Pembangkit Kasus Uji Berbasis Model Pada Antarmuka (GUI) Aplikasi Android

Ni Putu Surya Febyanti Kusumadewi[1], Sri Widowati [2], Jati Hiliamsyah Husen [3]

[1,2,3] Fakultas Informatika, Universitas Telkom, Bandung
[1]febyskusumadewi@students.telkomuniversity.ac.id, [2]sriwidowati@telkomuniversity.ac.id,
[3]jatihusen@telkomuniversity.ac.id,

Abstrak

Antarmuka pengguna grafis merupakan salah satu bagian penting dari perangkat lunak karena pengguna berinteraksi dengan perangkat lunak melalui widget seperti button, text field, dan image button yang terdapat pada antarmuka pengguna grafis. Membangun antarmuka pengguna grafis memerlukan source code yang cukup banyak sesuai dengan kebutuhan antarmuka yang dibangun, sehingga menyebabkan rentan terjadi kesalahan pada antarmuka pengguna grafis. Salah satu kesalahan pada antarmuka pengguna grafis adalah incorrect state of widgets. Kesalahan incorrect state of widgets menyebabkan keadaan dan respon dari widget berbeda dari hasilnya yang diharapkan. Kesalahan ini juga akan memberikan dampak yang negatif bagi pengguna aplikasi. Oleh karena itu, pengembang perlu melakukan salah satu tahapan penting dalam siklus hidup perangkat lunak yaitu pengujian. Untuk melakukan pengujian, diperlukan adanya test case. Penelitian ini bertujuan untuk membangun tool yang dapat membangkitkan test case secara otomatis untuk mengatasi kesalahan incorrect state of widgets pada antarmuka pengguna grafis. Adapun metode yang diusulkan adalah model-based testing. Hasil test case yang diperoleh melalui tool akan dievaluasi dengan menggunakan metode manual exploratory testing. Hasil dari evaluasi yang dilakukan menyatakan bahwa hasil test case dari tool lebih baik untuk digunakan dalam melakukan pengujian antarmuka pengguna grafis karena test case yang diperoleh hampir mencakup seluruh bagian widget dan condition pada antarmuka aplikasi.

Kata kunci : Antarmuka pengguna grafis, android, pembangkit test case, manual exploratory testing, incorrect state of widgets

Abstract

The graphical user interface is one of the important parts of software, because users interact with software through widgets such as buttons, text fields, and image buttons which was found on graphical user interfaces. Building a graphical user interface required a lot of source code in accordance with the needs of the interface that was built, thus causing errors in the graphical user interface. One of the errors in the graphical user interface was incorrect state of widgets. Incorrect state of widgets errors caused the state and response of widgets to differ from the expected results. This error would also have a negative impact on application users. Therefore, the developer needed to do one of the important stages in the software life cycle, namely testing. To do the test, a test case was needed. This study aimed to build a tool that could generate test cases automatically to overcome incorrect state of widgets in the graphical user interface. The proposed method was model-based testing. The test case results which was obtained through the tool would be evaluated by using the exploratory testing manual method. The results of the evaluation carried out stated that the results of the tool test case were better to be used in testing the graphical user interface, because the obtained test cases almost covered all parts of the widget and condition in the application interface.

Keywords : Graphical User Interface (GUI), android, test case generator, manual exploratory testing, incorrect state of widgets

1. Pendahuluan

Latar Belakang

Sistem operasi android menjadi sangat populer dan berkembang dengan pesat beberapa tahun terakhir. Pada Juli 2015, survei menyatakan bahwa Android memiliki jumlah pengguna aktif terbanyak dengan total angka mencapai satu miliar mengalahkan iOS [13] [15]. Dengan populernya sistem operasi android, aplikasi berbasis android juga mengalami perkembangan yang signifikan. Bahkan di era saat ini, aplikasi seluler menjadi bagian yang cukup sering digunakan dalam kehidupan sehari - hari karena menunjang keperluan pribadi maupun keperluan

bisnis [13]. Perkembangan tersebut menyebabkan pengembang aplikasi seluler memiliki banyak pesaing, sehingga pengembang aplikasi seluler berlomba - lomba untuk membangun dan mengembangkan aplikasi seluler dalam interval waktu yang singkat [13]. Salah satu bagian yang dikembangkan dalam aplikasi seluler adalah antarmuka pengguna grafis [2].

Antarmuka pengguna grafis menjadi salah satu bagian penting karena antarmuka pengguna grafis menjadi media bagi pengguna untuk berinteraksi dengan aplikasi. Pengguna berinteraksi dengan perangkat lunak melalui widget yang terdapat pada antarmuka pengguna grafis[12] [4]. Widget pada antarmuka pengguna grafis dibangun melalui source code yang biasanya terdapat pada file XML. Source code antarmuka pengguna grafis terbilang cukup banyak [2], hingga menyebabkan rentan terjadi kesalahan ataupun kekurangan yang berdampak pada antarmuka pengguna grafis. Salah satu kategori kesalahan yang sering terjadi dalam antarmuka pengguna grafis yaitu incorrect state of widgets [12]. Incorrect state of widgets terjadi jika keadaan widget berbeda dengan hasil yang diharapkan, misalnya button tidak dapat di-klik [12].

Kesalahan incorrect state of widgets menyebabkan keadaan dan respon dari widget berbeda dari hasil yang diharapkan [12]. Terjadinya kesalahan dalam antarmuka pengguna grafis seperti incorrect state of widgets memberikan dampak negatif terhadap pengguna aplikasi. Apabila kesalahan pada antarmuka pengguna grafis tidak terdeteksi dan tidak dilakukan perbaikan saat fase pengembangan, maka akan menimbulkan masalah yang lebih besar seperti ketidaknyamanan pengguna dalam berinteraksi dengan antarmuka pengguna grafis. Oleh karena itu, pengembang aplikasi seluler perlu melakukan salah satu tahapan penting dalam siklus hidup perangkat lunak yang disebut dengan pengujian.

Pengujian adalah aktivitas yang dilakukan untuk mengevaluasi kualitas produk yang digunakan untuk memperbaikinya dengan mengidentifikasi masalah yang ada perangkat lunak [1]. Salah satu bagian perangkat lunak yang perlu dievaluasi adalah antarmuka pengguna grafis. Untuk mengevaluasi kualitas antarmuka pengguna grafis, perlu dilakukan pengujian pengguna grafis sehingga kesalahan incorect state of widgets dapat diketahui. Pengujian antarmuka pengguna grafis tergolong ke dalam jenis black box testing karena ketika melakukan pengujian antarmuka pengguna grafis, tester tidak memiliki akses ke dalam source code aplikasi seluler [11]. Untuk melakukan pengujian antarmuka pengguna grafis, tester memerlukan dan harus memiliki test case. Test case adalah serangkaian kondisi dimana tester akan memeriksa apakah suatu perangkat lunak yang diuji memenuhi persyaratan atau dapat dikatakan berfungsi dengan benar. Test case juga memberikan langkah - langkah yang harus diperiksa pada suatu perangkat lunak.

Metode yang digunakan dalam penelitian ini adalah model-based testing. Model-based testing dipilih karena penelitian ini bertujuan untuk membangun tool pembangkitan test case berdasarkan model pada antarmuka pengguna grafis aplikasi android. Selain itu, dari beberapa penelitian yang telah dilakukan dapat dilihat bahwa model-based testing menjadi metode yang cukup populer untuk membangun model dalam melakukan pembangkitan test case [13] [7]. Fokus pembangkitan test case untuk kesalahan incorrect state of widgets seperti kesalahan pada button, imagebutton, imageview, textview, dan edittext. Kesalahan incorrect state of widgets diangkat karena setiap aplikasi seluler memiliki widget banyak dan pengguna berinteraksi dengan aplikasi android melalui widget, sehingga perlu dipastikan bahwa widget pada aplikasi seluler berfungsi dengan benar. Hasil test case ini akan digunakan sebagai bahan pengujian antarmuka pengguna grafis pada aplikasi android.

