F-048

Institut Sains dan Teknologi Terpadu Surabaya

Jalan Ngagel Jaya Tengah 73 – 77, Surabaya 60284 , Indonesia
Telp. (031) 5027920 Fax. (031) 5041509

PROPOSAL TUGAS AKHIR (TA)

Periode Bulan: April Tahun: 2022
Semester ~~Gasal~~ / Genap *) Tahun ajaran 2021 / 2022

Program / Program Studi : ~~D3~~ / S1 *) Informatika
Nama Mahasiswa : Yoshua Dwi Santoso
NRP Mahasiswa : 218116775
Bidang Keahlian (Major) : Software Technology
Sekaligus menjadi major pilihan, dan kesalahan pengisian major mengakibatkan GAGAL Tugas Akhir dan Yudisium
Judul Tugas Akhir :

Studi Pengkajian Automated Mobile App Testing Menggunakan Appium

Jenis Tugas Akhir : ☐ Hardware ☐ Software ☑ Studi ~~Literatur~~ / Pengkajian / ~~Analisa~~ *)
***) Coret yang tidak perlu**
Pembimbing Utama : Ir. Edwin Pramana, M.AppSc., Ph.D.
Co. Pembimbing :
Jumlah SKS **SUDAH** LULUS : 136 SKS IPK : 3.45 ECC Level : 4

Mengetahui,

Pembimbing Utama,	Co. Pembimbing,	Surabaya, 18 April 2022 Pemohon,
(Edwin Pramana, Ph.D.)	( )	(Yoshua Dwi Santoso)

Catatan Tambahan:

Menyetujui,

Dekan,	Ketua Program Studi,
( )	( )

Hasil Keputusan Ketua Program Studi
Periode APRIL 2022

Informasi Tugas Akhir/Tesis

NRP	: 218116775
Nama	: YOSHUA DWI SANTOSO
Judul	: Studi Pengkajian Automated Mobile App Testing Menggunakan Appium
Judul Baru	: Studi Pengkajian Automated Mobile App Testing Menggunakan Appium
Pembimbing	: Edwin Pramana, Ir., M.AppSc., Ph.D
Co-Pembimbing	:

Informasi Periode Tugas Akhir/Tesis

Tanggal Cetak : 18 April 2022, 03:35

Hasil Revisi

Judul Revisi	:
Kaprodi	
Syarat	:

Hasil Keputusan Sidang Proposal
Periode APRIL 2022

Informasi Tugas Akhir/Tesis

NRP	: 218116775
Nama	: YOSHUA DWI SANTOSO
Judul	: Studi Pengkajian Automated Mobile App Testing Menggunakan Appium
Judul Baru	: Studi Pengkajian Automated Mobile App Testing Menggunakan Appium
Pembimbing	: Edwin Pramana, Ir., M.AppSc., Ph.D
Co-Pembimbing	:

Informasi Periode Tugas Akhir/Tesis

Tanggal Sidang Proposal	: 6 April 2022
Tanggal Cetak	: 18 April 2022, 03:35

Review

Reviewer 1	: Evan Kusuma Susanto, S.Kom.
Status	: **DIPERBAIKI**
Pesan	: 1. Pastikan syarat studi pengkajian terpenuhi (80% fitur dicoba). 2. Karena salah satu keunggulan Appium dibandingkan dengan program lain pada tabel perbandingan adalah kemampuan testing berbagai jenis platform, ada baiknya agar dicoba lebih dari 1 OS, tidak hanya android saja 3. Skenario automated testing yang direncanakan cukup sedikit, pastikan sebagian besar fitur2 Appium dapat dicoba pada skenario2 tersebut. Mungkin ada baiknya juga apabila app bahan uji coba berbeda jenis (tidak hanya aplikasi note taking) atau mungkin dapat digunakan juga aplikasi sederhana buatan sendiri untuk menunjukkan peran Appium dalam membantu development aplikasi.
Tanggapan	:

1. Pada bagian Tujuan telah diperbaiki bahwa studi pengkajian ini akan membahas 80% fitur dari Appium dan fitur-fitur yang akan dicoba tertera pada lampiran.

2. Telah ditambahkan aplikasi berplatform IOS pada skenario uji coba untuk mencoba fitur Appium pada platform IOS. (Hal 9)

3. Jenis aplikasi yang digunakan telah diganti dengan aplikasi game sederhana Sejarah-kita , aplikasi hibrid Webview App , dan Addmodulo pada IOS (hal 9)

Campus iSTTS: Ngagel Jaya Tengah 73-77, Surabaya 60284 P.O.Box 1589, Indonesia
Telp. **(031) 5027920** (Hunting) Fax. **(031) 5041509**
Web: **www.istts.ac.id**

.....

.....

.....

.....

.....

.....

Reviewer 2 : Lukman Zaman P. C. S. W., S.Kom., M.Kom.

Status : **DIPERBAIKI**

Pesan : Jangan jadi manual atau tutorial Appium.
Bagian input output tidak jelas. Laporan seperti apa yang bisa diharapkan di kasus yang dipilih?
Kenapa tidak untuk melakukan test automasi pada program sendiri? Misalnya, saya punya ratusan program novel interaktif, dan saya ingin apakah pemilihan semua cabang bisa berjalan dengan baik, atau crash di salah satu cabang. Apa bisa melakukan hal tsb?

Tanggapan :

1. Pada bagian tujuan telah diperbaiki bahwa studi pengkajian ini akan membahas setidaknya 80% fitur dari Appium yang tertera pada lampiran. (hal2 2)

2. untuk contoh script telah diperbaiki pada bab Ruang Lingkup subbab input dan output dengan contoh penggunaan salah satu fitur milik Appium yaitu currectActivity (hal 7)

.....

.....

.....

.....

.....

.....

.....

.....

.....

Reviewer 3 : Suhatati Tjandra, Ir., M.Kom.

