BAB II
TINJAUAN PUSTAKA

Penelitian ini akan melibatkan beberapa aspek yang relevan untuk membandingkannya dengan penelitian – penelitian sebelumnya yang telah di dijalankan.

2.1 Penelitian Terdahulu

2.1.1 Penelitian Pertama

Penelitian yang dilaksanakan oleh (Kusyadi, Mulyati, Setiany, Noviyanto, & Aisah, 2022) dengan judul ***"Pengujian Aplikasi Kas Keuangan Menggunakan Katalon"*** meneliti tentang pengujian aplikasi kas keuangan, dimana aplikasi tersebut adalah aplikasi ini dirancang untuk melaporkan arus kas baik dalam bentuk pengeluaran ataupun penerimaan kas dalam suatu periode. Aplikasi ini juga memberikan informasi tentang sumber-sumber yang menjadi penyebab keluarnya dan masuknya kas tersebut. Tujuan utama dari aplikasi ini ialah untuk menguji kualitas website serta aplikasi kas keuangan dipastikan berjalan tanpa adanya kesalahan (*error*). Selain itu, aplikasi ini juga melakukan analisis terhadap tampilan web untuk mengevaluasi sejauh mana kemudahan pemahaman bagi *user*. Namun, ruang lingkup pengujian ini mencakup pengukuran kualitas perangkat lunak atau web melalui penggunaan metode pengujian antarmuka grafis (GUI) menggunakan alat bantu Katalon Studio. Aspek pengujian yang dibahas meliputi *backend performance test, frontend performance test, functional test,* serta *user Interface test*. Dengan demikian, kesimpulan dari penelitian ini adalah bahwa Katalon Studio sangat membantu dalam pengujian dan mengoptimalkan fitur-fitur utilitas pengujian website. Perbedaan dengan penelitian yang akan dilakukan yaitu pengujian dilakukan dengan *metode behavior driven development* dalam implementasi *automation test*, terdapat juga *coverage reporting* untuk mengetahui laporan hasil pengujian.

2.1.2 Penelitian Kedua

Penelitian yang dilakukan oleh (Rusli, 2020) dengan judul ***"Analisa Perbandingan Black-box Automated Testing dan Manual Testing pada Aplikasi ACCMART"*** meneliti tentang pengujian bug pada aplikasi *ACCMart* yang sedang dalam tahap pengembangan dan untuk membandingkan perbedaan antara pengujian manual oleh pengguna dan pengujian otomatis. *ACCMart* adalah sebuah aplikasi yang dikembangkan oleh perusahaan Astra Credit Companies untuk memfasilitasi transaksi jual beli mobil dan lelang. Dalam penelitian ini, ruang lingkup pengujian terbatas pada aplikasi *ACCMart* dan hanya sejumlah fungsi yang telah dipilih sebelumnya yang akan diuji, yaitu 11 dari total 25 fungsi yang tersedia dalam aplikasi tersebut. Untuk melakukan pengujian, digunakan alat pengujian otomatis Katalon Studio dengan menerapkan metode *black-box*. Setelah melakukan penelitian, dapat diambil kesimpulan bahwa membandingkan antara pengujian otomatis dan manual adalah sebagai berikut. Pengujian otomatis dengan Katalon Studio membutuhkan waktu yang lebih lama, dengan selisih waktu rata-rata 61,47%, dibandingkan dengan pengujian manual, itu merupakan temuan yang menarik.. Hal ini disebabkan oleh beberapa faktor penting, seperti proses memulai aplikasi dan waktu yang dibutuhkan untuk langkah-langkah lainnya. Perbedaan dengan penelitian yang akan dilakukan adalah pengujian dilakukan secara *automation test* dengan *project* yang dalam proses pengembangan masih berjalan 50%, pengujian hanya *focus* pada pengujian *website* dengan ketentuan fitur prioritas dengan *metode behavior driven development* dan hasil reporting akan dihasilkan secara otomatis dengan *tools katalon studio*.