2. Studi Terkait

2.1 Antarmuka Pengguna Grafis

Antarmuka pengguna grafis adalah front-end grafis pada sistem perangkat lunak yang menerima input dari pengguna dan menghasilkan output grafis [2]. Pengguna berinteraksi dengan perangkat lunak melalui widget seperti radio button, text field, menu drop-down, checkboxes yang terdapat pada antarmuka pengguna grafis [12] [4]. Peristiwa input yang terjadi pada antarmuka pengguna grafis seperti klik mouse, memilih widget, mengetik pada text field yang kemudian akan mengubah keadaan pada widget antarmuka pengguna grafis [2] [12]. Contohnya seperti menekan button hapus, maka akan menghasilkan tindakan yang akan menghapus sesuatu dari perangkat lunak [12].

Antarmuka pengguna grafis terdapat di semua jenis gadget, salah satunya terdapat pada gadget dalam aplikasi android. Antarmuka pengguna grafis pada aplikasi android memiliki bagian penting yang disebut layout. Layout mendefinisikan struktur visual untuk antarmuka pengguna seperti widget [3], dan setiap layout memiliki beberapa widget didalamnya. Widget adalah objek untuk melakukan kontrol pada aplikasi android. Jenis - jenis widget dalam antarmuka android dijabarkan pada Tabel 1 sebagai berikut :

Tabel 1. Widget pada antarmuka android [14]

No	Widget	Deskripsi Widget
1	ImageView	Kontrol antarmuka untuk menampilkan file gambar.
2	TextView	Kontrol antarmuka untuk mengatur dan menampilkan teks.
3	EditText	Kontrol antarmuka untuk memasukkan atau memodifikasi teks.
4	Button	Kontrol antarmuka untuk melakukan tindakan setiap kali diklik atau diketuk.
5	ImageButton	Kontrol antarmuka yang menampilkan tombol dengan gambar dan dapat melakukan tindakan setiap kali diklik atau diketuk.
6	CheckBoxes	Kontrol antarmuka untuk memilih opsi satu atau lebih dari satu opsi.
7	RadioButton	Kontrol antarmuka untuk mengubah status kontrol.
8	ProgressBar	Kontrol antarmuka untuk menunjukkan kemajuan operasi. Contoh : mengunduh dan mengunggah file.

Setiap widget pada antarmuka android memiliki condition. Condition yang dimiliki oleh widget dapat berbeda - beda. Jenis - jenis condition dalam antarmuka android dijabarkan pada Tabel 2 sebagai berikut :

Tabel 2. Condition pada antarmuka android [3]

No	Condition	Deskripsi Condition
1	Checkable	Mengubah status tampilan dari kondisi saat ini refresh
2	Checked	Memastikan tampilan dapat diperiksa.
3	Clickable	Widget dapat diklik.
4	Enabled	Widget diaktifkan.
5	Focusable	Widget yang diklik terlihat lebih mencolok.
6	Focused	Widget memiliki fokus.
7	Scrollable	Layout dapat di-scroll.
8	Long-clickable	Widget dapat diklik dalam waktu yang cukup panjang atau lama.
9	Password	Widget dapat dimasukkan kata sandi.
10	Selected	Widget atau layout yang saat ini dipilih.

Antarmuka pengguna grafis memiliki beberapa kesalahan seperti yang dijabarkan pada penelitian [12]. Penelitian ini mengklasifikasi kategori kesalahan pada antarmuka pengguna grafis. Salah satu kategori kesalahan pada antarmuka pengguna grafis yaitu GUI Structure and Aesthetics. Kesalahan GUI Structure and Aesthetics dijabarkan pada Tabel 3 berikut ini :

Tabel 3. Kesalahan pada GUI Structure and Aesthetics [12]

Kategori Kesalahan	Letak Kesalahan	Kesalahan yang mungkin terjadi
Incorrect layout of widgets	Sejajar, kedalaman, dimensi, orientasi	Posisi dari dua widget terbalik. Teks tidak terlihat karena ukuran text field yang terlalu kecil.
Incorrect state of widgets	Dapat dilihat, dapat diaktifkan, dapat dipilih, memiliki fokus, dapat diedit, dapat diperbesar	Tidak dapat mengklik button karena button tidak diaktifkan. Sebuah window tidak terlihat, sehingga widget tidak dapat digunakan. Tidak dapat menulis di text field karena text field tidak diaktifkan.
Incorrect appearance of widgets	Font, warna, ikon, label	Ikon pada button tidak terlihat. Dalam antarmuka pembangkit listrik, warna yang mencerminkan status kritis pompa adalah warna hijau, seharusnya warna yang mencerminkan status kritis pompa adalah warna merah.

2.2 Pengujian Antarmuka Pengguna Grafis

Pengujian adalah aktivitas yang dilakukan untuk mengevaluasi kualitas produk yang digunakan untuk memperbaikinya dengan mengidentifikasi defect dan masalah [1]. Pengujian perangkat lunak adalah aktivitas yang dilakukan untuk mengevaluasi kualitas produk yang digunakan untuk memperbaikinya dengan mengidentifikasi

defect dan masalah [1]. Aktivitas pengujian bertujuan untuk mengungkap kesalahan pada perangkat lunak, agar kesalahan tersebut dapat diperbaiki [2]. Kesalahan pada perangkat lunak biasanya dilakukan oleh seorang pengembang perangkat lunak atau programmer ditahap implementasi perangkat lunak [2]. Salah satu kesalahan yang ada di perangkat lunak terdapat pada antarmuka pengguna grafis, sehingga diperlukan adanya aktivitas pengujian antarmuka pengguna grafis.

Pengujian antarmuka pengguna grafis adalah pengujian pada sistem perangkat lunak yang memiliki bagian front-end atau antarmuka pengguna grafis [6]. Pengujian antarmuka pengguna grafis berfokus pada urutan peristiwa atau interaksi pada widget dengan memeriksa kebenaran dari widget antarmuka pengguna grafis seperti radio button, text field, menu drop-down, checkboxes, dll [2]. Pengujian antarmuka pengguna grafis dapat dilakukan dengan menggunakan teknik pengujian black box.

2.2.1 Black Box Testing

Pengujian black box adalah pengujian yang dilakukan tanpa mengetahui cara dalam kerja aplikasi dan hanya memeriksa aspek - aspek mendasar dari sistem [11]. Saat melakukan pengujian black box, seorang tester harus mengetahui arsitektur sistem karena tester tidak memiliki akses ke source code aplikasi [11]. Keuntungan dari black box testing yaitu dapat membangun test case secara lebih cepat dan efisien untuk melakukan pengujian dalam lingkup source code yang besar [11]. Untuk melakukan pengujian perangkat lunak dengan teknik apapun, sangat diperlukan untuk membangun sebuah test case.

2.2.2 Model-Based Testing

Model-Based Testing (MBT) adalah proses dimana test case dapat dihasilkan secara otomatis dengan menggunakan sebuah model [13]. Model dapat diperoleh dengan cara manual atau otomatis [13]. Setelah menghasilkan sebuah model, model dapat digunakan untuk menghasilkan test case. MBT dapat digunakan untuk aplikasi web, desktop, dan aplikasi pada telepon seluler [13]. Teknik MBT berfokus pada kebenaran fungsional yang akan meningkatkan efektivitas dari pengujian antarmuka pengguna grafis [5].

