

Pengujian Black Box pada Aplikasi Penghitungan Parkir Swalayan ADA Menggunakan Teknik Equivalence Partitions

Bangkit Fajar Nugraha¹, Fahmi Aditama², Muhammad Arrofi³, Solakhul Umam Ahmad⁴, Yulianti Yulianti⁵

Teknik Informatika, Universitas Pamulang, Jl. Raya Puspitak No.46, Buaran, Serpong, Tangerang Selatan, Banten, Indonesia, 15310

e-mail: ¹bangkitfajar20@gmail.com, ²fahmiaditama354@gmail.com, ³arrofinita21@gmail.com, ⁴solakhulumam12@gmail.com, ⁵yulianti@unpam.ac.id

Submitted Date: June 03rd, 2020

Reviewed Date: June 17th, 2020

Revised Date: June 30th, 2020

Accepted Date: June 30th, 2020

Abstract

To ensure that the application (software) Calculation of parking is no error then in need of testing to test the quality of the application. This application has problems with the calculation of the parking in the application, then the application will be tested by the Black Box method based on Equivalence where the application in the overall test is related to the use, benefit, and result of the utilization of such software. The parking application must be error-free, because if there is an error then it will be very detrimental to the self-service that use the parking application, specifically in terms of financial income. The Black Box-based method of Equivalence Partition tests the quality of the application by documenting the software testing which will look at the error in each form, and the errors will be divided into three models, errors, functions, data structures and interfaces. The results of this test, to ensure the quality of the free parking calculation application error by finding an accidental error in the parking calculation application.

Keywords: Testing; Software; Black Box; Equivalence Partition

Abstrak

Untuk menjamin aplikasi (*software*) Penghitungan Parkir yang tidak ada kesalahan maka diperlukan pengujian untuk menguji kualitas aplikasi. Aplikasi ini memiliki masalah pada penghitungan parkir diaplikasi, maka aplikasi akan diuji dengan Metode *Black Box* berbasis *Equivalence* di mana aplikasi diuji secara keseluruhan terkait dengan penggunaan, manfaat, dan hasil dari pemanfaatan perangkat lunak tersebut. Aplikasi parkir harus bebas dari kesalahan, karena jika terdapat kesalahan maka hal tersebut akan sangat merugikan pihak swalayan yang menggunakan aplikasi parkir, khususnya dalam hal pemasukan keuangan. Metode *Black Box* berbasis *Equivalence Partition* menguji kualitas aplikasi dengan cara melakukan dokumentasi dari pengujian perangkat lunak yang mana akan terlihat kesalahannya pada setiap form, dan kesalahan akan dibagi menjadi tiga model di antaranya, kesalahan fungsi, struktur data dan *interface*. Hasil dari pengujian ini, untuk menjamin kualitas aplikasi penghitungan parkir bebas dari kesalahan dengan menemukan adanya kesalahan yang tidak disengaja pada aplikasi penghitungan parkir.

Kata Kunci: Pengujian; *Software*; *Black Box*; *Equivalence Partition*

1. Pendahuluan

Di zaman yang serba modern ini tentu semua hal dalam bidang apapun sangat membutuhkan keakuratan yang mencapai 100% di mana hal tersebut sangat menentukan dalam hasil yang ingin diperoleh. Termasuk dalam

bidang parkir, tidak banyak yang mengetahui bahwa bidang parkir di suatu tempat adalah bidang yang sangat membantu dalam hal menambah penghasilan bagi pemilik tempat. Swalayan ADA contohnya, pengunjung yang datang ke swalayan memang tujuan utamanya

adalah untuk berbelanja, namun hampir rata-rata dari pengunjung pasti membawa kendaraan yang memerlukan tempat parkir untuk memarkir kendaraannya, dengan keadaan tersebut pihak swalayan dapat memanfaatkan lahan yang ada untuk mendapatkan penghasilan tambahan dari biaya parkir kendaraan pengunjung, namun dikarenakan pengunjung yang banyak dan jumlah kendaraan pun banyak, maka sudah tidak memungkinkan lagi untuk menggunakan cara konvensional, dan pihak swalayan memerlukan sebuah aplikasi yang membantu menghitung biaya parkir kendaraan, dengan akurasi penghitungan 100%. Maka dari itu, perlu dilakukannya pengujian terhadap aplikasi parkir yang sudah ada di swalayan, sehingga aplikasi yang digunakan dapat berfungsi sesuai dengan fungsinya dan juga semua *tools* yang ada di dalam aplikasi parkir berjalan dengan semestinya.