Status : **DIPERBAIKI**

Campus iSTTS: Ngagel Jaya Tengah 73-77, Surabaya 60284 P.O.Box 1589, Indonesia
Telp. **(031) 5027920** (Hunting) Fax. **(031) 5041509**
Web: **www.istts.ac.id**

Pesan : 1. Tujuan:"untuk menjelaskan fitur-fitur testing dari Appium". Jelaskan apa yang akan dianalisa, jangan hanya untuk menjelaskan "how to use".
2. Pastikan yang dibahas meliputi minimal 80% fitur
3. perhatikan pembahasan dan penyajian hasil uji cobanya di buku tugas akhir (karena bobot terbesar ada di buku)

Tanggapan :

1. Pada bagian Tujuan telah diperbaiki bahwa studi pengkajian ini akan membahas 80% fitur dari Appium dan akan membahas fitur-fitur Appium yang terdapat pada lampiran.
(hal 2)

2. Pada tugas akhir akan membahas setidaknya 80% dari fitur Appium.

3. pada tugas akhir ini akan memfokuskan pada penyajian dan pembahasan fitur uji coba milik Appium

Reviewer 4 : Kevin Setiono, S.Kom.

Status : **DIPERBAIKI**

Pesan : 1. Tambahkan website STTS untuk uji coba
2. Tuliskan fitur apa saja yang akan dicoba, dan 80% fitur dicoba

Tanggapan :

1. pada studi pengkajian ini berfokus pada testing di Mobile App.

2. Pada bagian Tujuan telah diperbaiki bahwa studi pengkajian ini akan membahas 80% fitur dari Appium dan fitur-fitur yang akan dicoba terdapat pada bagian lampiran.

Campus iSTTS: Ngagel Jaya Tengah 73-77, Surabaya 60284 P.O.Box 1589, Indonesia
Telp. **(031) 5027920** (Hunting) Fax. **(031) 5041509**
Web: **www.istts.ac.id**

.....
.....
.....
.....
.....

Reviewer 5 : Iwan Chandra, S.Kom., M.Kom.
Status : **DITERIMA**
Pesan : 1. Pastikan pembahasan tentang Appium mencakup paling tidak 90% dari keseluruhan fitur yang ada pada Appium
Tanggapan :

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Hasil Sidang Proposal

Status : **DIPERBAIKI**
Syarat : 1. 80% Fitur dari Appium Wajib dicoba dan dibahas
2. Pada Laporan Buku TA tidak boleh menjadi buku manual/tutorial Appium. Wajib dilengkapi analisa dan hasilnya
3. Tambahkan contoh Input dan Output dari kasus yang dijalankan pada proposal
4. Tambahkan program lain untuk uji coba
Tanggapan :

.....

Campus iSTTS: Ngagel Jaya Tengah 73-77, Surabaya 60284 P.O.Box 1589, Indonesia
Telp. **(031) 5027920** (Hunting) Fax. **(031) 5041509**
Web: **www.istts.ac.id**

1. pada tugas akhir ini akan membahas setidaknya 80% fitur dari Appium. (Hal 2)

2. Tugas akhir ini akan berfokus membahas fitur-fitur dari Appium dengan analisa dan hasilnya. (hal 8 , lampiran)

3. Contoh input dan output dari program telah ditambahkan pada bagian Ruang lingkup subbab input dan output (Hal 7)

4. untuk melakukan pengkajian Appium telah mengganti program dengan aplikasi Sejarah-kita(Native) , Webview App(Hibrid Android) , AddModulo(IOS) ... (hal 9)

Saran : Harus diusahakan untuk mencoba pada perangkat IOS

Tanggapan :

untuk pengujian pada perangkat IOS kemungkinan dapat dijalankan karna sudah mendapat perangkat dan file .IPA pada untuk dilakukan testing.

Campus iSTTS: Ngagel Jaya Tengah 73-77, Surabaya 60284 P.O.Box 1589, Indonesia
Telp. **(031) 5027920** (Hunting) Fax. **(031) 5041509**
Web: **www.istts.ac.id**

PROPOSAL TUGAS AKHIR
STUDI PENGKAJIAN AUTOMATED MOBILE APP TESTING MENGGUNAKAN APPIUM

Nama	: Yoshua Dwi Santoso
NRP	: 218116775
Jurusan/ Prodi/ Major	: Informatika / S1 / Software Engineering
Dosen Pembimbing	: Ir. EDWIN PRAMANA, M.AppSc., Ph.D.

I. Latar Belakang

Pada pembuatan aplikasi perlu sekali dilakukan testing atau quality control. Fungsi dari testing atau quality control adalah untuk mencari bug dan memastikan kualitas dari program sesuai dengan kebutuhan penggunanya. Tidak bisa dipungkiri dalam pembuatan sebuah aplikasi tidak lepas dari yang namanya bug atau error. Bug atau error sebisa mungkin ditemukan sedini mungkin agar pengembang dapat melakukan pembenahan program tersebut. Karena bisa jadi bahaya apabila suatu program terjadi error atau bug dalam sistem yang bersifat penting seperti contohnya melakukan transaksi. Setelah dilakukan testing dan bila menemukan bug atau ketidak sesuaian , maka aplikasi tersebut akan dibenahi oleh pengembang agar kualitas dari aplikasi yang dibuat sesuai dengan kebutuhan pengguna.

Dulu testing dilakukan oleh manual oleh manusia dengan mencobai tiap sistem dan semua kemungkinan yang ada. Jika terjadi perubahan dalam suatu sistem , aplikasi tersebut harus dilakukan testing ulang. Tentunya melakukan hal yang sama dan dikerjakan berkali-kali dapat melelahkan dan membuang banyak waktu. Namun dengan menggunakan automated testing tools, dapat membantu mengefisiensi proses testing pada aplikasi.