2.2.3 Exploratory Testing

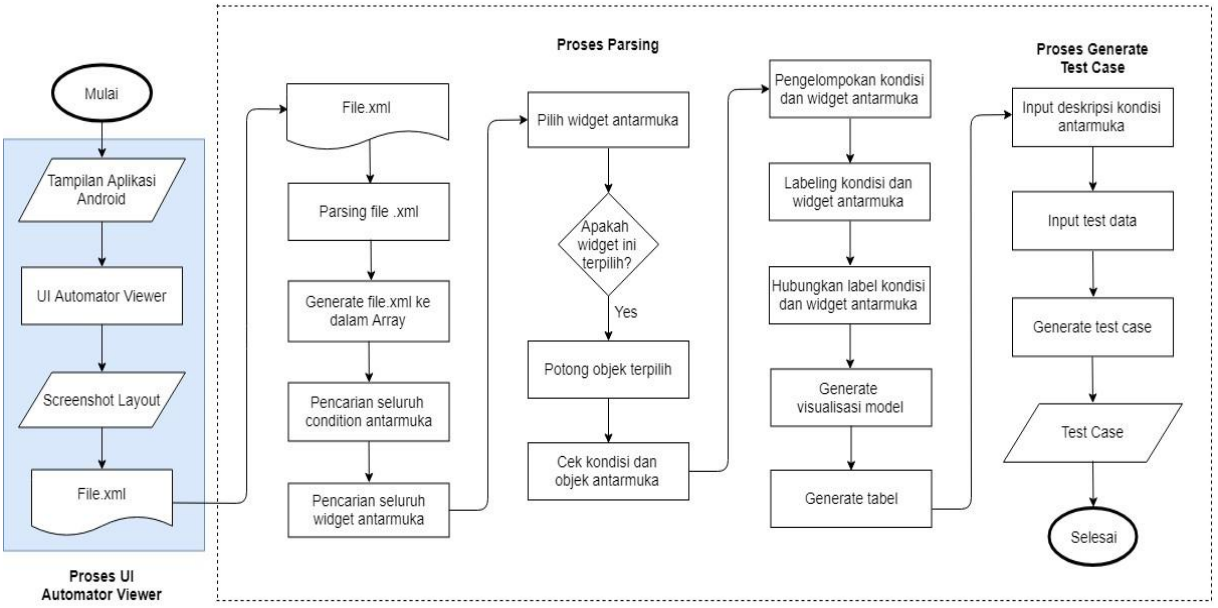
Exploratory testing adalah bentuk pengujian dimana tester berinteraksi dengan sistem perangkat lunak berdasarkan pengetahuan, pengalaman dan intuisi untuk menemukan bug [8] [10]. Pendekatan exploratory testing tidak bergantung pada dokumentasi test case dalam melakukan pengujian [10]. Exploratory testing telah diterima di industri dan dianggap sebagai cara efektif dalam menemukan bug. Literatur praktisi berpendapat bahwa exploratory testing memiliki kelebihan dalam mengurangi overhead, membantu tim untuk memahami fitur dan perilaku perangkat lunak serta memungkinkan tester untuk fokus pada area tertentu selama pengujian [8]. Namun, exploratory testing juga memiliki kekurangan dimana tester hanya mampu menguji sebagaian fungsi perangkat lunak dalam waktu tertentu dan sulit bagi tester untuk menentukan bagian perangkat lunak yang telah diuji selama sesi pengujian sehingga meningkatkan risiko untuk melewatkan beberapa bagian perangkat lunak yang belum diuji [8].

2.2.4 Test Case

Test case adalah serangkaian kondisi dimana tester akan memeriksa apakah suatu perangkat lunak yang diuji memenuhi persyaratan atau dapat dikatakan berfungsi dengan benar. Dua pendekatan yang digunakan dalam membangun test case, yaitu test case melalui requirement dan desain, dan menggunakan source-code [9]. Kedua pendekatan test case tersebut membantu untuk menemukan masalah dalam perangkat lunak. Pembuatan test case dapat dilakukan melalui dua cara yakni secara otomatis atau secara manual [9] [13]. Test case harus memiliki expected result, dan actual result. Untuk menentukan expected result pada test case diperlukan adanya test data. Test data adalah data yang diperlukan untuk melakukan pengujian perangkat lunak. Test data juga digunakan untuk memverifikasi perilaku perangkat lunak terhadap input-an yang valid dan tidak valid.

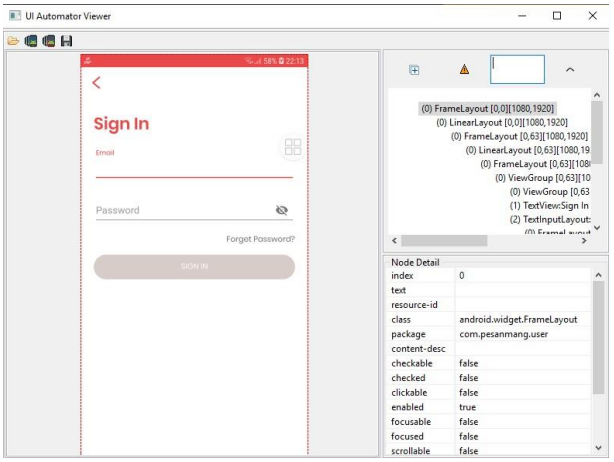
3. Sistem yang Dibangun

Pembangkit test case dibangun menggunakan bahasa pemrograman Python 3, pada kernel Jupyter Notebook. Adapun library yang digunakan, yaitu : import xml.etree.ElementTree as et, import pandas as pd, import networkx as nx, dan import matplotlib.pyplot as plt. Pembangkit test case memiliki tiga proses utama, yaitu proses UI Automator Viewer, proses parsing, dan proses generate test case. Gambaran umum alur pembangunan sistem dapat dilihat pada gambar di bawah ini :



Gambar 1. Diagram Alur Pembangunan Tool

3.1 UI Automator Viewer



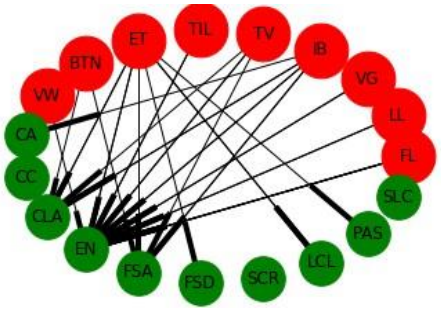
Gambar 2. Diagram Alur Pembangunan Sistem

UI automator viewer adalah sebuah tool dari Android SDK. UI automator viewer memberikan akses untuk memindai dan menganalisis komponen antarmuka yang sedang ditampilkan pada perangkat android [3]. UI automator viewer digunakan untuk mengekstrak setiap kontrol, memeriksa hierarki tata letak, dan melihat komponen dari layout aplikasi android. Layout aplikasi android akan di screenshot oleh tool secara otomatis dan akan menghasilkan gambar hasil screenshot aplikasi dan format file XML. Penelitian [7] juga menggunakan Tool UI automator viewer untuk menghasilkan file teks XML. File XML berisi struktur code dari layout antarmuka pengguna grafis aplikasi android. File XML terdiri dari index, text, resource-id, class, package, condition, dan, bounds. File XML yang diperoleh akan menjadi data dan digunakan untuk melakukan proses selanjutnya yaitu parsing.