Aplikasi penghitungan parkir merupakan cara penghitungan untuk menentukan Biaya parkir dari sebuah kendaraan yang masuk ke wilayah parkir swalayan ADA. Jadi, Sistem yang dikembangkan bertujuan untuk menghasilkan penghitungan terhadap biaya parkir, yang akurat serta dengan proses yang cepat dan mudah digunakan. Dalam aplikasi penghitungan parkir ini terdapat 2 proses yang dilakukan yaitu, proses memasukkan kendaraan saat memasuki parkir, dan proses kendaraan keluar parkir, di mana diproses parkir masuk, user harus memasukkan nomer polisi dan jenis kendaraan yang akan masuk ke wilayah parkir, dan data kendaraan tersebut akan masuk ke database. Dan diproses parkir keluar, terdapat proses mencari nomor polisi yang sudah ada di dalam database yang sebelumnya dimasukkan di dalam proses parkir masuk, lalu secara otomatis keluar biaya yang harus dibayar dan berapa lama waktu parkir.

Pada aplikasi penghitungan parkir terdapat kesalahan pada saat proses keluar kendaraan, kendaraan yang sudah keluar tidak terhapus dari database dan waktu yang muncul tidak sesuai dengan waktu *realtime* saat ini sehingga merugikan pihak swalayan dalam penghitungan biaya parkir. Maka, aplikasi ini perlu dilakukan pengujian agar kesalahan yang tidak disengaja dapat ditemukan, dan aplikasi ini dapat berjalan sesuai dengan fungsinya, sehingga lebih meminimalisir kesalahan dalam penghitungan biaya, sehingga tidak memunculkan kerugian bagi pengguna aplikasi parkir ini.

Untuk menjamin aplikasi penghitungan parkir tidak ada kesalahan maka, aplikasi ini perlu diuji agar kesalahan yang tidak disengaja dapat ditemukan. Tujuan dari pengujian adalah untuk memastikan bahwa aplikasi yang dihasilkan sesuai dengan kebutuhan dan layak untuk digunakan (Arwaz, Putra, Putra, Kusumawijaya, & Saifudin, 2019).

Pengujian ini akan menggunakan metode pengujian *Black Box* yang memiliki arti bahwa pengujian ini hanya memeriksa suatu perangkat lunak dari hasil yang sudah dieksekusi, tanpa harus tau lebih detail ke dalam source dari perangkat lunak tersebut, dengan kata lain, bahwa kita hanya melihat secara fungsionalnya saja apakah sudah berfungsi sesuai dengan fungsinya. *Black Box* adalah teknik pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, penguji dapat mendefinisikan kumpulan kondisi masukan dan melakukan pengetesan pada spesifikasi fungsional program (Hidayat & Muttaqin, 2018). *Black Box* hanya mengevaluasi bagian dari tampilan antarmuka (*interface*) dan fungsional tanpa mengetahui apa yang terjadi dalam proses detailnya yang berarti bahwa hanya dapat mengetahui masukan dan keluaran saja, maka dilakukan langkah pemeriksaan untuk memastikan bahwa data tersebut telah sesuai dengan yang ditetapkan dan bertujuan untuk memastikan bahwa data akan dimasukkan ke dalam basis data itu telah diketahui dan dapat dijelaskan kebenaran datanya (MZ, 2016).

Black Box Testing bukanlah solusi alternatif dari *White Box Testing* tapi lebih sebagai pelengkap untuk menguji hal-hal yang tidak dicakup oleh *White Box Testing*. *Black Box Testing* mengidentifikasi jenis kesalahan dalam beberapa kategori antara lain fungsi yang tidak benar atau tidak ada, kesalahan antarmuka (*interface errors*), kesalahan pada struktur data dan akses basis data, kesalahan performansi (*performance errors*) dan kesalahan inisialisasi maupun terminasi (Mustaqbal, Firdaus, & Rahmadi, 2015).

Metode *Equivalence Partitions* adalah metode dari teknik pengujian *Blackbox* dengan cara domain masukan dari suatu program dipecah atau dibagi ke dalam kelas-kelas data agar memperoleh *Test Case*. Perancangan *Test Case Equivalence Partitions* didapat dari evaluasi kelas *Equivalence* yang menggambarkan kondisi masukan, kumpulan keadaan yang valid atau tidak. Kondisi masukan biasanya berupa nilai *numeric*, *range* nilai, dan kumpulan nilai-nilai yang

berhubungan dengan kondisi *Boolean*. Semua tahapan akan dibahas pada pembahasan selanjutnya.

