red rat or any other IR blaster device
- Set-Top Box
- HDMI Cable
- E-POWER SWITCH

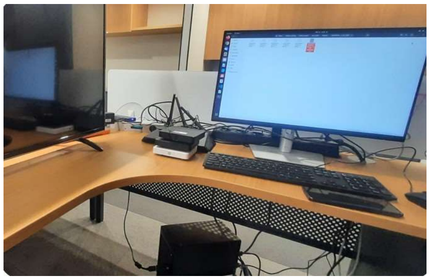


Figure 6.1: Setup

6.3 CONNECTION DIAGRAM

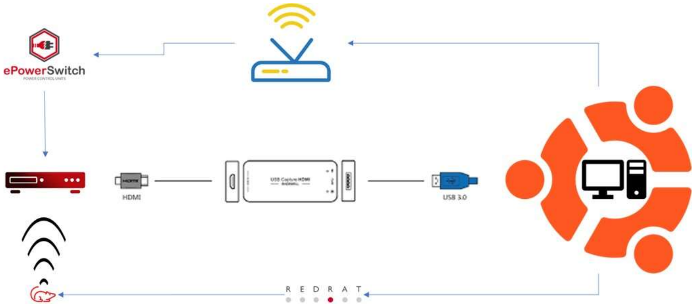


Figure 6.2: Connection Diagram

6.4 HIGH LEVEL DESIGN

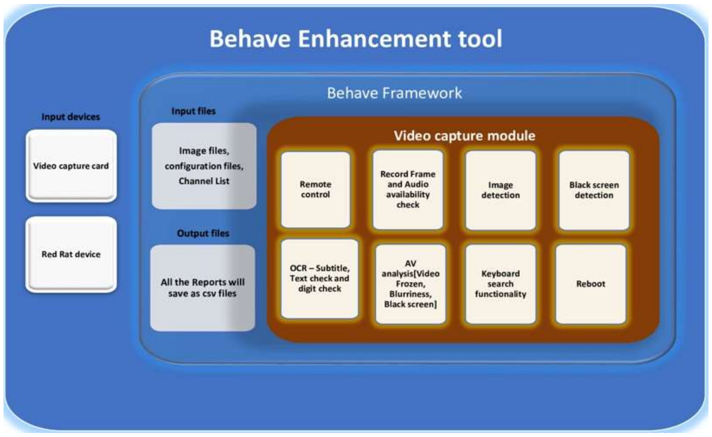

Behave Enhancement tool
Input devices
Video capture card
Red Rat device
Behave Framework
Input files
Image files, configuration files, Channel List
Output files
All the Reports will save as csv files
Video capture module
Remote control
Record Frame and Audio availability check
Image detection
Black screen detection
OCR – Subtitle, Text check and digit check
AV analysis[Video Frozen, Blurriness, Black screen]
Keyboard search functionality
Reboot

Figure 6.3: High Level Design

6.5 FLOW OF THE TOOL

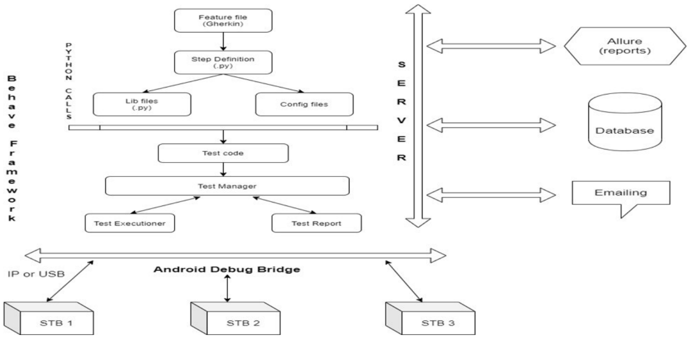


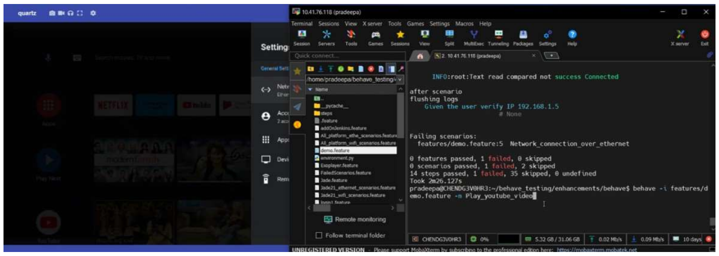
Figure 6.4: Flow Of The Tool

6.6 IMPLEMENTED FEATURES

- Video channel change and zapping, video continuity check feature
- Black screen detection
- OCR Integrated - Full screen - Region wise
- Reboot and Reset
- IR control using REDRAT