3.2 Parsing

Proses parsing merupakan rangkaian kedua dari penelitian ini. Tahap pertama tool akan meminta masukkan nama file xml yang akan diproses. Tahap kedua akan dilanjutkan dengan melakukan proses parsing file xml. Untuk melakukan proses parsing, tool menggunakan library import xml.etree.ElementTree as et. Hasil parsing file xml dimasukkan ke dalam array dan digunakan untuk proses selanjutnya. Pada tahap ketiga, dicari seluruh condition yang terdapat didalam array dan menghasilkan seluruh condition yang terdapat didalam array. Pada tahap keempat,

dicari seluruh widget yang terdapat didalam array dan menghasilkan seluruh widget yang terdapat didalam array. Tahap kelima akan dilakukan pemilihan widget. Pemilihan widget dilakukan dengan menggunakan aturan, dimana hanya widget yang ingin untuk dijadikan bahan pengujian yang akan dipilih. Setiap Widget yang terpilih akan memiliki keterangan unik yang berasal dari ID, content description, text, dan bounds atau letak dari widget tersebut. Setelah memperoleh widget yang akan dijadikan sebagai bahan pengujian, maka akan dilakukan tahapan keenam yakni cut xml. Cut xml merupakan proses dimana akan dihilangkan widget tidak terpilih dan hanya memilih widget yang akan dijadikan sebagai bahan pengujian. Hasil dari widget berserta keterangan unik yang terpilih akan dipasangkan dengan condition yang bernilai True dari widget tersebut. Keluaran dari tahap ini adalah jenis widget, condition, dan keterangan unik pada widget. Hasil dari tahap ketujuh akan dikelompokkan berdasarkan condition dari widget. Pada tahap kesembilan, dilakukan pemberian label berupa kode terhadap seluruh condition dan widget. Pemberian label digunakan untuk membangun visualisasi hubungan antara condition dengan widget ke dalam bentuk graph. Untuk membangun visualisasi condition dengan widget digunakan library import networkx as nx dan untuk menghasilkan gambar visualisasi digunakan library import matplotlib.pyplot as plt. Hasil dari tahap kesembilan sebagai berikut :



Gambar 3. Visualisasi Hubungan Condition dan Widget

Gambar diatas memvisualisasikan hubungan antara setiap widget dengan condition nya. Bagian yang berwarna merah merupakan berbagai macam widget yang terdapat pada hasil parsing file xml, dan bagian yang berwarna hijau merupakan berbagai macam condition yang terdapat pada hasil parsing file xml.

Pada tahapan terakhir, dilakukan generate tabel untuk menjabarkan widget berserta condition-nya. Untuk tahap ini digunakan library import pandas as pd. Hasil dari tahap ini sebagai berikut :

	Widget	checkable	checked	clickable	enabled	focusable	focused	scrollable	long-clickable	password	selected
0	android.widget.ImageView - id com.gojek.app:id...	0	0	0	1	0	0	0	0	0	0
1	android.widget.ImageView - id com.gojek.app:id...	0	0	0	1	0	0	0	0	0	0
2	android.widget.ImageView - id com.gojek.app:id...	0	0	0	1	0	0	0	0	0	0
3	android.widget.TextView - id com.gojek.app:id/...	0	0	0	1	0	0	0	0	0	0
4	android.widget.TextView - id com.gojek.app:id/...	0	0	0	1	0	0	0	0	0	0
5	android.widget.Button - id com.gojek.app:id/bt...	0	0	1	1	1	0	0	0	0	0
6	android.widget.Button - id com.gojek.app:id/bt...	0	0	1	1	1	0	0	0	0	0
7	android.widget.ImageView - id com.gojek.app:id...	0	0	0	1	0	0	0	0	0	0
8	android.widget.TextView - id com.gojek.app:id/...	0	0	0	1	0	0	0	0	0	0
9	android.widget.TextView - id com.gojek.app:id/...	0	0	1	1	1	0	0	1	0	0

Gambar 4. Hasil Tabel Widget dan Condition

Pada gambar diatas, terdapat berbagai macam widget dengan condition-nya. Setiap widget memiliki dua nilai condition yang berbeda, yaitu condition bernilai True = 1 dan condition bernilai False = 0. Seperti contoh gambar diatas, widget button memiliki tiga condition true yakni enable, clickable, dan focusable. Oleh karena itu, bagian condition untuk widget button pada gambar bernilai True = 1 dibagian enable, clickable, dan focusable.

3.3 Generate Test Case

Proses generate test case merupakan rangkaian terakhir dari penelitian ini. Pertama - tama dilakukan input deskripsi dari seluruh condition yang diperoleh pada file xml, lalu dilanjutkan ke tahapan kedua yakni input test data. Ditahap ini akan di-inputkan deskripsi test data dari setiap condition yang mengacu pada nilai true dan false dari suatu widget. Test data yang digunakan pada penelitian tugas akhir, dijabarkan pada Tabel 4 dibawah ini :

Tabel 4. Test Data

No	Condition	True	False
1	Checkable	1. Widget dapat ditekan. 2. Setelah widget ditekan, layout akan mengalami perubahan dan menampilkan layout baru	1. Widget tidak dapat ditekan. 2. Setelah widget ditekan, tidak terjadi perubahan apapun pada layout.
2	Checked	1. Widget dapat ditekan. 2. Widget dapat ditandai.	1. Widget tidak dapat ditekan. 2. Widget tidak dapat ditandai.
3	Clickable	1. Widget dapat ditekan	1. Widget tidak dapat ditekan.
4	Enabled	1. Widget dapat ditemukan. 2. Widget dapat dilihat	1. Widget tidak dapat ditemukan 2. Widget tidak dapat dilihat
5	Focusable	1. Widget dapat ditekan 2. Tampilan widget menjadi berbeda dari widget yang lain	1. Widget tidak dapat ditekan 2. Tampilan tidak berubah
6	Focused	1. Tampilan widget terlihat jelas	1. Tampilan widget buram atau tidak jelas
7	Scrollable	1. Scroll ke atas 2. Scroll ke bawah	1. Tidak dapat scroll ke atas 2. Tidak dapat scroll ke bawah
8	Long-clickable	1. Widget dapat ditekan 2. Widget dapat di-hold atau ditekan dalam waktu cukup lama 3. Layout tidak berganti / refresh	1. Widget tidak dapat ditekan 2. Widget tidak dapat di-hold atau ditekan dalam waktu cukup lama 3. Layout berganti / refresh
9	Password	1. Widget dapat ditekan 2. Widget dapat dimasukkan teks 3. Hasil masukkan teks disamarkan	1. Widget tidak dapat ditekan 2. Widget tidak dapat dimasukkan teks 3. Hasil masukkan teks tidak disamarkan
10	Selected	1. Widget dapat ditekan 2. Widget dapat dipilih	1. Widget tidak dapat ditekan 2. Widget tidak dapat dipilih

Generate test data dilakukan untuk menentukan expected result pada test case. Setelah proses input test data selesai, akan dilakukan tahapan terakhir yakni generate test case. Test case di-generate ke dalam bentuk tabel dengan bagian - bagian seperti Test Case ID, Test Case Description, Object UI, Condition, Information, Test Data, Expected Result, Actual Result, dan Komentar. Hasil keluaran dari tool yang dibangun adalah sebuah test case dari layout aplikasi android. Hasil test case pada tool akan ditranformasi ke dalam file .csv untuk mempermudah membaca dan pengisian test case. Test case ini akan digunakan sebagai bahan pengujian antarmuka pengguna grafis pada aplikasi android.

Test Case ID	Test Case Description	Object UI	Condition	Information	Test Data	Expected Result	Actual Result	Komentar
SI-0	Memastikan bahwa android.widget.FrameLayout de...	android.widget.FrameLayout	enabled	bounds [0,0][1080,1920]	1. dapat menemukan android.widget.FrameLayout	sukses	NaN	NaN
SI-1	Memastikan bahwa android.widget.FrameLayout de...	android.widget.FrameLayout	enabled	bounds [0,0][1080,1920]	1. tidak dapat menemukan android.widget.FrameL...	gagal	NaN	NaN
SI-2	Memastikan bahwa android.widget.LinearLayout d...	android.widget.LinearLayout	enabled	bounds [0,0][1080,1920]	1. dapat menemukan android.widget.LinearLayout	sukses	NaN	NaN
SI-3	Memastikan bahwa android.widget.LinearLayout d...	android.widget.LinearLayout	enabled	bounds [0,0][1080,1920]	1. tidak dapat menemukan android.widget.Linear...	gagal	NaN	NaN

Gambar 5. Hasil Test Case

4. Evaluasi

4.1 Hasil Tool Pembangkit Test Case

Pembangkitan test case dilakukan dengan menggunakan tool yang sudah dibangun. Layout dari aplikasi Gojek yang dipilih cukup beragam karena terdapat widget yang berbeda - beda disetiap layout aplikasi Gojek. Setiap layout akan direkam dan menghasilkan file xml. File xml layout aplikasi Gojek mengalami proses parsing hingga menghasilkan widget dan condition. Dari setiap layout aplikasi Gojek, diperoleh hasil widget dan condition yang

berbeda - beda. Tabel 5 merupakan hasil widget dan condition yang diperoleh dari masing - masing layout aplikasi Gojek.