2.1.3 Penelitian Ketiga

Penelitian yang dijalankan oleh (Kokasih & Cahyono, 2021) dengan judul ***"Automation testing tool dalam pengujian aplikasi the point of sale"*** meneliti tentang pengujian *automation* pada aplikasi TPOS. Aplikasi TPOS ialah sebuah situs web yang diaplikasikan untuk menyelenggarakan penjualan produk di berbagai toko di seluruh Indonesia. Tujuan dari penelitian ini ialah untuk menguji kualitas perangkat lunak sesuai metode pengujian *Blackbox* pada aplikasi *The Point Of Sale* (TPOS). Penelitian ini akan mencakup pengujian kualitas perangkat lunak berbasis web dengan memakai metode pengujian *Blackbox* yang fokus pada beberapa aspek pengujian, termasuk pengukuran perangkat lunak sesuai uji fungsional, pengukuran kinerja antarmuka depan, pengukuran kinerja bagian belakang aplikasi, dan *user interface test*. Dengan demikin, kesimpulan dari penelitian tersebut yaitu kebutuhan *testing* yang dilakukan pada aplikasi *the point of sale*, jika diperlukan pengujian yang berulang-ulang atau melibatkan berbagai platform dan data yang besar, sebaiknya menggunakan *skrip otomatisasi*. Namun, jika pengujian hanya dilakukan sekali atau membutuhkan aspek perasaan atau pengalaman langsung, pengujian manual juga dapat dipertimbangkan, terutama dalam situasi darurat. Disarankan untuk menggunakan RAM 8GB, karena berdasarkan pengalaman, saat menggunakan RAM 4GB, proses pemutaran kembali (*playback*) di Katalon dapat terhenti dan pengujian selalu menghasilkan kegagalan (*fail*). Setelah meningkatkan RAM menjadi 8GB, Katalon tidak mengalami masalah sama sekali selama pengujian. Perbedaan dengan penelitian yang akan dilakukan adalah pengujian dilakukan dengan *automation test* pada tahap pengembangan *project* pada aplikasi berbasis *website* dengan metode *behavior driven development (bdd)* dan mengutamakan pengujian secara berulang yang menghasilkan laporan *bug* secara otomatis secara cepat didapat.

2.1.4 Penelitian Keempat

Penelitian yang dilakukan (Zualianto, et al., 2021) dengan judul ***"Pemanfaatan Katalon Studio Untuk Otomatisasi Pengujian Black-Box Pada Aplikasi Iposyandu"*** meneliti tentang pengujian otomatis pada aplikasi Iposyandu. Aplikasi iPosyandu dibuat dengan tujuan untuk memberikan kemudahan bagi kader dalam menyusun laporan mengenai aktivitas di posyandu dan membuat laporan tersebut. Tujuan dari penelitian ini yaitu Melaporkan hasil pengujian otomatis menggunakan alat pengujian untuk aplikasi iPosyandu, terutama untuk aplikasi berbasis *website*. Akhirnya tujuannya untuk *quality assurance* aplikasi iPosyandu. Ruang lingkup pada penelitian ini yaitu pengujian melibatkan lebih dari sekadar debugging. Selain digunakan untuk menemukan dan memperbaiki cacat, pengujian juga berperan dalam validasi, verifikasi proses, dan pengukuran reliabilitas pengujian pada level unit, integrasi ataupun pengujian sistem. Dengan demikian, hasil penelitian menunjukkan bahwa pengujian mengungkapkan adanya fungsionalitas yang perlu diperbaiki serta kebutuhan untuk menambahkan fungsionalitas lain. Pemanfaatan Katalon Studio membantu dalam meningkatkan efektivitas pengujian dan mengurangi kesalahan manusia yang mungkin terjadi saat pengujian dilakukan secara manual. Mendapatkan laporan hasil pengujian menjadi lebih mudah karena Katalon Studio menyediakan Katalon *TestOps*. Pengujian untuk aplikasi *mobile* perlu dilakukan, terutama aplikasi iPosyandu dan digunakan oleh kalangan kader. Perbedaan dengan penelitian yang akan dilakukan adalah pengujian untuk aplikasi berbasis website dengan *automation testing* menggunakan metode *behavior driven development* dalam implementasi dan menggunakan *coverage reporting* untuk mengetahui hasil uji secara otomatis.