- Opening any App like Youtube/Netflix and validating the same including launch time
- Audio Basic Check
- Keyboard Search Functionality
- Jira Integration

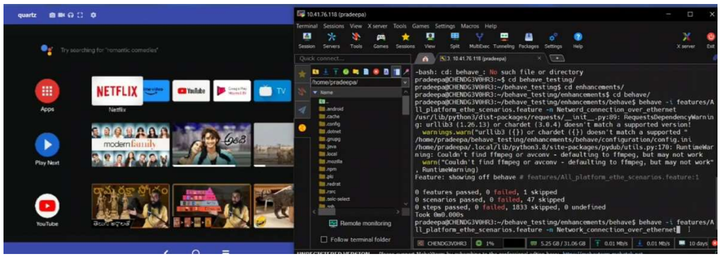
6.7 IMAGE/AUDIO/VIDEO VERIFICATION



```
    INFO:root:Text read compared not success Connected

after scenario
flushing logs
    Given the user verify IP 192.168.1.5
                        # None


Failing scenarios:
  features/demo.feature:5  Network_connection_over_ethernet

0 features passed, 1 failed, 0 skipped
0 scenarios passed, 1 failed, 2 skipped
14 steps passed, 1 failed, 35 skipped, 0 undefined
Took 2m26.127s
pradeepa@CHENDG3V0HR3:~/behave_testing/enhancements/behave$ behave -i features/d
emo.feature -n Play_youtube_video
```

Figure 6.5: IMAGE/AUDIO/VIDEO VERIFICATION

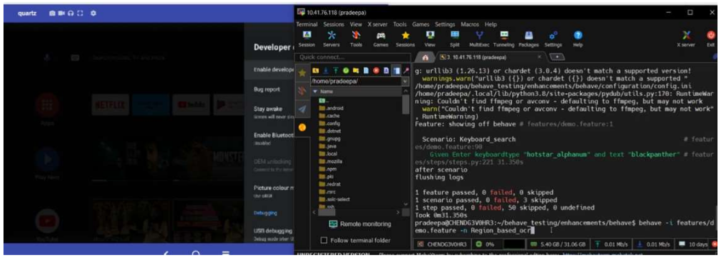
6.8 OCR



```
-bash: cd: behave_: No such file or directory
pradeepa@CHENDG3V0HR3:~$ cd behave_testing/
pradeepa@CHENDG3V0HR3:~/behave_testing$ cd enhancements/
pradeepa@CHENDG3V0HR3:~/behave_testing/enhancements$ cd behave/
pradeepa@CHENDG3V0HR3:~/behave_testing/enhancements/behave$ behave -i features/A
ll_platform_ethe_scenarios.feature -n Netword_connection_over_ethernet
/usr/lib/python3/dist-packages/requests/__init__.py:89: RequestsDependencyWarnin
g: urllib3 (1.26.13) or chardet (3.0.4) doesn't match a supported version!
  warnings.warn("urllib3 ({}) or chardet ({}) doesn't match a supported "
/home/pradeepa/behave_testing/enhancements/behave/configuration/config.ini
/home/pradeepa/.local/lib/python3.8/site-packages/pydub/utils.py:170: RuntimeWar
ning: Couldn't find ffmpeg or avconv - defaulting to ffmpeg, but may not work
  warn("Couldn't find ffmpeg or avconv - defaulting to ffmpeg, but may not work"
, RuntimeWarning)
Feature: showing off behave # features/All_platform_ethe_scenarios.feature:1

0 features passed, 0 failed, 1 skipped
0 scenarios passed, 0 failed, 47 skipped
0 steps passed, 0 failed, 1833 skipped, 0 undefined
Took 0m0.000s
pradeepa@CHENDG3V0HR3:~/behave_testing/enhancements/behave$ behave -i features/A
ll_platform_ethe_scenarios.feature -n Network_connection_over_ethernet
```