Tabel 5. Hasil widget dan condition pada layout aplikasi Gojek

No	Tampilan Antarmuka	Jumlah Widget dan Condition
1	Welcome layout	17
2	Register layout	27
3	Home layout	46
4	Go ride layout	18

Hasil condition dari proses parsing diberikan deskripsi kondisi dan diberikan test data untuk menentukan expected result pada test case. Hasil widget dan condition yang berbeda - beda menyebabkan hasil test case dari setiap layout aplikasi gojek memiliki jumlah yang berbeda - beda. Tabel 6 merupakan total dari test case yang diperoleh dari masing - masing layout aplikasi gojek.

Tabel 6. Hasil test case pada layout aplikasi Gojek

No	Tampilan Antarmuka	Jumlah Test Case
1	Welcome layout	35
2	Register layout	57
3	Home layout	92
4	Go ride layout	36

4.2 Evaluasi Hasil Tool Pembangkit Test Case

Objektivitas yang dievaluasi adalah jumlah widget dan condition yang diperoleh dari masing - masing layout aplikasi gojek dengan menggunakan tool pembangkit test case dan metode manual exploratory testing. Manual exploratory testing dilakukan oleh sepuluh responden. Empat responden adalah orang sudah pernah melakukan pengujian pada perangkat lunak dan enam lainnya adalah orang yang belum pernah melakukan pengujian pada perangkat lunak. Skenario yang dilakukan untuk evaluasi yaitu setiap responden diberikan lima gambar layout aplikasi gojek, lalu responden akan menganalisis lima gambar tersebut hingga menghasilkan widget dan condition. Hasil evaluasi dari responden, dianalisis dan dihitung jumlah widget dan condition yang sesuai dan tidak sesuai antara hasil tool pembangkit test case dengan hasil metode manual exploratory testing.

4.2.1 Welcome Layout

Dari tampilan antarmuka welcome layout aplikasi gojek, tool pembangkit test case dan metode manual exploratory testing memperoleh tiga jenis widget yang sama. Rincian widget dari tampilan antarmuka welcome layout aplikasi gojek dapat dilihat pada Tabel 7, dan untuk tampilan antarmuka welcome layout dapat dilihat pada lembar lampiran Gambar 8.

Tabel 7. Hasil Jenis Widget pada Welcome Layout

No	Widget Tool	Widget Manual Exploratory Testing
1	ImageView	ImageView
2	Button	Button
3	TextView	TextView

Dari tampilan antarmuka welcome layout aplikasi gojek, tool pembangkit test case memperoleh empat jenis condition. Sedangkan metode manual exploratory testing hanya memperoleh dua jenis condition. Rincian jenis condition dari tampilan antarmuka welcome layout dapat dilihat pada Tabel 8.

Tabel 8. Hasil Jenis Condition pada Welcome Layout

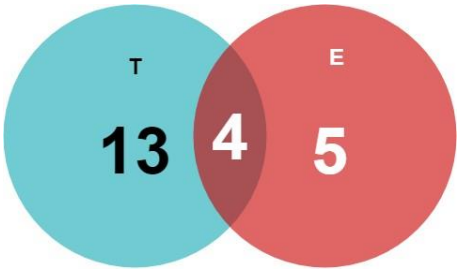
No	Condition Tool	Condition Manual Exploratory Testing
1	Enabled	Enabled
2	Clickable	Clickable
3	Focusable	-
4	Long-clickable	-

Dari tampilan antarmuka welcome layout aplikasi gojek, tool pembangkit test case memperoleh tujuh belas widget dan condition. Sedangkan metode manual exploratory testing hanya memperoleh sembilan widget dan condition. Rincian jenis widget dan condition dari tampilan antarmuka welcome layout dapat dilihat pada Tabel 8.

Tabel 9. Hasil Widget dan Condition pada Welcome Layout

No	Widget	Condition	Tool	Manual Exploratory Testing	Keterangan
1	ImageView	enabled	Ada	Tidak ada	id/img logo
2	ImageView	clickable	Tidak ada	Ada	id/img logo
3	ImageView	enabled	Ada	Tidak ada	id/img country picker
4	ImageView	enabled	Ada	Tidak ada	id/img auth on boarding illustration
5	ImageView	enabled	Ada	Tidak ada	id/img auth facebook logo
6	Button	clickable	Tidak ada	Ada	id/img country picker
7	Button	clickable	Ada	Ada	id/btn login
8	Button	enabled	Ada	Tidak ada	id/btn login
9	Button	focusable	Ada	Tidak ada	id/btn login
10	Button	clickable	Ada	Ada	id/btn signup
11	Button	enabled	Ada	Tidak ada	id/btn signup
12	Button	focusable	Ada	Tidak ada	id/btn signup
13	Button	clickable	Tidak ada	Ada	continue facebook
14	TextView	enabled	Ada	Tidak ada	id/text welcome
15	TextView	enabled	Ada	Tidak ada	id/text welcome body
16	TextView	enabled	Ada	Tidak ada	id/text fb cta
17	TextView	clickable	Ada	Ada	id/text terms
18	TextView	enabled	Ada	Ada	id/text terms
19	TextView	focusable	Ada	Tidak ada	id/text terms
20	TextView	clickable	Tidak ada	Ada	privacy policy
21	TextView	enable	Tidak ada	Ada	privacy policy
22	TextView	long-clickable	Ada	Tidak ada	id/text terms

Berdasarkan data dari Tabel 9, diperoleh diagram venn seperti berikut ini :



Gambar 6. Hasil Diagram Venn Welcome Layout

Dari diagram venn diatas terdapat himpunan T yang berisi jumlah seluruh widget dan condition yang diperoleh melalui tool pembangkit test case. Himpunan E berisi jumlah seluruh widget dan condition yang diperoleh melalui metode manual exploratory testing. Hasil irisan dari diagram venn diatas, diperoleh empat widget dan condition yang sama, yaitu : button - clickable - id/btn_login, button - clickable - id/btn_signup, textview - clickable - id/text_terms, textview - enabled - id/text_terms.

Tool pembangkit test case menghasilkan widget dan condition yang bervariasi karena bersumber pada file xml. File tersebut berisi struktur dan informasi layout secara lengkap yang diperoleh secara langsung dari rekaman welcome layout dengan menggunakan tool UI automator viewer. Oleh karena itu, hasil widget dan condition dari tool pembangkit test case sudah mencakup seluruh keadaan pada layout. Sedangkan beberapa widget dan condition dari metode manual exploratory testing memiliki hasil yang lebih sedikit dan beririsan dengan hasil dari tool pembangkit test case. Hal ini dikarenakan mayoritas responden memilih widget yang terlihat jelas dan memilih condition sesuai dengan trigger yang diberikan pada widget tersebut.

4.2.2 Register Layout

Dari tampilan antarmuka register layout aplikasi gojek, tool pembangkit test case memperoleh empat jenis widget. Sedangkan metode manual exploratory testing hanya memperoleh tiga jenis widget. Rincian jenis widget dari tampilan antarmuka register layout aplikasi gojek dapat dilihat pada Tabel 10, dan untuk tampilan antarmuka register layout aplikasi gojek dapat dilihat pada lembar lampiran Gambar 9.

Tabel 10. Hasil Jenis Widget pada Register Layout

No	Widget Tool	Widget Manual Exploratory Testing
1	ImageView	ImageView
2	TextView	-
3	EditText	EditText
4	Button	Button

Dari tampilan antarmuka register layout aplikasi gojek, tool pembangkit test case memperoleh empat jenis condition. Sedangkan metode manual exploratory testing hanya memperoleh tiga jenis condition. Rincian jenis condition dari tampilan antarmuka register layout aplikasi gojek dapat dilihat pada Tabel 11.