2.2 Teori Terkait

2.2.1 Testing

Testing ialah pengujian adalah cara untuk mengevaluasi kinerja suatu entitas atau sistem. Dalam konteks manusia, pengujian digunakan sebagai alat untuk mengukur tingkat keterampilan atau pengetahuan yang telah diperoleh seseorang. Pada proses mengembangkan suatu perangkat lunak *(software)*, pengujian diaplikasikan sebagai langkah penting dalam proses keseluruhan untuk memastikan apakah tujuan telah tercapai. Sebagai contoh, dalam mengembangkan suatu perangkat lunak *(software)*, pengguna *software* tersebut akan mencoba produk dalam sehari-hari untuk menguji fungsionalitas, kegunaan,dan kepuasan pengguna. Setelah desain selesai, pengkodean dijalankan, dan kemudian *code software* akan dilakukan pengujian *unit test* oleh programmer dan tim QA (*Quality Assurance*) (Rouse, 2020).

2.2.1.1 *Software Testing*

Software testing adalah proses eksekusi program atau aplikasi untuk mendeteksi keberadaan *bug* dalam *software* yang sedang dikembangkan (Suhartono, 2016). Definisi lain dari pengujian perangkat lunak adalah proses yang dijalankan guna memperoleh pemahaman yang lebih baik tetang kemungkinan risiko yang terkait dengan *software* yang sedang diuji serta penemuan *bug* atau kesalahan dalam *software* tersebut. (Dhuha, 2018). *Software testing* juga merupakan kegiatan pemeriksaan sistem *software* untuk memastikan bahwa tidak ada *bug* atau cacat produk tersebut, agar hal tersebut dapat diidentifikasi dan diperbaiki sebelum *release* kepada pengguna akhir (*end-user*) (Sitemaster, 2019).

2.2.2 Manual Testing

Manual testing merupakan metode pengujian di mana seorang pengujian menyediakan kasus uji secara manual dan menjalankannya untuk mengidentifikasi cacat dalam perangkat lunak (Dobles, 2019). Pada umumnya, penggunaan *manual*

testing dalam pelaksanaan end-to-end testing melibatkan eksekusi pengujian dengan menjalankan perangkat lunak sesuai dengan skenario yang tertera pada kasus uji. Setelahnya, dilakukan membandingkan antara output yang diharapkan dari setiap *test cases* dan output yang dihasilkan oleh aplikasi.

2.2.3 Automation Testing

Untuk menguji program komputer secara automatis, *script* uji coba dibuat menggunakan bahasa skrip seperti *java script*, *phyton*, atau *tool command language* (TCL), sehingga kasus uji coba dapat dilakukan oleh komputer dengan sedikit campur tangan manusia (Dudekula, 2011) Selain itu, diuraikan bahwa pengujian software otomatif ialah proses pembuatan sebuah program, atau *script test*, yang memengaruhi langkah-langkah tes kasus manual dalam bahasa pemrograman apa pun dengan *external automation helper tool*. menggunakan program komputer untuk melakukan tes *end to end*.

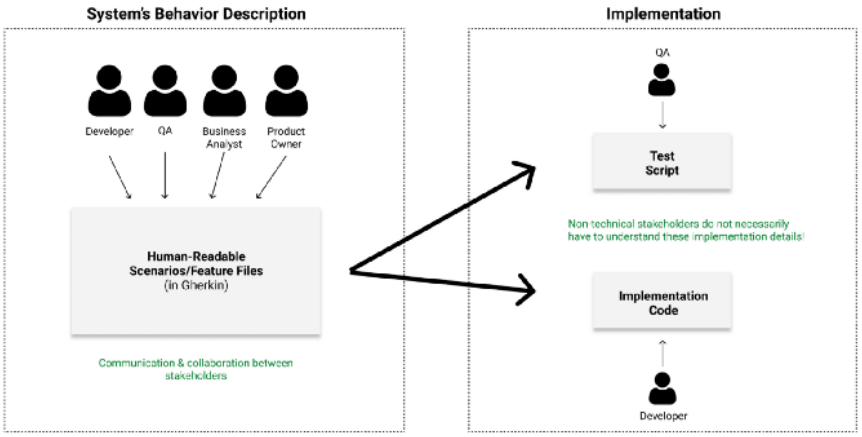
Menurut hasil penelitian sebelumnya oleh (Kokasih & Cahyono, 2021), bahwa *automation test* memungkinkan pengulangan urutan perintah pengujian, yang memungkinkan pengujian menjadi lebih cepat, mengulang kegagalan, dan memungkinkan pengujian otomatis untuk mencatat waktu respons secara otomatis.