Figure 6.6: OCR

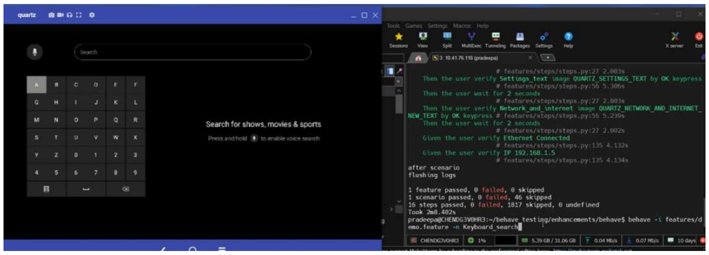
6.9 Region Based OCR



```
g: urllib3 (1.26.13) or chardet (3.0.4) doesn't match a supported version!
  warnings.warn("urllib3 ({}) or chardet ({}) doesn't match a supported "
/home/pradeepa/behave_testing/enhancements/behave/configuration/config.ini
/home/pradeepa/.local/lib/python3.8/site-packages/pydub/utils.py:170: RuntimeWar
ning: Couldn't find ffmpeg or avconv - defaulting to ffmpeg, but may not work
  warn("Couldn't find ffmpeg or avconv - defaulting to ffmpeg, but may not work"
, RuntimeWarning)
Feature: showing off behave # features/demo.feature:1

  Scenario: Keyboard_search                                              # featur
es/demo.feature:98
    Given Enter keyboardtype "hotstar_alphanum" and text "blackpanther" # featur
es/steps/steps.py:221 31.350s
after scenario
flushing logs

1 feature passed, 0 failed, 0 skipped
1 scenario passed, 0 failed, 3 skipped
1 step passed, 0 failed, 50 skipped, 0 undefined
Took 0m31.350s
pradeepa@CHENDG3V0HR3:~/behave_testing/enhancements/behave$ behave -i features/d
emo.feature -n Region_based_ocr
```

Figure 6.7: Region Based OCR

6.10 Keyboard Search



```
                         # features/steps/steps.py:27 2.003s
    Then the user verify Settings_text image QUARTZ_SETTINGS_TEXT by OK keypress
                         # features/steps/steps.py:56 5.306s
    Then the user wait for 2 seconds
                         # features/steps/steps.py:27 2.003s
    Then the user verify Network_and_internet image QUARTZ_NETWORK_AND_INTERNET_
NEW_TEXT by OK keypress # features/steps/steps.py:56 5.239s
    Then the user wait for 2 seconds
                         # features/steps/steps.py:27 2.002s
    Given the user verify Ethernet Connected
                         # features/steps/steps.py:135 4.132s
    Given the user verify IP 192.168.1.5
                         # features/steps/steps.py:135 4.134s
after scenario
flushing logs

1 feature passed, 0 failed, 0 skipped
1 scenario passed, 0 failed, 46 skipped
16 steps passed, 0 failed, 1817 skipped, 0 undefined
Took 2m8.402s
pradeepa@CHENDG3V0HR3:~/behave_testing/enhancements/behave$ behave -i features/d
emo.feature -n Keyboard_search
```

Figure 6.8: Keyboard Search

Chapter 7

Recent Research in Automation

These objectives have been pursued by researchers using a different strategy, leading to important developments. Much recent research has been done to improve the precision and effectiveness of automation testing for Android set-top boxes (STBs). These capabilities seek to reduce the money and time spent on testing procedures. The main goal of the research is to reduce the manual work required while keep thorough test coverage by automating the test case generation process. The automatic generation of test cases using machine learning algorithms and the creation of test prioritization strategies that lower the number of test cases required while retaining appropriate coverage are some of the primary topics of research in this field.