Automated testing tools adalah aplikasi yang dapat menjalankan perintah untuk melakukan testing secara otomatis. Karena dijalankan secara otomatis, maka kesalahan tester seperti melewatkan test case bisa diminimalisir dan kegiatan testing dapat dilakukan dengan waktu yang cukup singkat.

Appium adalah salah sattu contoh testing tools untuk perangkat Android dan iOS. Appium bersifat open source jadi bisa digunakan secara gratis. Appium dapat melakukan pengujian di aplikasi web seluler, aplikasi hibrid di android dan IOS, dan aplikasi native. Appium juga mendukung banyak bahasa pemrograman seperti Java, Ruby, PHP , Python , C# , dan masih banyak lagi. Sehingga tester bisa leluasa meimilih bahasa mana yang menjadi acuan dalam penulisan script.

II. Tujuan

Tujuan dari dibuatnya Studi Pengkajian Menggunakan Appium untuk menjelaskan fitur-fitur testing dari Appium. Dalam tugas akhir ini setidaknya akan menganalisa 80% dari fitur yang dimiliki oleh Appium. Appium dapat digunakan untuk melakukan otomasi testing mobile app yang bersifat native atau hibrid pada platform IOS atau Android. Untuk melakukan hal tersebut maka akan dilakukan automated testing aplikasi mobile native pada Android aplikasi native pada IOS , dan aplikasi hibrid pada Android.

III. Teori Penunjang

Berikut adalah beberapa teori yang menunjang studi pengkajian mobile app automation testing :

1. Automated Testing

Automated testing adalah teknik melakukan testing yang dieksekusi oleh sebuah automation tools dalam pembahasan ini tools yang digunakan adalah Appium. Automated testing dilakukan dengan menuliskan script testing yang akan dieksekusi nantinya dan membandingkannya dengan hasil yang diprediksi. Dengan automated testing , tahap testing bisa dilakukan dengan cepat dan menghemat banyak waktu development.

2. Java

Dalam pembuatan aplikasi berbasis android , ada banyak bahasa pemrograman yang bisa digunakan tetapi yang masih populer hingga saat ini adalah bahasa pemrograman Java. Dalam penulisan script testing pada Appium dapat menggunakan Java.

3. Aplikasi Native

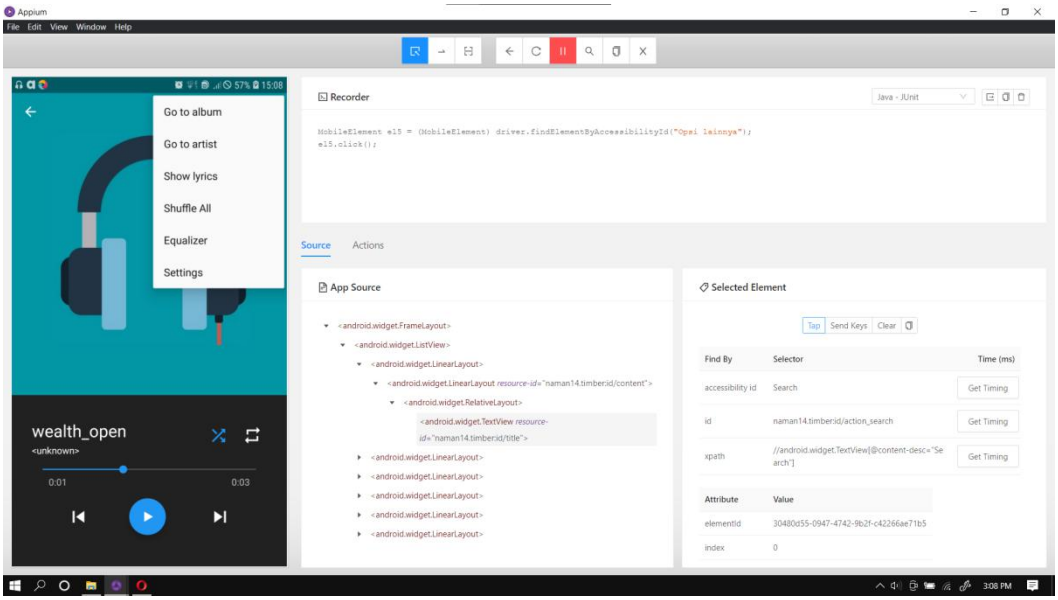
Aplikasi Native adalah aplikasi yang dibangun dengan bahasa pemrograman yang spesifik dan hanya digunakan di platform tertentu. Platform yang populer saat ini adalah iOS dan Android. Untuk membuat aplikasi yang di dua platform yang berbeda , dibutuhkan bahasa pemrograman yang berbeda juga. Sebagai contoh untuk aplikasi di platform iOS dapat digunakan bahasa pemrograman Swift dan Objective-C lalu untuk platform android dapat digunakan Java , Kotlin , dan Flutter. Biasanya aplikasi native harus diunduh di Marketplace Aplikasi seperti Google play untuk Android dan App Store untuk iOS.

4. Aplikasi Hybrid

Aplikasi Hybrid adalah aplikasi web yang ditransformasikan menjadi kode native pada platform Android atau iOS. Aplikasi hybrid biasanya menggunakan browser untuk mengijinkan aplikasi web mengakses fitur dari mobile device seperti Push Notification , Contacts, dan Offline Data Storage. Untuk keperluan cross platform , aplikasi hybrid dapat dikembangkan lebih cepat dan menghemat biaya daripada aplikasi native.