Tabel 11. Hasil Jenis Condition pada Register Layout

No	Condition Tool	Condition Manual Exploratory Testing
1	Clickable	Clickable
2	Enabled	-
3	Focusable	-
4	Long-clickable	-
5	-	Input
6	-	Selected

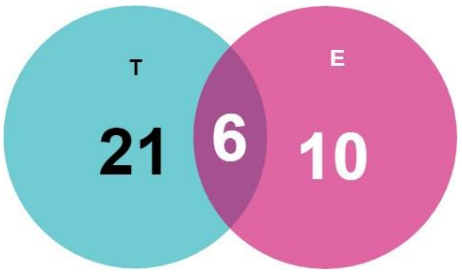
Dari tampilan antarmuka register layout aplikasi gojek, tool pembangkit test case memperoleh dua puluh tujuh widget dan condition. Sedangkan metode manual exploratory testing hanya memperoleh enam belas widget dan condition. Rincian widget dan condition dari tampilan antarmuka register layout dapat dilihat pada Tabel 12.

Tabel 12. Hasil Widget dan Condition pada Register Layout

No	Widget	Condition	Tool	Manual Exploratory Testing	Keterangan
1	ImageView	clickable	Ada	Ada	id/auth back button
2	ImageView	enabled	Ada	Tidak ada	id/auth back button
3	ImageView	focusable	Ada	Tidak ada	id/auth back button
4	ImageView	clickable	Ada	Ada	id/auth help button
5	ImageView	enabled	Ada	Tidak ada	id/auth help button
6	ImageView	focusable	Ada	Tidak ada	id/auth help button
7	ImageView	enabled	Ada	Tidak ada	id/country picker image
8	ImageView	selected	Tidak ada	Ada	flag
9	TextView	enabled	Ada	Tidak ada	id/sign in title
10	TextView	enabled	Ada	Tidak ada	text Full Name
11	TextView	enabled	Ada	Tidak ada	textView2
12	TextView	enabled	Ada	Tidak ada	text Phone Number

No	Widget	Condition	Tool	Manual Exploratory Testing	Keterangan
13	TextView	enabled	Ada	Tidak ada	id/text country
14	EditText	clickable	Ada	Ada	id/input name
15	EditText	enabled	Ada	Tidak ada	id/btn signup
16	EditText	focusable	Ada	Tidak ada	id/btn signup
17	EditText	long-clickable	Ada	Tidak ada	id/btn signup
18	EditText	clickable	Ada	Ada	id/input email
19	EditText	enabled	Ada	Tidak ada	id/input email
20	EditText	focusable	Ada	Tidak ada	id/input email
21	EditText	long-clickable	Ada	Tidak ada	id/input email
22	EditText	clickable	Ada	Ada	id/input phone
23	EditText	enabled	Ada	Tidak ada	id/input phone
24	EditText	focusable	Ada	Tidak ada	id/input phone
25	EditText	long-clickable	Ada	Tidak ada	id/input phone
26	EditText	input	Tidak ada	Ada	Full Name
27	EditText	input	Tidak ada	Ada	Email
28	EditText	input	Tidak ada	Ada	Phone Number
29	Button	clickable	Ada	Ada	id/continue button
30	Button	enabled	Ada	Tidak ada	id/continue button
31	Button	focusable	Ada	Tidak ada	id/continue button
32	Button	clickable	Tidak ada	Ada	Code Number
33	Button	clickable	Tidak ada	Ada	Question
34	Button	clickable	Tidak ada	Ada	Full Name
35	Button	clickable	Tidak ada	Ada	Email
36	Button	clickable	Tidak ada	Ada	Phone Number
37	Button	clickable	Tidak ada	Ada	Back

Berdasarkan data dari Tabel 12, diperoleh diagram venn seperti berikut ini :



Gambar 7. Hasil Diagram Venn Register Layout

Dari gambar diatas terdapat himpunan T yang berisi jumlah seluruh widget dan condition yang diperoleh melalui tool pembangkit test case. Himpunan E berisi jumlah seluruh widget dan condition yang diperoleh melalui metode manual exploratory testing. Hasil irisan dari diagram venn diatas, diperoleh enam widget dan condition yang sama, yaitu : imageview - clickable - id/auth_back_button, imageview - clickable - id/auth_help_button, edittext - clickable - id/input_name, edittext - clickable - id/input_email, edittext - clickable - id/input_phone, button - clickable - id/continue_button.

Tool pembangkit test case menghasilkan widget dan condition yang bervariasi karena bersumber pada file xml. File tersebut berisi struktur dan informasi layout secara lengkap yang diperoleh secara langsung dari rekaman register layout dengan menggunakan tool UI automator viewer. Oleh karena itu, hasil widget dan condition dari tool pembangkit test case sudah mencakup seluruh keadaan pada layout. Sedangkan beberapa widget dan condition dari metode manual exploratory testing memiliki hasil yang lebih sedikit dan beririsan dengan hasil dari tool pembangkit test case. Hal ini dikarenakan mayoritas responden memilih widget yang terlihat jelas dan memilih condition sesuai dengan trigger yang diberikan pada widget tersebut.

4.3 Home Layout

Dari tampilan antarmuka home layout aplikasi gojek, tool pembangkit test case hanya memperoleh dua jenis widget. Sedangkan metode manual exploratory testing memperoleh tiga jenis widget. Rincian jenis widget dari tampilan antarmuka home layout aplikasi gojek dapat dilihat pada Tabel 13, dan untuk tampilan antarmuka home layout dapat dilihat pada lembar lampiran Gambar 10.

Tabel 13. Hasil Jenis Widget pada Home Layout

No	Widget Tool	Widget Manual Exploratory Testing
1	TextView	TextView
2	ImageView	ImageView
3	-	Button

Dari tampilan antarmuka home layout aplikasi gojek, tool pembangkit test case hanya memperoleh dua jenis condition. Sedangkan metode manual exploratory testing memperoleh tiga jenis condition. Hasil condition yang diperoleh dari tool dan metode manual exploratory testing tidak ada yang sama. Rincian jenis condition dari tampilan antarmuka home layout dapat dilihat pada Tabel 14.

Tabel 14. Hasil Jenis Condition pada Home Layout

No	Condition Tool	Condition Manual Exploratory Testing
1	Enabled	-
2	Selected	-
3	-	Clickable
4	-	Input
5	-	Swipe

Dari tampilan antarmuka home layout aplikasi gojek, tool pembangkit test case memperoleh empat puluh enam widget dan condition. Sedangkan metode manual exploratory testing hanya memperoleh dua puluh empat widget dan condition. Rincian widget dan condition dari tampilan antarmuka home layout aplikasi gojek dapat dilihat pada Tabel 15.