7.0.1 Integration with Cloud-based Testing Platforms:

Due to its affordability, and scalability, integration with cloud-based testing platforms is growing in famous in the area of automation testing for Android set-top boxes (STBs). The ideal integration of Android STB testing with these cloud-based platforms is a topic of active research. With the help of this connection, testers may run tests concurrently on a variety of devices, thus reducing testing time and expense.

7.0.2 Artificial Intelligence (AI) and Machine Learning (ML):

Researchers are look into the use of Machine Learning and AI to automate the test case generation process and increase testing accuracy. This requires employing techniques like deep learning to produce test cases that cover the most critical paths and scenarios analyze the application’s behavior.

7.0.3 Test Orchestration:

The management and coordination of the execution of several tests across diverse platforms and devices is test orchestration, a crucial component of automation testing. Researchers are continually looking into ways to automate the test orchestration process so that testers may run tests simultaneously on many devices. In addition to saving time, this automation lowers the overall cost of testing activities.

7.0.4 Integration with Continuous Integration/Continuous Deployment (CI/CD) pipelines:

This entails automating testing and integrating it with the development and deployment processes to guarantee that any faults are discovered early in the development cycle. Researchers are investigating approaches to connect automation testing for android STBs with CI/CD pipelines, which are extensively used for software development and deployment.

7.0.5 Test Automation Frameworks:

This required automating testing and integrating it with the development and deployment processes to guarantee that any faults are discovered early in the development cycle. Researchers are investigating approaches to connect automation testing for android STBs with CI/CD pipelines, which are extensively used for software development and deployment. Researchers are exploring ways to improve test automation frameworks for android STBs, enabling testers to write and execute tests more efficiently. This involves using techniques such as model-based testing and code generation to automate the test case generation process and reduce the time required for testing.

In conclusion, recent research in automation testing for android STBs has focused on improving the accuracy and efficiency of testing, reducing the cost and time required for testing, and integrating testing with development and deployment processes. These advances will help ensure the quality and reliability of android STBs and enable developers to deliver better products to their customers.

Chapter 8

Future Directions

These technologies can be used to improve the efficiency and accuracy of testing by minimizing the amount of manual work necessary for testing while also boosting the accuracy of test results. Further advance in machine learning and AI are anticipated to be involved in the future of automation testing for Android STBs. Moreover, continuing development of tools and frameworks for automation testing is inevitable, giving new approaches to test Android STBs. Trends such as growing usage of AI and ML, more powerful test automation frameworks, more attention on performance testing, integration with DevOps, and increased use of cloud-based testing are likely to effect the future of automation testing for android STBs.

8.0.1 More advanced test automation frameworks:

These frameworks will help testers develop and run tests more effectively and quickly, which will save them time and effort. Future test automation frameworks are anticipated to be more comprehensive and advanced, providing testers with more productivity and usability. Model-based testing, where testers build models that mimic the behavior and functionality of the Android set-top boxes (STBs), is one potential enhancement. to automatically generate test cases the manual work involved in creating test cases can be reduce by using these models.

8.0.2 Increased use of AI and ML:

Artificial intelligence (AI) and machine learning (ML), as was previously indicated, are currently used in automation testing for Android set-top boxes (STBs), and their use is anticipated to increase in the future. Due to the fact that AI and ML technologies have a lot to suggest in terms of improving testing findings' accuracy and decreasing the total time and cost involved in testing procedures.

8.0.3 More focus on performance testing:

As android STBs get more capable and powerful of running more difficult applications, performance testing will become increasingly crucial. This will procedures capable of accurately measuring the performance of these devices and necessitate the creation of new performance testing tools.

8.0.4 Increased use of cloud-based testing:

At the same time and reducing expense and testing time and will allow testers to conduct tests on different devices. As they become more scalable and beneficial cloud-based testing platforms are effect to grow more famous in the future.

8.0.5 Integration with DevOps:

It will require integrated and automated testing with the development and deployment processes to find requires early in the development cycle. Automation testing for Android STB with DevOps methods is likely to become more and more common in the future.