2. Metodologi

Pengujian adalah hal yang harus dilakukan seorang programmer untuk menghasilkan perangkat lunak yang dibangun bebas dari kesalahan dan semakin berkualitas. Pengujian perangkat lunak merupakan sebuah proses pengujian program yang dimaksudkan untuk mencari kesalahan pada software dan memperbaikinya sehingga memiliki kualitas yang baik (Febrian, Ramadhan, Faisal, & Saifudin, 2020). Melalui pengujian, seorang programmer dapat mengetahui kekurangan dan kesalahan yang terdapat di dalam sebuah sistem sehingga dapat segera memperbaikinya. Tujuannya adalah menyesuaikan fungsi-fungsi dengan spesifikasi yang diinginkan (Andriansyah, 2018).

Pengujian *software* sangat diperlukan untuk memastikan apakah aplikasi (*software*) yang sedang dibuat bebas dari kesalahan dan berjalan sesuai yang diharapkan. Pengujian harus dilakukan secara teliti dan lengkap, karena jika pengujian yang dilakukan tidak lengkap dan efektif dapat menimbulkan masalah dan menyebabkan kerugian ketika software digunakan (Rosalina, Rassi, Hadi, Ubaidillah, & Desyani, 2020). Penguji *software* harus menyiapkan sesi khusus untuk menguji program yang sudah dibuat agar kesalahan ataupun kekurangan dapat dideteksi sejak awal dan dikoreksi secepatnya. (Shi, 2010).

Pengujian *Black Box* bertujuan menemukan klasifikasi kesalahan dalam beberapa kategori, yaitu fungsi-fungsi yang salah atau hilang, kesalahan *interface*, kesalahan dalam struktur data atau akses database eksternal, kesalahan performa, kesalahan dan terminasi (Agarwal, Tayal, & Gupta, 2010). Dalam pengujian *Black Box Testing* digunakan alat untuk pengumpulan data yang disebut dengan *user acceptance test*, dokumen ini terdiri deskripsi indikator dari prosedur pengujian fungsionalitas dari perangkat lunak. Dengan metode *Black Box Testing* ini, pengembang perangkat lunak parkir ini dapat menguji keseluruhan fitur atau menu yang terdapat pada sistem tersebut (Setiyani, 2019). Metode *Equivalence Partitions* merupakan pengujian berdasarkan masukan setiap menu yang terdapat pada sistem Aplikasi Parkir di setiap menu masukan dilakukan pengujian melalui

klasifikasi dan pengelompokan berdasarkan fungsinya (Hidayat & Muttaqin, 2018).

Penelitian ini akan dilakukan beberapa tahapan. Pada tahapan pertama diawali dengan menentukan *Test Case* perangkat lunak yang akan diuji dengan metode *Equivalence Partitions* kemudian menginisialisasi *standar grade partition* masukan dan keluaran. Hal ini dilakukan untuk mendapatkan dataset berupa dokumentasi pengujian dengan metode *Equivalence Partitions* dan nilai tingkat efektifitas metode *Equivalence Partitions*.

Berdasarkan form dan teknik pengujian *Black Box* dibuat tabel rancangan *Test Case* yang berfungsi untuk menyimpulkan apakah sistem berhasil dalam pengujian tipe tersebut atau tidak. Berikut terdapat penjelasan dari beberapa tabel Rancangan *Test Case* berdasarkan *Equivalence Partition*.

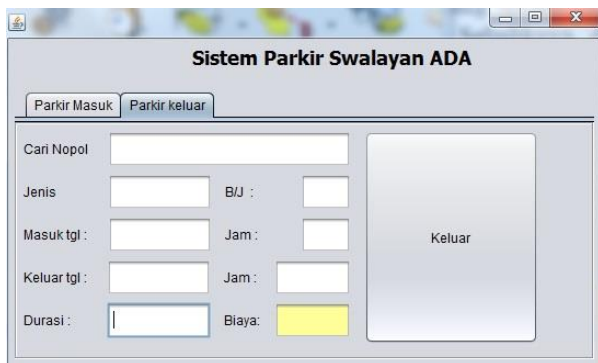
Pada form pada Gambar 1 terdapat beberapa rencana pengujian. Rencana pengujian *no polisi* data akan valid jika *ID* diisi dengan memasukkan kombinasi alfabet dan angka "B 23476 TU" yang terdiri dari maksimal 15 digit dan tidak boleh kosong. Sebaliknya, data tidak valid jika memasukkan hanya abjad atau angka saja. Pada rencana pengujian Jenis kendaraan data akan valid jika Jenis dipilih salah satu dari jenis kendaraan yang sudah ada. Sebaliknya, data tidak valid jika field jenis kendaraan tidak dipilih, dan akan muncul catatan "harap pilih jenis kendaraan" dan Pada saat user klik tombol simpan, nomor polisi yang sudah dimasukkan akan masuk ke database dan akan ditampilkan pada tabel, dan akan muncul tanggal masuk pada tabel, tanggal masuk sesuai dengan tanggal *realtime* pada saat kendaraan masuk.