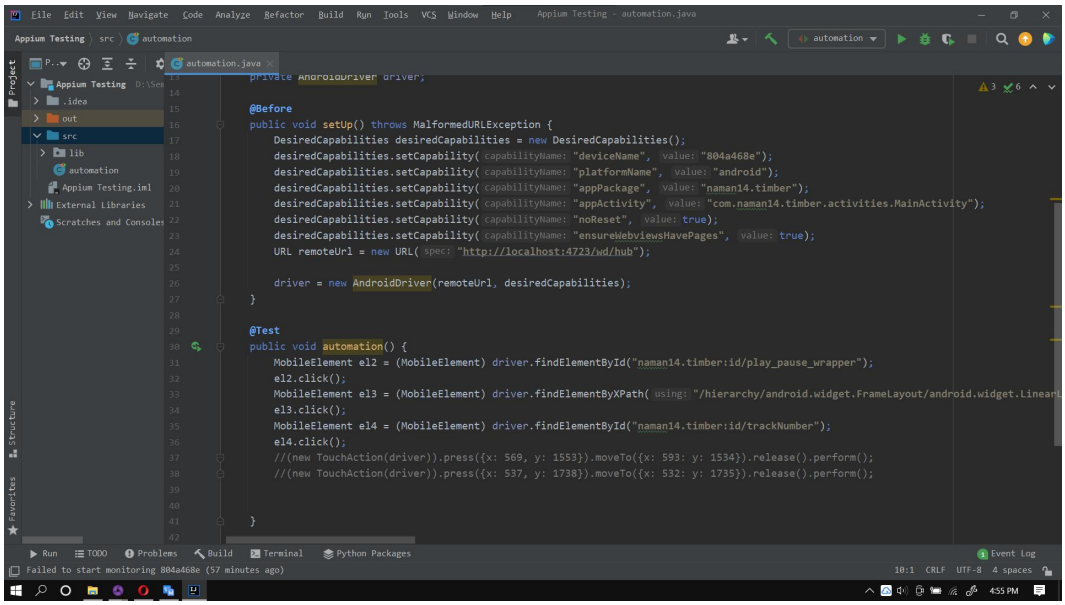
5. Appium Desktop

Appium desktop adalah automation tools yang mensupport testing pada platform android dan iOS. Dalam pembuatan scriptnya, Appium bersifat "cross-platform" yang artinya dapat melakukan eksekusi testing dalam android dan iOS dengan menggunakan API yang sama. Bahasa pemrograman yang digunakan untuk melakukan penulisan script testing pada Appium beragam contohnya Java, PHP , Python , Ruby, dan masih banyak lagi. Dalam tugas akhir bahasa pemrograman yang digunakan adalah java. Untuk penulisan scriptnya dibutuhkan IDE untuk membantu mempermudah penulisan script. Appium Desktop terbatas hanya untuk melakukan perekaman pada perangkat asli atau dengan emulator. Berikut adalah contoh dari tampilan Appium Desktop :



Gambar 3.1
Tampilan Appium Dekstop

Gambar 3.1 adalah contoh tampilan dari Appium Desktop. Di Appium Desktop tester pelakukan perekaman bagaimana aplikasi dijalankan lalu Appium akan memberi source code tersebut. Source code ini lah yang dapat diolah menjadi beberapa test case lain dalam IDE. Berikut adalah contoh penulisan test case dalam IDE :



Gambar 3.2
Script Appium

Gambar 3.2 adalah contoh script Appium yang dipindah pada IDE Intelij. Di script tersebut tester dapat menambahkan test case lain seperti menambah data dummy sebelum test dijalankan. Dengan begitu pengujian dalam sebuah aplikasi dapat mencakup banyak fitur.

Langkah awal penggunaan Appium untuk otomasi testing adalah mengaktifka server Appium melalui Apium desktop atau melalui Command Prompt. Lalu pada IDE hanya perlu memasukan URL dimana server appium berjalan. Setelah itu tester menulis script otomasi testing pada IDE dan Appium akn menjalankan perntah otomasi yang ditulis pada emulator atau pada perangkat yang terhubung.

IV. Ruang Lingkup

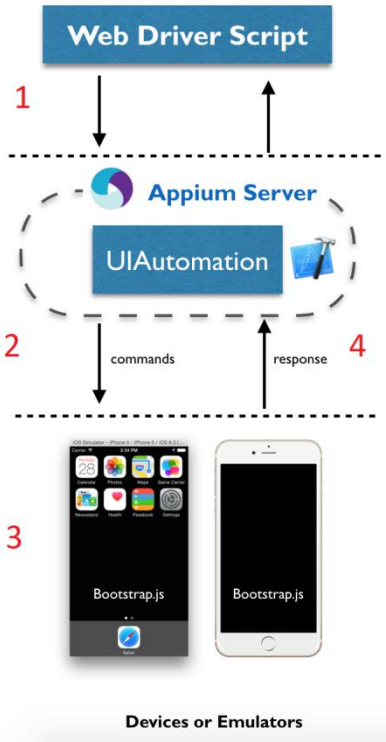
Proposal tugas akhir ini membahas fitur-fitur dari automated testing tools Appium dengan melakukan testing pada aplikasi mobile. Ruang lingkup yang akan dibahas pada proposal tugas akihr adalah sebagai berikut :

1. Arsitektur Sistem

Dalam tugas akhir ini fokus mengkaji fitur dari automated testing tools Appium. Appium pada intinya adalah server web yang mengekspos REST API. Appium menerima koneksi dari klien, menerima perintah, menjalankan perintah tersebut di perangkat seluler yang terhubung, dan merespons dengan respons HTTP yang mewakili hasil eksekusi perintah. Appium memiliki arsitektur klien/server. Tester dapat menulis kode pengujian dalam bahasa apa pun yang memiliki API klien HTTP.

Terdapat dua cara untuk melakukan otomasi testing dengan Appium. Yang pertama dengan merekam sentuhan pada decive dan hasil rekaman tersebut akan menjadi script yang digenerate oleh Appium. Script yang telah digenerate oleh Appium di copy pada IDE untuk diolah lebih lanjut. Lalu script akan dieksekusi ulang untuk melakukan testing. Tetapi cara ini memliki keterbatasan yaitu hanya dapat menggunakan beberapa method bawaan dari Appium.