Chapter 9

Conclusion

The usage of automation testing has several advantages, including improved test coverage, greater testing productivity, and lower testing expenses. Automation testing is an efficient method for testing Android set-top boxes, and numerous tools and approaches have been created to help with it. Further advance in ML and AI are likely to provide new approaches to automate testing and increase the accuracy and efficiency of testing results in the future of automated testing for Android STBs. For software development in this industry automation testing for android STBs is a critical component. Automation testing methods and methodologies are continually expanding, and current research has focused on enhancing testing accuracy and efficiency, lowering testing costs and time, and integrating testing with development and deployment processes. Artificial Intelligent and machine learning are rapidly being utilised in android STB testing to automate the test case generating process and increase testing accuracy. CI/CD pipelines and Cloud-based testing platforms are also being integrated into the testing process, allowing testers to run tests on numerous devices at the same time, expense and decreasing testing time. As a result of these advances, android STB developers may provide better goods to their consumers, benefiting both the developers and the end-users..

Bibliography

[1] Saja Khalid Alferidah, Shakeel Ahmed. (2020). Automated Software Testing Tools.

[2] Joseph Bosas. (2018). Automated Testing Importance and Impact.

[3] Chen, L., and Liu. (2017). Development of Automated Test System for Android Set-top Box.

[4] Claus Klammer, Rudolf Ramler. (2017). A Journey from Manual Testing to Automated Test Generation in an Industry Project.

[5] Tian, Y., and Zhang. (2018). Performance Comparison of UI Automator and Appium for Android Set-top Box Testing.

[6] Fathima Naja Musthafa , Syeda Mansur , Adika Wibawanto. (2020). Automated Software Testing on Mobile Applications: A Review with Special Focus on Android-Platform.

[7] Elis Pelivani, Betim Cico. (2021). A comparative study of automation testing tools for web applications.

[8] Arundhatti Bezbaruah, Bhanu Pratap, Sandeep B Hake. (2020). Automation of Tests and Comparative Analysis between Manual and Automated testing.

[9] Lee, S., and Bae, D. (2016). An Evaluation of the Effectiveness of Android-based Test Automation Tools for a Set top Box.

[10] Kenish Rajesh Halani, Kavita, Rahul Saxena. (2021). Critical Analysis of Manual Versus Automation Testing.

[11] Yvan Labiche. (2018). Test Automation—Automation of What?

[12] Wang, K., Liu, J. (2018). Research on Test Automation Based on Behave Framework for Android Set-top Box.

[13] Wang Junmei, Wu Jihong. (2019). Research on Automation Testing Technology Based on Image Recognition.