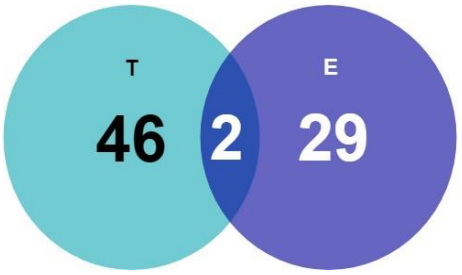
Tabel 15. Hasil Widget dan Condition pada Home Layout

No	Widget	Condition	Tool	Manual Exploratory Testing	Keterangan
1	TextView	enabled	Ada	Tidak ada	id/placeholder search
2	ImageView	enabled	Ada	Tidak ada	id/image view my voucher
3	TextView	enabled	Ada	Tidak ada	id/tv notification text
4	ImageView	enabled	Ada	Tidak ada	id/go pay logo view
5	TextView	enabled	Ada	Ada	id/go pay balance
6	ImageView	enabled	Ada	Tidak ada	id/gopay bar image
7	TextView	enabled	Ada	Tidak ada	id/gopay bar label
8	ImageView	enabled	Ada	Tidak ada	id/gopay bar image
9	TextView	enabled	Ada	Tidak ada	id/gopay bar label
10	ImageView	enabled	Ada	Tidak ada	id/gopay bar image
11	TextView	enabled	Ada	Tidak ada	id/gopay bar label
12	ImageView	enabled	Ada	Tidak ada	id/gopay bar image
13	TextView	enabled	Ada	Tidak ada	id/gopay bar label
14	ImageView	enabled	Ada	Tidak ada	id/product image view
15	TextView	enabled	Ada	Tidak ada	id/product name view
16	ImageView	enabled	Ada	Tidak ada	id/product image view
17	TextView	enabled	Ada	Tidak ada	id/product name view
18	ImageView	enabled	Ada	Tidak ada	id/product image view
19	TextView	enabled	Ada	Tidak ada	id/product name view
20	ImageView	enabled	Ada	Tidak ada	id/product image view

No	Widget	Condition	Tool	Manual Exploratory Testing	Keterangan
21	TextView	enabled	Ada	Tidak ada	id/product name view
22	ImageView	enabled	Ada	Tidak ada	id/product image view
23	TextView	enabled	Ada	Tidak ada	id/product name view
24	ImageView	enabled	Ada	Tidak ada	id/product image view
25	TextView	enabled	Ada	Tidak ada	id/product name view
26	ImageView	enabled	Ada	Tidak ada	id/product image view
27	TextView	enabled	Ada	Tidak ada	id/product name view
28	ImageView	enabled	Ada	Tidak ada	id/product image view
29	TextView	enabled	Ada	Tidak ada	id/product name view
30	ImageView	enabled	Ada	Tidak ada	id/product image view
31	ImageView	enabled	Ada	Tidak ada	id/product logo view
32	TextView	enabled	Ada	Ada	id/article title view
33	TextView	enabled	Ada	Tidak ada	id/article desc view
34	ImageView	enabled	Ada	Tidak ada	bounds [76,1802][139,1865]
35	ImageView	selected	Ada	Tidak ada	bounds [76,1802][139,1865]
36	TextView	enabled	Ada	Tidak ada	text Home
37	TextView	selected	Ada	Tidak ada	text Home
38	ImageView	enabled	Ada	Tidak ada	id/order view
39	TextView	enabled	Ada	Tidak ada	text Orders
40	ImageView	enabled	Ada	Tidak ada	bounds [508,1802][571,1865]
41	TextView	enabled	Ada	Tidak ada	text Chat
42	ImageView	enabled	Ada	Tidak ada	bounds [721,1802][784,1865]
43	TextView	enabled	Ada	Tidak ada	text Inbox
44	ImageView	enabled	Ada	Tidak ada	id/settings icon view
45	ImageView	enabled	Ada	Tidak ada	id/settings tab badge
46	TextView	enabled	Ada	Tidak ada	text Account
47	Button	clickable	Tidak ada	Ada	Kupon
48	Button	clickable	Tidak ada	Ada	Voucher
49	Button	clickable	Tidak ada	Ada	Pay
50	Button	clickable	Tidak ada	Ada	Promo
51	Button	clickable	Tidak ada	Ada	Top Up
52	Button	clickable	Tidak ada	Ada	More Gopay
53	Button	clickable	Tidak ada	Ada	Go Ride
54	Button	clickable	Tidak ada	Ada	Go Car
55	Button	clickable	Tidak ada	Ada	Go Food
56	Button	clickable	Tidak ada	Ada	Go Bluebird
57	Button	clickable	Tidak ada	Ada	Go Send
58	Button	clickable	Tidak ada	Ada	Go Deals
59	Button	clickable	Tidak ada	Ada	Go Pulsa
60	Button	clickable	Tidak ada	Ada	More
61	Button	clickable	Tidak ada	Ada	Home
62	Button	clickable	Tidak ada	Ada	Go Chat

No	Widget	Condition	Tool	Manual Exploratory Testing	Keterangan
63	Button	clickable	Tidak ada	Ada	Go Inbox
64	Button	clickable	Tidak ada	Ada	Go Account
65	ImageView	input	Tidak ada	Ada	Searching Order
66	ImageView	clickable	Tidak ada	Ada	Banner Image
67	ImageView	clickable	Tidak ada	Ada	Iklan Go Tix

Berdasarkan data dari Tabel 15, diperoleh diagram venn seperti berikut ini :



Gambar 8. Hasil Diagram Venn Home Layout

Dari gambar diatas terdapat himpunan T yang berisi jumlah seluruh widget dan condition yang diperoleh melalui tool pembangkit test case. Himpunan E berisi jumlah seluruh widget dan condition yang diperoleh melalui metode manual exploratory testing. Hasil irisan dari diagram venn diatas, diperoleh dua widget dan condition yang sama, yaitu : textview - enabled - id/go_pay_balance, textview - enabled - id/article_title_view.

Tool pembangkit test case menghasilkan widget dan condition yang bervariasi karena bersumber pada file xml. File tersebut berisi struktur dan informasi layout secara lengkap yang diperoleh secara langsung dari rekaman register layout dengan menggunakan tool UI automator viewer. Oleh karena itu, hasil widget dan condition dari tool pembangkit test case sudah mencakup seluruh keadaan pada layout. Sedangkan beberapa widget dan condition dari metode manual exploratory testing memiliki hasil yang lebih sedikit dan beririsan dengan hasil dari tool pembangkit test case. Hal ini dikarenakan mayoritas responden memilih widget yang terlihat jelas dan memilih condition sesuai dengan trigger yang diberikan pada widget tersebut.

Pada kasus ketiga ini, terdapat perbedaan mencolok pada widget. Tool UI automator viewer membaca icon pada button sebagai gambar atau desain yang diberi method :onTouchListener() sehingga hanya memiliki condition enable. Berbeda dengan hasil dari metode manual exploratory testing, dimana responden melihat icon sebagai imagebutton yang dapat diklik.

4.4 Go Ride Layout

Dari tampilan antarmuka go ride layout aplikasi gojek, tool pembangkit test case dan metode manual exploratory testing sama - sama memperoleh empat jenis widget. Rincian jenis widget dari tampilan antarmuka go ride layout aplikasi gojek dapat dilihat pada Tabel 16, dan untuk tampilan antarmuka go ride layout dapat dilihat pada lembar lampiran Gambar 13.

Tabel 16. Hasil jenis widget pada Go Ride Layout

No	Widget Tool	Widget Manual Exploratory Testing
1	ImageView	-
2	ImageButton	-
3	TextView	TextView
4	EditText	EditText
5	-	Button
6	-	Maps

Dari tampilan antarmuka go ride layout, tool pembangkit test case dan metode manual exploratory testing sama - sama memperoleh empat jenis condition. Hanya saja, jenis condition yang dihasilkan cukup berbeda. Rincian jenis condition dari tampilan antarmuka home layout dapat dilihat pada Tabel 17.

Tabel 17. Hasil Jenis Condition pada Go Ride Layout

No	Condition Tool	Condition Manual Exploratory Testing
1	Enabled	-
2	Clickable	Clickable
3	Focusable	-
4	Long-clickable	-
5	-	Input
6	-	Show
7	-	Pinch

Dari tampilan antarmuka go ride layout aplikasi gojek, tool pembangkit test case memperoleh delapan belas widget dan condition. Sedangkan metode manual exploratory testing hanya memperoleh tujuh widget dan condition. Rincian widget dan condition dari tampilan antarmuka go ride layout dapat dilihat pada Tabel 18.