Cara yang kedua yaitu dengan mengetik script menggunakan bahasa pemrograman yang didukung. Mirip seperti cara pertama , script tersebut akan dieksekusi oleh appium untuk mengotomasi testing pada perangkat. Dengan menuliskan script dari awal , cara ini memungkinkan tester menggunakan seluruh functionalitas dari method Appium yang tersedia



[1]

[1]Sumber :https://subscription.packtpub.com/book/application_development/9781787280168/1/ch01lvl1sec11/appium-architecture

Gambar 3.3
Arsitektur Sistem Appium

Web Driver Script adalah script yang digunakan untuk menjalankan otomasi pada device (Gambar 3.3). Selanjutnya Web Driver script tersebut akan dijalankan pada IDE dan melalui Appium Server. Cara kerja Appium Server mirip dengan Server milik Selenium , Appium akan menunggu koneksi dari client dan menjalankan perintah dari Web Driver Script (Gambar 3.3 nomor 2). Selanjutnya eksekusi akan dijalan kan pada device atau emulator yang terhubung dengan Appium Server (Gambar 3.3 nomor3). Lalu hasil dari eksekusi test tersebut akan menghasilkan logging yang dapat ditampilkan pada IDE (Gambar 3.3 nomor 4).

2. Input dan Output

Pada tugas akhir automated testing dibutuhkan input untuk mendapatkan output yang diiginkan. Berikut adalah input yang dibutuhkan dan output yang diprediksi pada pembuatan automated testing :

1. Script

Script testing didapat dengan cara melakukan record pada aplikasi lalu mengcopy code yang dihasilkan oleh Appium atau diketik secara manual. Script ini digunakan untuk eksekusi otomasi pada aplikasi. Berikut adalah contoh script untuk automate testing :

```
@Test()
public void cekCurrentActivity() throws MalformedURLException {

    /* contoh untuk mengecek sudah berpindah ke activity Intro ke MainActivity */
    homePageOmniNotes = new HomePageOmniNotes((AndroidDriver) driver);
    String currentActivity = ((AndroidDriver<?>) driver).currentActivity();
    if(currentActivity.equalsIgnoreCase( anotherString: ".intro.IntroActivity")){
        skipWelcome();
    }

    currentActivity = ((AndroidDriver<?>) driver).currentActivity();
    Assert.assertEquals(currentActivity, s1: ".MainActivity");
}
```

Gambar 3.4
Script testing

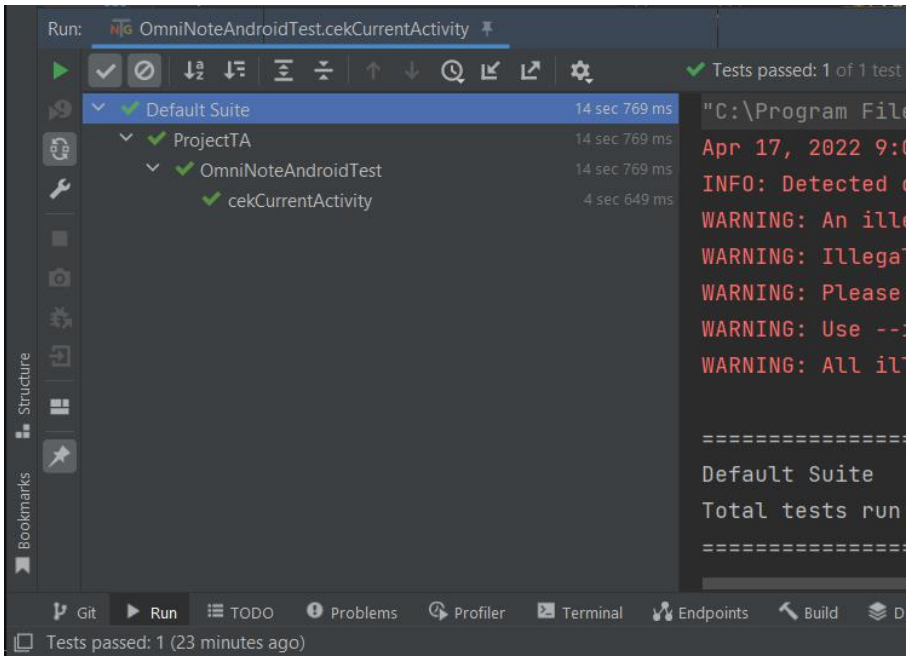
Gambar diatas adalah contoh script untuk testing. Script diatas adalah contoh penggunaan menggunakan syntax “currentActivity()” pada class Android Driver untuk mendapat activity yang sedang berjalan pada device atau emulator.

2. Test Case

Test case adalah serangkaian script untuk menentukan apakah suatu hal bekerja semestinya. Tester dapat membuat sebuah test case dan menggabungkanya dengan beberapa test case lain untuk menghemat waktu penulisan script. Test case utama dari tugas akhir ini adalah memastikan semua fitur dari aplikasi yang di test berjalan dengan semestinya.

3. Test Result log

Test Result Log adalah artefak penting selama pengujian yang berisi ringkasan terperinci dari keseluruhan uji coba dan menunjkukan tes mana saja yang lulus dan gagal. Test Result Log juga berisi detail dan informasi tentang berbagai pengujian termasuk sumber masalah dan alasan test tersebut gagal. Fokus dari artefak ini adalah memungkinkan diagnosis kegagalan dan cacat pasca eksekusi pada aplikasi.



Gambar 3.5
Contoh test log

Gambar 3.5 adalah contoh test log dari test untuk mengecek currentActivity pada gambar 3.4. untuk melakukan Assertion , diperlukan library tambahan untuk testing yaitu TestNG atau Junit untuk Java. Gambar 3.5 menjelaskan bahwa test cekCurrectActivity sukses.