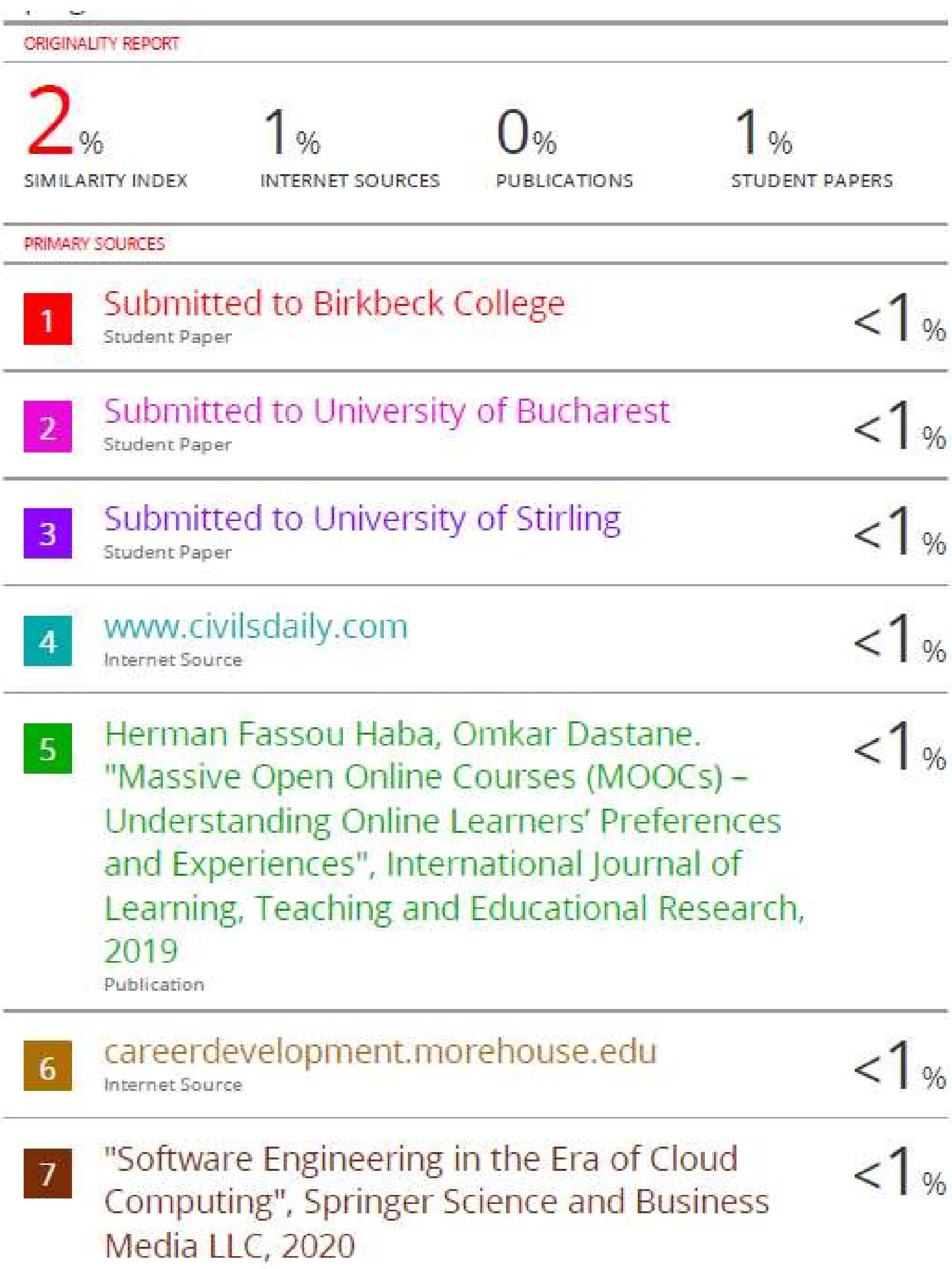
[14] Matija Pul, Mario Vranješ, Vukota Peković, Ratko Grbić. (2018). Automatic Functionality Verification of Hybrid Set-Top Boxes With Dynamic User Interface.

[15] Apeksha Deshpande,S.V Veenadevi, Suresh Aleti. (2021). Test Automation and Continuous Integration using Jenkins for Smart Card OS Interface.

[16] Sangeetha Yalamanchili, K. Sitha Kumari. (2017). Comparison of manual and automatic testing using genetic algorithm for information handling system.

[17] Kalyani V. Kumbhar, Deeplaxmi V. Niture. (2022). Development of Test Automation Scripts for Panel Logic module using Vector CANoe system.

[18] P. Nagarani, R. VenkataRamanaChary. (2012). A tool based approach for automation of GUI applications.