Tabel 18. Hasil Widget dan Condition pada Go Ride Layout

No	Widget	Condition	Tool	Manual Exploratory Testing	Keterangan
1	ImageView	enabled	Ada	Tidak ada	bounds [13,1209][171,1267]
2	ImageButton	clickable	Ada	Tidak ada	id/fab back or menu
3	ImageButton	enabled	Ada	Tidak ada	id/fab back or menu
4	ImageButton	focusable	Ada	Tidak ada	id/fab back or menu
5	TextView	enabled	Ada	Tidak ada	id/tv card title
6	EditText	clickable	Ada	Tidak ada	id/search box
7	EditText	enabled	Ada	Tidak ada	id/search box
8	EditText	focusable	Ada	Tidak ada	id/search box
9	EditText	long-clickable	Ada	Tidak ada	id/search box
10	ImageView	enabled	Ada	Tidak ada	id/iv icon
11	TextView	enabled	Ada	Tidak ada	id/tv name
12	TextView	enabled	Ada	Tidak ada	id/tv address
13	ImageView	enabled	Ada	Tidak ada	id/iv icon
14	TextView	enabled	Ada	Tidak ada	id/tv name
15	TextView	enabled	Ada	Tidak ada	id/tv address
16	ImageView	enabled	Ada	Tidak ada	id/iv icon
17	TextView	enabled	Ada	Tidak ada	id/tv nam
18	TextView	enabled	Ada	Tidak ada	id/tv address
19	Button	clickable	Tidak ada	Ada	Back
20	EditText	input	Tidak ada	Ada	Search Place Field
21	TextView	clickable	Tidak ada	Ada	Address
22	ImageView	Show location	Tidak ada	Ada	Maps
23	ImageView	clickable	Tidak ada	Ada	Maps
24	ImageView	pinch	Tidak ada	Ada	Maps
25	TextView	clickable	Tidak ada	Ada	History

Berdasarkan data dari Tabel 18, diperoleh diagram venn seperti berikut :

Gambar 9. Hasil Diagram Venn Go Ride Layout

Dari gambar diagram venn diatas himpunan T berisi jumlah seluruh widget dan condition yang diperoleh melalui tool pembangkit test case. Himpunan E berisi jumlah seluruh widget dan condition yang diperoleh melalui metode manual exploratory testing. Untuk kasus ini, tidak ada diperoleh irisan widget dan condition untuk go ride layout. Hal ini dikarenakan sudut pandang responden dalam memilih widget dan condition berbeda dengan hasil widget dan condition yang diperoleh dari tool UI automator viewer.

4.5 Diskusi

Berdasarkan evaluasi yang telah dilakukan, diperoleh hasil seperti berikut :

1. Dari penelitian ini, tool pembangkit test case yang dibangun menghasilkan jenis widget yang lebih seragam dan konsisten seperti Button, ImageView, ImageButton, EditText, dan TextView. Condition yang diperoleh melalui tool juga lebih beragam seperti enabled, clickable, focusable. Setiap widget yang dihasilkan oleh tool memiliki keterangan unik seperti ID content description, text, dan bounds yang membedakan widget satu dengan widget yang lainnya.
2. Tool pembangkit test case sangat bergantung pada style source code programmer dan hasil rekaman hasil screenshot model.
3. Dari kasus home layout, banyak programmer yang menggunakan icon dari desainer dan diberikan method :onTouchListener() sehingga icon tersebut dapat di-klik. Akan tetapi tool UI Automator Viewer tidak dapat mendeteksi icon sebagai sebuah ImageButtton. Tool UI Automator Viewer merekam icon sebagai widget ImageView yang hanya memiliki condition enabled.
4. Widget yang diperoleh melalui metode manual exploratory testing menghasilkan jenis widget yang asal - asalan. Hal ini terjadi karena keterbatasan pengetahuan responden terhadap jenis - jenis widget. Condition yang diperoleh melalui metode manual exploratory testing sangat sedikit dengan mayoritas condition clickable. Hal ini dikarenakan responden tidak memikirkan condition lainnya selain clickable.
5. Hasil tool pembangkit test case memperoleh hasil widget dan condition yang lebih banyak karena setiap widget memiliki minimal satu condition dan maksimal memiliki empat condition. Widget dan condition yang diperoleh dengan metode manual exploratory testing lebih sedikit karena memiliki minimal satu condition dan memiliki maksimal dua condition

5. Kesimpulan

Kesimpulan yang diperoleh dari penelitian ini adalah tool pembangkit test case menghasilkan widget dan condition yang lebih beragam sehingga test case yang dihasilkan pun jadi lebih banyak. Tool pembangkit test case pada kasus antarmuka welcome layout aplikasi gojek menghasilkan 17 pasang widget dan condition, antarmuka register layout menghasilkan 27 pasang widget dan condition, antarmuka home layout menghasilkan 46 pasang widget dan condition, dan antarmuka go ride layout menghasilkan 18 pasang widget dan condition. Lalu hasil dari metode manual exploratory testing pada kasus antarmuka welcome layout aplikasi gojek menghasilkan 9 pasang widget dan condition, antarmuka register layout menghasilkan 16 pasang widget dan condition, antarmuka home layout menghasilkan 24 pasang widget dan condition, dan antarmuka go ride layout menghasilkan 17 pasang widget dan condition. Tool pembangkit test case memiliki kelemahan dimana widget dan condition untuk test case sangat bergantung pada rekaman hasil screenshot model. Saran yang dapat diberikan untuk penelitian selanjutnya, yakni tool pembangkit test case untuk antarmuka aplikasi android dapat dikembangkan dengan memperhatikan file .java agar perpindahan layout di aplikasi android menjadi bagian dari test case.

Daftar Pustaka

[1] A. Abran, J. W. Moore, P. Bourque, R. Dupuis, and L. Tripp. Software engineering body of knowledge. IEEE Computer Society, Angela Burgess, 2004.

[2] P. Aho, T. Kanstrén, T. Räty, and J. Röning. Automated extraction of gui models for testing. In Advances in Computers, volume 95, pages 49–112. Elsevier, 2014.

[3] android. Ui automator. https://developer.android.com. Accessed: 2019-06-26.

[4] G. Bae, G. Rothermel, and D.-H. Bae. On the relative strengths of model-based and dynamic event extraction-based gui testing techniques: An empirical study. In 2012 IEEE 23rd International Symposium on Software Reliability Engineering, pages 181–190. IEEE, 2012.

[5] Y.-M. Baek and D.-H. Bae. Automated model-based android gui testing using multi-level gui comparison criteria. In Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering, pages 238–249. ACM, 2016.

[6] I. Banerjee, B. Nguyen, V. Garousi, and A. Memon. Graphical user interface (gui) testing: Systematic mapping and repository. Information and Software Technology, 55(10):1679–1694, 2013.

[7] A. R. Espada, M. d. M. Gallardo, A. Salmerón, and P. Merino. Using model checking to generate test cases for android applications. arXiv preprint arXiv:1504.02440, 2015.

[8] T. D. Hellmann and F. Maurer. Rule-based exploratory testing of graphical user interfaces. In 2011 Agile Conference, pages 107–116. IEEE, 2011.

[9] I. Hooda and R. Chhillar. A review: Study of test case generation techniques. International Journal of Computer Applications, 107(16), 2014.

[10] J. Itkonen, M. V. Mantyla, and C. Lassenius. How do testers do it? an exploratory study on manual testing practices. In 2009 3rd International Symposium on Empirical Software Engineering and Measurement, pages 494–497. IEEE, 2009.

[11] M. E. Khan, F. Khan, et al. A comparative study of white box, black box and grey box testing techniques. Int. J. Adv. Comput. Sci. Appl, 3(6), 2012.

[12] V. Lelli, A. Blouin, and B. Baudry. Classifying and qualifying gui defects. In 2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST), pages 2–3. IEEE, 2015.

[13] A. Mateen and K. Abbas. Optimization of model based functional test case generation for android applications. In 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), pages 90–95. IEEE, 2017.

[14] tutlane. Android view and viewgroup with examples. ://www.tutlane.com/tutorial/android/android-view-and-viewgroup-with-examples.

[15] A. Usman, N. Ibrahim, and I. A. Salihu. Test case generation from android mobile applications focusing on context events. In Proceedings of the 2018 7th International Conference on Software and Computer Applications, pages 25–30. ACM, 2018.