3. Method Appium

Berikut akan dijelaskan tentang method-method yang dimiliki oleh Appium untuk melakukan automation testing pada mobile apps. Dalam melakukan automation testing , diperlukan driver untuk menjalankan command yang akan dikirimkan ke server Appium.

Appium memliki sebuah class driver yaitu AppiumDriver dan diturunkan menjadi tiga driver yang lebih spesifik yaitu Android Driver, IOS Driver , dan Windows Driver. Sesuai namanya tiap driver turunan Appium Driver berfungsi untuk mengirim command ke device sesuai dengan platformnya. Berikut adalah method-method dari tiap driver :

A. Appium Driver

Merupakan superclass dari Android Driver. Appium Driver memiliki method-method yang sifatnya umum sehingga dapat digunakan oleh class turunannya. Daftar method yang dimliki oleh Appium Driver dapat dilihat pada lampiran 1.

B. Android Driver

Android Driver merupakan class turunan dari Appium Driver. Android Driver memiliki method yang lebih spesifik untuk melakukan automated testing menggunakan Appium pada device dengan platform Android. Method-method adari Android Driver dapat dilihat pada lampiran 2.

C. IOS Driver

Merupakan class turunan dari Appium driver yang ditujukan untuk otomasi pada aplikasi berbasis IOS. IOSDriver memiliki method-method yang lebih spesifik untuk aplikasi IOS. Method-method dari IOSDriver dapat dilihat pada Lampiran 3.

4. Batasan

Pada pembuatan Tugas akhir Studi Pengkajian Automated Mobile Testing dengan Appium ini terdapat batasan-batasan yang diberikan. Berikut adalah batasan-batasan yang diberikan :

1. Uji coba berfokus membahas 80% fitur-fitur yang dimiliki Appium.
2. Uji coba hanya dilakukan pada aplikasi Native dan Hybrid di platform Android dan IOS.

5. Perbandingan fitur

Appium tentu memiliki beberapa kekurangan dan kelebihan dengan automation testing tools yang lain. Berikut adalah perbandingan Appium dengan beberapa Mobile automation tools yang lain :

	Appium	**XCUITest**	**Espresso**
Tipe Aplikasi	Mobile Web, Native/Hybrid Mobile	Native/Hybrid Mobile Apps	Native/Hybrid Mobile Apps
Sistem Operasi	Cross-Platform	iOS	Android
Bahasa Pemrograman	Hampir semua bahasa pemrograman	Objective-C / Swift	Java/Kotlin
Tipe Testing	Black box	Gray Box	Gray Box
Butuh Source Code	Tidak	Ya	Ya

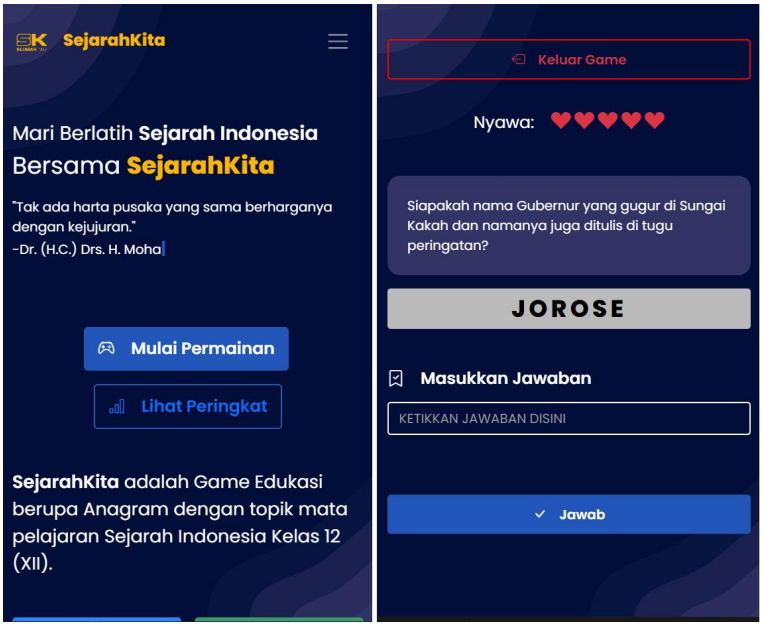
Tabel 4.3
Perbandingan Tools Appium

6. Skenario uji coba

Pada tugas akhir ini ada skenario yang akan dilakukan dengan beberapa aplikasi mobile. Aplikasi yang akan dilakukan testing yang akan dilakukan pada tugas akhir Automated Mobile Testing dengan Appium adalah Sejarah-kita, WebviewApp , dan AddModulo. Berikut adalah penjelasan singkat dari aplikasi yamg akan di ujicoba :

1. Sejarah-Kita (Android)

   Sejarah-kita adalah game sederhana tentang sejarah indonesia untuk anak SMA. Aplikasi Sejarah-kita merupakan aplikasi native android yang dibangun menggunakan bahasa java. Untuk saat ini aplikasi sejarah kita belum tersedia di PlayStore dan hanya bisa diakses melalui APK atau build pada Android studio.