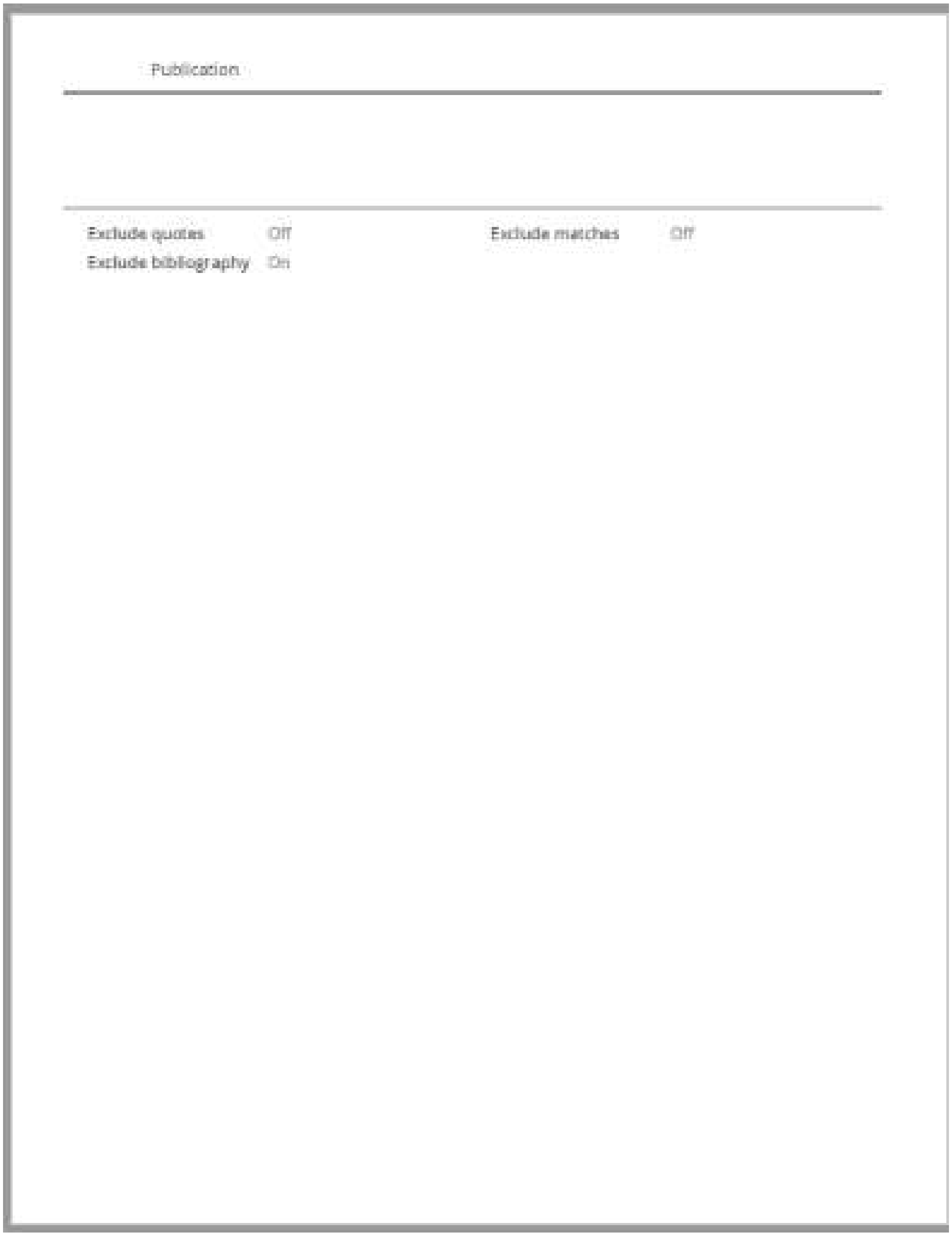


ORIGINALITY REPORT

2%	1%	0%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

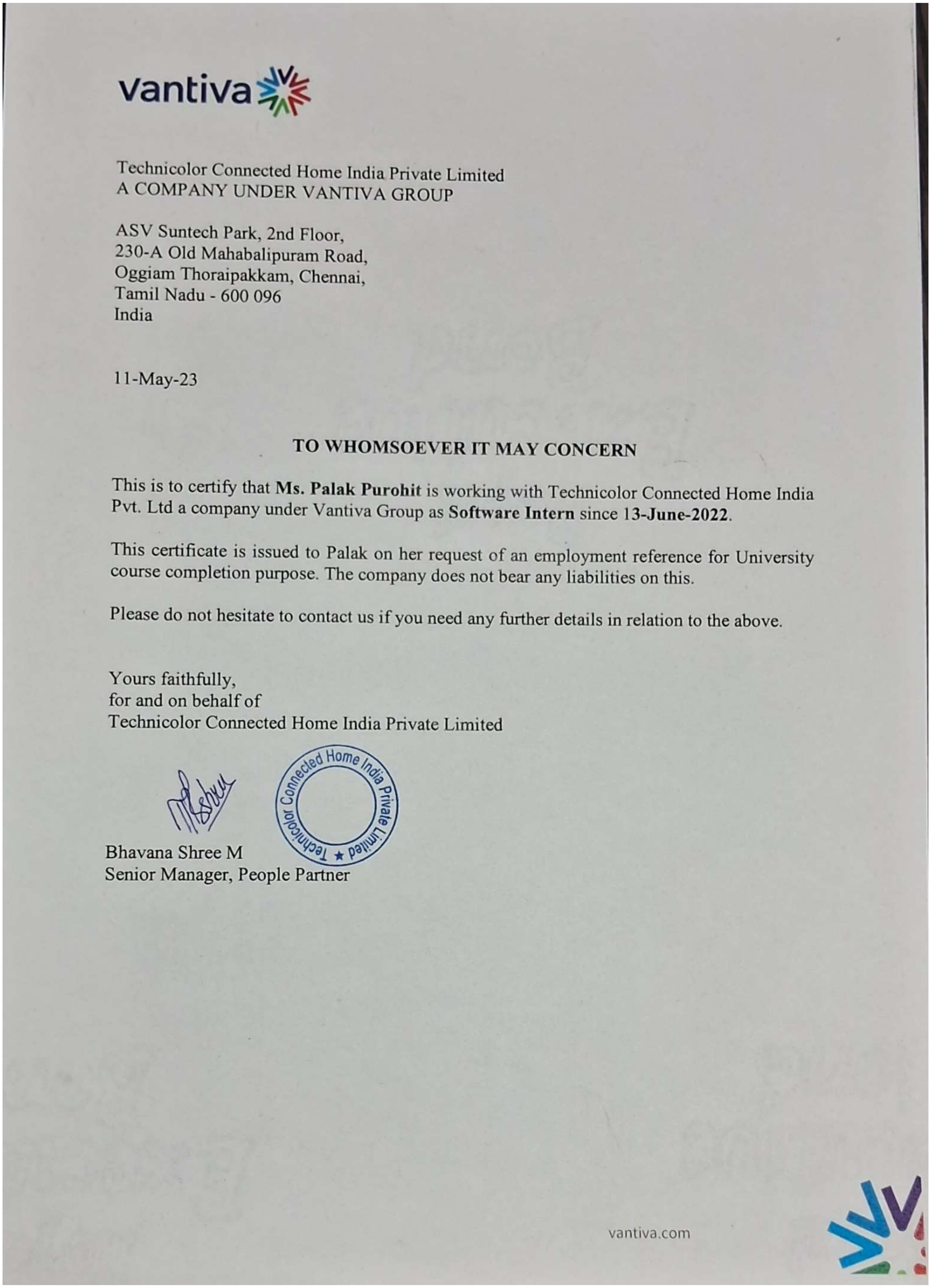
PRIMARY SOURCES

1	Submitted to Birkbeck College Student Paper	<1%
2	Submitted to University of Bucharest Student Paper	<1%
3	Submitted to University of Stirling Student Paper	<1%
4	www.civilsdaily.com Internet Source	<1%
5	Herman Fassou Haba, Omkar Dastane. "Massive Open Online Courses (MOOCs) – Understanding Online Learners' Preferences and Experiences", International Journal of Learning, Teaching and Educational Research, 2019 Publication	<1%
6	careerdevelopment.morehouse.edu Internet Source	<1%
7	"Software Engineering in the Era of Cloud Computing", Springer Science and Business Media LLC, 2020	<1%



Publication

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		



vantiva

Technicolor Connected Home India Private Limited
A COMPANY UNDER VANTIVA GROUP

ASV Suntech Park, 2nd Floor,
230-A Old Mahabalipuram Road,
Oggiam Thoraipakkam, Chennai,
Tamil Nadu - 600 096
India

11-May-23

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Ms. Palak Purohit** is working with Technicolor Connected Home India Pvt. Ltd a company under Vantiva Group as **Software Intern** since 1**3-June-2022**.

This certificate is issued to Palak on her request of an employment reference for University course completion purpose. The company does not bear any liabilities on this.

Please do not hesitate to contact us if you need any further details in relation to the above.

Yours faithfully,
for and on behalf of
Technicolor Connected Home India Private Limited

Technicolor Connected Home India Private Limited ★

Bhavana Shree M
Senior Manager, People Partner

vantiva.com