2.2.4 Behavior Driven Development

Teknik *Behavior Driven Development (BDD)* menurut (Hatko, Mersmann, & dan Puppe, 2014) ialah antara lain: *"Test cases are produced before the actual software system under the agile Software Engineering (SE) process known as "Behavior-Driven Development" (BDD). By using a pattern matching method, BDD directly derives executable test cases from requirements expressed in natural language"*. Berarti *Behavior Driven Development* (BDD) adalah jenis pengembangan perangkat lunak agile menggunakan prinsip uji coba pertama, di mana kasus uji dibuat sebelum sistem dibuat. Test case pada BDD dilandaskan pada persyaratan yang disatukan oleh *stakeholder*.

Menurut (Cucumber, n.d.) BDD mendorong bekerja dalam iterasi yang cepat, terus-menerus memecah masalah pengguna menjadi potongan-potongan kecil yang dapat mengalir melalui proses pengembangan secepat mungkin.

Prinsip utama BDD adalah mendeskripsikan tindakan sebuah sistem tanpa menjelaskan penerapannya.



Gambar 2. 1 Ilustrasi Konsep dasar prinsip BDD

Dalam pendekatan BDD, semua *stakeholders* (baik yang berkecimpung di bidang teknis maupun non-teknis) bekerja sama untuk menggambarkan kebutuhan pengguna dan kebutuhan fungsional (*behavior*) sistem dengan file-fitur (*feature files*). Deskripsi yang terdapat dalam file-fitur berfungsi bagi logika pengujian, yang menjadi dasar untuk *Quality Assurance* (QA) dalam menulis skrip pengujian untuk penerapannya, serta menjadi landasan bagi pengembang dalam menulis kode implementasi untuk membangun sistem.

Pada penulisan *feature file test* pada *Behavior Driven Development (BDD)* terdapat 3 *keyword* sebagai elemen utama dalam skenario BDD:

1. *Given*, mengartikan konteks dari skenario
2. *Then*, mengartikan *outcome* dari perilaku yang dijalankan oleh *user*
3. *When*, mengartikan perilaku yang dijalankan oleh *user*

Skenario yang terdapat dalam file-fitur tersebut akan diaplikasikan untuk petunjuk dalam implementasi *skrip* pengujian yang disebut "*steps*" dalam pendekatan BDD.

2.2.5 *Test Driven Development*

Test Driven Development (TDD) adalah salah satu praktik umum dalam metodologi pengembangan Agile (Khanam & Ahsan, 2017). Dalam metode TDD, pengembang diminta untuk mengidentifikasi proses bisnis dan merancang perangkat lunak yang dibutuhkan sebelum mulai menulis kode. Pendekatan TDD digunakan untuk mengembangkan perangkat lunak dengan menggabungkan desain dan pengujian logika program. Dengan demikian, metode ini dapat dianggap sebagai penggabungan dari pendekatan "*Test First Development"*, di mana pengujian unit dilakukan sebelum menulis kode implementasi.

2.2.6 Katalon Studio

Dengan Katalon Studio, *Quality Assurance* dapat lebih mudah menguji aplikasi yang sedang dikembangkan (Imam, 2019). Katalon Studio merupakan teknologi KMS menghasilkan solusi otomatisasi yang sederhana namun efektif untuk penguji di seluruh dunia. Dengan menyederhanakan kerangka kerja otomatisasi uji sumber terbuka seperti Selenium dan Appium, pengembang dapat dengan mudah mengatasi hambatan teknis yang mungkin mereka hadapi. Ini memungkinkan mereka untuk fokus pada pengembangan tes yang efektif dan efisien tanpa harus menghabiskan banyak waktu dan upaya dalam menyiapkan dan mengkonfigurasi kerangka kerja, Katalon Studio mengubah penggunaan mereka untuk melaporkan, menjalankan, mengatur, membuat, serta mengelola pengujian otomatis dengan *efisiensi*. Katalon Studio memiliki keunggulan utama dalam hal kemudahan penggunaan dan integrasi yang lebih baik dibandingkan dengan Selenium, yang merupakan salah satu pemimpin pasar di bidang otomatisasi uji.

Selain itu menurut penelitian sebelumnya oleh (Zualianto, et al., 2021), katalon studio terdapat beberapa batasan yang perlu diperhatikan diantaranya beberapa set fiturnya masih dalam tahap pengembangan dan penyempurnaan, memiliki komunitas pengguna yang aktif, namun dukungan komunitasnya mungkin tidak sebesar yang dimiliki oleh beberapa alat otomatisasi uji lainnya.