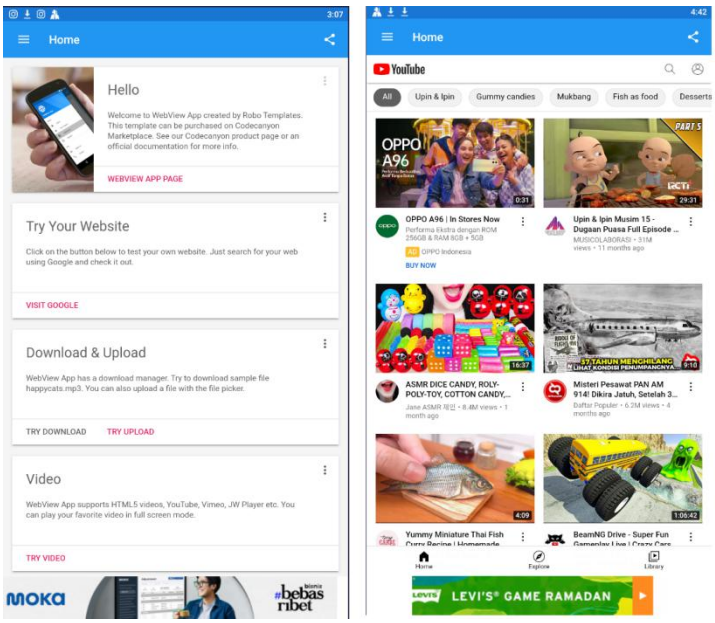


Gambar 4. 1
Contoh tampilan Sejarah-kita

Gambar 4.1 adalah contoh halaman setelah login dan halaman saat bermain. Untuk menggunakan aplikasi Sejarah-kita user perlu login terlebih dahulu. Lalu pada halaman dashboard user dapat memilih tingkat kesulitan game yang akan dimainkan setelah mengeklik tombol mulai bermain. Fitur yang dicoba untuk menguji fitur Appium mulai dari login , memainkan game berdasarkan tingkat kesulitan, dan edit profile.

2. Webview App(Hibrid Android)

Aplikasi webview App adalah aplikasi demo yang bersifat hibrid. Pada aplikasi ini dibagun menggunakan HTML yang dibungkus WebView oleh Android. Aplikasi Webview App dapat di download pada playstore dan bersifat gratis. Fitur dalam aplikasi ini adalah membuka browser youtube, dan google map secara internal, membuka Toast , SnackBar dan Dialog.



Gambar 4.2
Contoh tampilan Webview App.

Gambar 4.2 adalah tampilan home dari Webview App dan salah satu fiturnya untuk membuka youtube secara internal. Fitur yang dicoba nanti adalah buka website internal , open map internal , membuka web browser secara internal dan beberapa fitur lainnya.

3. App Modulo(IOS Native)

App Modulo adalah aplikasi game matematika puzzle sederhana pada IOS. User akan memainkan game ini melawan user lain. Cara memainkan game ini cukup sederhana user harus mengeliminasi tangan lawan dengan mendistribusikan angka ke tangan lawan sehingga angka pada tangan lawan menjadi nol. Game selesai bila angka pada kedua tangan lawan adalah nol. Aplikasi Add Modulo diakses dengan menggunakan file .ipa.

Gambar 4.3
Contoh tampilan Add Modulo

Gambar 4.3 adalah contoh tampilan home dan gameplay dari Add Modulo. Fitur yang akan dicoba adalah membuka halaman ‘about’ dan ‘how to play’dan mencoba pada gameplay dari Add modulo. Melakukan test bagaimana kondisi user menang atau kalah.

V. Tahapan Penyelesaian Tugas Akhir

Pada bab ini akann dijelaskan tahap penyelesaian Tugas akhir mulai dari awal pengerjaan hingga akhir . Berikut adalah tahapan dalam pengerjaan Tugas Akhir :

1. Mepelajari tools yang akan digunakan untuk automated testing.
2. Mencari Aplikasi yang menjadi objek testing.
3. Mempelajari struktur dan fungsionalitas dari aplikasi tersebut.
4. Merencanakan test case pada uji coba yang akan dilakukan pada aplikasi mobile yang dipilih.
5. Menganalisa fitur Appium yang dapat dicoba pada aplikasi yang dipilih.
6. Menulis script , menentukan input dan prediksi output dalam aplikasi mobile yang dipilih.
7. Melakukan eksekusi script testing pada aplikasi mobile.

VI. Daftar Pustaka

1. Anonymous, 2017, *Introduction to Appium*[Online] Available at : https://www.Appium.io [Accessed 5 Mei 2021]

2. Devopedia , 2018 , *Types of Mobile Apps*[Online] Available at : https://devopedia.org/types-of-mobile-apps [Accessed 10 Desember 2021]

3. Anonymous,2018,*Test Result*[Online] Available at: https://www.professionalqa.com/test-results [Accessed 11 Desember 2021]

4. Roy Nuriel , 2017, *XCUItest vs Appium vs Espresso* [Online] Available at : https://www.perfecto.io/blog/rise-espresso-xcuitest-fall-appium [Accessed 2 Januari 2022]

5. Thomas Hamilton ,2022 , *Interrupt Testing in Mobile Application*[Online] Available at : https://www.guru99.com/interrupt-testing.html [Accessed 2 Januari 2022]

6. Thomas Hamilton,2022, *Mobile App Performance Testing:Checklist and Tools*[Online] Available at : https://appiumpro.com/editions/102-mobile-app-performance-testing [Accessed 5 Januari 2022]

Lampiran 1
Method-method Appium Driver

NO	Method	Deskripsi
1	getStatus();	Mendapat status server yang sedang dijalankan
2	executeScript();	Menjalankan perintah aplikasi native
3	AppiumDriver();	Menjalankan server Appium dengan membuat instace AppiumDriver
4	driver.quit();	menghentikan session appium
5	getSessionDetails();	Mendapat capabilities dari session yang dijalankan
6	driver.back();	Menavigasikan mundur pada browser history bila memungkinkan (hanya untuk web)
7	getScreenshotAs();	Mendapat screeenshot halaman
8	getPageSource();	Mendapat current aplication hierarchy XML untuk native APP atau page sourve untuk WEB
9	manage().timeouts().pageLoadTimeout();	Mengatur waktu yang dapat dijalankan oleh operasi tertentu sebelum dibatalkan
10	manage().timeouts().implicitlyWait();	Mengatur jumlah waktu yang harus ditunggu saat mencari elemen
11	manage().timeouts().setScriptTimeout();	Tetapkan jumlah waktu, dalam milidetik, skrip asinkron yang dieksekusi oleh eksekusi async diizinkan untuk berjalan sebelum dibatalkan (khusus konteks web)
12	getOrientation();	Mendapat orientasi device atau browser
13	rotate();	Mengatur orientasi device atau browser
14	location();	untuk mendapat lokasi geografis (hanya untiuk driver yang mengimplentasi LocationContext)
15	getAvailableLogTypes();	Dapatkan jenis log yang tersedia sebagai daftar string
16	manage().logs().get();	Dapatkan log untuk jenis log tertentu. Buffer log

		diatur ulang setelah setiap permintaan
17	logEvent();	Menyimpan custom event
18	getEvents();	Mendapat event yang disimpan di Appium Server
19	getSettings();	Mendapat pengaturan dari device yang digunakan
20	setSetting();	Mengupdate pengaturan dari device yang digunakan
21	executeDriverScript();	jalankan skrip WebdriverIO terhadap sesi saat ini, memungkinkan eksekusi banyak perintah dalam satu permintaan Appium.
22	startActivity();	Mulai aktivitas Android dengan memberikan nama paket dan nama aktivitas
23	currentActivity();	Dapatkan nama aktivitas Android saat ini
24	getCurrentPackage();	Dapatkan nama paket Android saat ini
25	installApp();	Instal aplikasi yang diberikan ke perangkat
26	isAppInstalled();	Periksa apakah aplikasi yang ditentukan diinstal pada perangkat
27	launchApp();	Luncurkan aplikasi yang sedang diuji di perangkat
28	runAppInBackground();	Kirim aplikasi yang sedang berjalan untuk sesi ini ke latar belakang
29	driver.closeApp();	Tutup aplikasi di perangkat
30	driver.resetApp();	Setel ulang aplikasi yang sedang berjalan untuk sesi ini
31	removeApp();	Hapus aplikasi dari perangkat
32	terminateApp();	Hentikan aplikasi yang diberikan pada perangkat
33	queryAppState();	Dapatkan status aplikasi yang diberikan di perangkat
34	endTestCoverage();	Dapatkan data cakupan pengujian
35	getClipboard();	Dapatkan konten clipboard sistem berupa plain text

36	getClipboardText();	Dapatkan konten clipboard sistem
37	setClipboard();	Atur konten clipboard sistem
38	setClipboardText();	Atur konten clipboard sistem
39	setPowerAC();	Meniru perubahan status daya pada emulator yang terhubung.
40	setPowerCapacity();	Meniru perubahan kapasitas daya pada emulator yang terhubung.
41	pushFile();	Mengupload sebuah file pada storage device
42	pullFile();	Mendapat sebuah file dari storage device
43	findElementByAccessibilityId();	Mendapat sebuah elemen dari aplikasi dari berdasarkan ID
44	findElementByClassName();	Mendpata sebuah elemen dari aplikasi berdasarkan class nya
45	findElementsByClassName();	Mendapat beberapa elemen dari aplikasi berdasarkan classname
46	findElementsByAccessibilityId();	Mendapat beberapa elemen dari aplikasi berdasarkan Accessibility ID nya
47	element.click();	Mensimulasikan sentuhan pada element mobile
48	element.sendKeys();	mengirim value string pada element mobile
49	element.clear();	Menghapus value pada element mobile
50	lockDevice();	Mengunci Device

Lampiran 2

Method-method Android Driver

No	Method	Deskripsi
1	currentActivity()	Mendapat Aktivitas dari aplikasi yang dijalankan
2	endTestCoverage(String intent, String path)	Mendapat data cakupan tes.
3	findElement(org.openqa.selenium.By by)	Mencari WebElement Pertama menggunaakan method yang digunakan
4	getConnection()	Mendapat setelan network dari perangkat yang digunakan.
5	ignoreUnimportantViews(Boolean compress)	Menyetel `ignoreUnimportantViews`.
6	isLocked()	Mengecek apakah perangkat yang digunakan terkunci.
7	lockDevice()	Method untuk mengunci device.
8	longPressKeyCode(int key)	mengirim long key event pada perangkat
9	longPressKeyCode(int key, Integer metastate)	Mengiring long key event dengan sebuah Android metastate pada perangkat.
10	openNotifications()	Membuka panel notifikasi pada perangkat android.
11	pressKeyCode(int key)	Mengirim key event pada perangkat android.
12	pushFile(String remotePath, byte[] base64Data)	Menyimpan base64 encoded data sebagai file pada perangkat.
13	pushFile(String remotePath, File file)	Menyimpan file pada perangkat.

14	setConnection(Connection connection)	Menyetel koneksi internet pada perangkat
15	startActivity(String appPackage, String appActivity)	Method yang digunakan untuk menjalankan Activity dari suatu aplikasi.
16	swipe(int startx, int starty, int endx, int endy, int duration)	Mensimulasikan gerak menggeser pada layar
17	toggleLocationServices()	Mengalihkan layanan lokasi
18	unlockDevice()	Method untuk membuka kunci perangkat

Lampiran 3

IOS Driver

No	Method	Deskripsi
1	execute(String driverCommand, Map<String,?> parameters)	Menjalankan perintah aplikasi IOS
2	findElement(org.openqa.selenium.By by)	Mencari Element menggunakan class By
3	getMouse()	mendapat mouse (deprecated)
4	hideKeyboard(String keyName)	mendapat mouse (deprecated)
5	hideKeyboard(String strategy, String keyName)	mendapat mouse (deprecated)
6	lockDevice(int seconds)	mensimulasikan mengunci device
7	shake()	mensimulasikan menggoyang device
8	swipe(int startx, int starty, int endx, int endy, int duration)	mensimulasikan gerakan swipe pada device