Gambar 1. Form parkir masuk

Tabel 1. Rancangan Test Case Form Parkir Masuk

Id	Deskripsi Pengujian	Hasil yang Diharapkan
PM 01	Mengisi <i>No Polisi</i> dengan “B 23476 TU”, Dan pilih jenis kendaraan dengan “Motor” kemudian klik tombol “Simpan”	Sistem berhasil menerima penambahan Data, dan muncul Tanggal masuk sesuai dengan Tanggal Realtime
PM 02	Mengisi <i>No Polisi</i> dengan “123” atau tidak diisi, dan jenis tidak dipilih kemudian klik tombol Daftar	Sistem menolak untuk menyimpan data dan muncul notifikasi



Gambar 2. Form Parkir Keluar

Pada form pada Gambar 2 terdapat beberapa rencana pengujian. Rencana pengujian cari nopol data akan valid jika nopol yang dicari sudah dimasukkan pada form Parkir masuk dan tidak boleh kosong. Setelah memasukkan data

pencaharian nopol selesai dan ditekan enter, maka akan Muncul Jenis kendaraan, Tanggal Masuk, dan jam masuk, sesuai dengan yang sudah dimasukkan di form parkir masuk. Ketika user klik tombol keluar, maka sistem akan memproses data kendaraan keluar, dan dihapus dari database, akan muncul jam keluar dan tanggal keluar sesuai dengan Tanggal realtime, biaya pun akan dihitung secara otomatis dari penghitungan waktu masuk sampai dengan waktu keluar. Sebaliknya data tidak akan valid jika field Cari nopol kosong atau nomer polisi belum dimasukkan pada form parkir masuk, dan akan muncul notifikasi “Data tidak ditemukan”. Jika Field cari nopol masih kosong tombol keluar pun tidak akan berfungsi untuk memproses hapus data dari database, dan menghitung biaya parkir yang harus dibayarkan.

Tabel 2. Rancangan Test Case Form Parkir Keluar

Id	Deskripsi Pengujian	Hasil yang Diharapkan
PK 01	Mengisi <i>Cari nopol</i> dengan memasukkan nomor polisi “B 23476 TU” sesuai dengan nomor polisi pada form parkir masuk, dan enter	Sistem mampu Menampilkan Jenis kendaraan, Tanggal masuk, dan jam Masuk
PK 02	Klik Tombol “Keluar” setelah nomor polisi yang dimasukkan sesuai dengan nomor polisi yang dimasukkan di form parkir masuk	Sistem mampu memproses data kendaraan dihapus dari database, dan menampilkan Jam keluar, durasi, dan biaya parkir
PK 03	Mengisi <i>Cari nopol</i> dengan “123” tidak sesuai dengan nomor polisi pada form parkir masuk, atau tidak diisi sama sekali, dan enter	Sistem menolak menampilkan data dan muncul noti “data tidak ada”
PK 04	Klik Tombol “Keluar” saat nopol yang dimasukkan salah atau tidak diisi	Sistem menolak memproses data

3. Hasil dan Pembahasan

Berdasarkan rencana pengujian yang telah disusun, dan direncanakan maka dapat dilakukan pengujian sebagai berikut:

Tabel 3. Hasil Pengujian

Id	Deskripsi Pengujian	Hasil yang diharapkan	Hasil Pengujian	Kesimpulan
K1	Mengisi <i>No Polisi</i> dengan “B 23476 TU”, Dan pilih jenis kendaraan dengan “Motor” kemudian klik tombol “Simpan”	Sistem berhasil menerima penambahan data, dan muncul Tanggal masuk sesuai dengan Tanggal Realtime	Tampil Pop Up data Berhasil disimpan, dan lanjut untuk memasukkan nopol selanjutnya	Tidak sesuai
K2	Mengisi <i>No Polisi</i> dengan “123” atau tidak diisi	Sistem menolak untuk menyimpan data dan muncul notifikasi	Tampilan muncul notifikasi “data tidak sesuai”	Sesuai
K3	Mengisi <i>Cari nopol</i> dengan “B 23476 TU”, dan enter	Sistem mampu Menampilkan Jenis kendaraan, Tanggal masuk, dan jam Masuk	Sistem mampu Menampilkan Jenis kendaraan, Tanggal masuk, dan jam Masuk	Sesuai
K4	Klik Tombol “Keluar” setelah nomor polisi yang dimasukkan sesuai dengan nomor polisi yang dimasukkan di form parkir masuk	Sistem mampu memproses data kendaraan dihapus dari database, dan menampilkan Jam keluar, durasi, dan biaya parkir	Sistem mampu memproses data kendaraan dihapus dari database, dan menampilkan jam keluar, namun durasi, dan biaya parkir tidak muncul	Tidak Sesuai
K5	Mengisi <i>Cari nopol</i> dengan “123”, atau tidak diisi, dan enter	Sistem menolak menampilkan data dan muncul noti “data tidak ada”	Sistem menolak menampilkan data dan muncul noti “data tidak ada”	Sesuai
K6	Klik Tombol “Keluar” saat nopol yang dimasukkan salah atau tidak diisi	Sistem menolak memproses data	Sistem menolak memproses data	Sesuai

Terdapat 2 Jumlah Form yang diuji dengan form Parkir masuk diuji sebanyak 2 kali, form parkir keluar diuji sebanyak 4 kali, sehingga total pengujian dilakukan 6 kali. Jumlah form yang ditemukan eror sebanyak 1 eror yang ada di form parkir masuk, dan 1 eror ditemukan di form parkir keluar.

4. Kesimpulan

Pengujian perangkat lunak dengan menggunakan metode *Black Box* dapat menghasilkan pengujian yang menentukan perangkat lunak yang diuji apakah dapat berjalan dengan yang sesuai diharapkan atau tidak. Berdasarkan pengujian kualitas aplikasi penghitungan parkir, dapat diambil kesimpulan bahwa pengujian dengan metode *Black Box* berbasis *Equivalence Partitions* dapat membantu proses pembuatan *case* pengujian, uji kualitas dan

menemukan kesalahan yang tidak disengaja. Dalam pengujian aplikasi penghitungan parkir ditemukan kesalahan dalam pengaturan (*setting*) waktu, sehingga hal tersebut menyebabkan proses penghitungan biaya parkir tidak berjalan dengan semestinya. Hal ini dapat segera diperbaiki agar aplikasi berjalan dengan baik.

5. Saran

Berdasarkan proses pengujian kualitas aplikasi penghitungan parkir, penulis memberikan saran pada penelitian selanjutnya untuk mempelajari dengan sungguh-sungguh metode yang akan digunakan dalam pelaksanaan penelitian, dengan cara mencari referensi dari sumber-sumber yang terkait, sehingga penelitian yang akan dilakukan dapat berjalan dengan baik dan hasilnya pun sesuai dengan yang diharapkan.

References

- Agarwal, B., Tayal, S. P., & Gupta, M. (2010). *Software Engineering & Testing*. Boston: AuthorJones & Bartlett Learning.
- Andriansyah, D. (2018). Pengujian Kotak Hitam Boundary Value Analysis Pada Sistem Informasi Manajemen Konseling Tugas Akhir. STMIK Nusa Mandiri Jakarta. *Indonesia Journal on Networking and Security*, 7(1), 13-18.
- Arwaz, A. A., Putra, K., Putra, R., Kusumawijaya, T., & Saifudin, A. (2019). Pengujian Black Box pada Aplikasi Sistem Seleksi Pemenang Tender Menggunakan Teknik Equivalence Partitions. *Jurnal Teknologi Sistem Informasi dan Aplikasi*, 2(4), 130-134. doi:10.32493/jtsi.v2i4.3708
- Febrian, V., Ramadhan, M. R., Faisal, M., & Saifudin, A. (2020). Pengujian pada Aplikasi Penggajian Pegawai dengan menggunakan Metode Blackbox. *Jurnal Informatika Universitas Pamulang*, 5(1), 61-66. doi:10.32493/informatika.v5i1.4340
- Hidayat, T., & Muttaqin, M. (2018). Pengujian Sistem Informasi Pendaftaran dan Pembayaran Wisuda Online menggunakan Black Box Testing dengan Metode Equivalence Partitioning dan Boundary Value Analysis. *Jurnal Teknik Informatika UNIS*, 6(1), 25-29.
- MZ, M. M. K. (2016). Pengujian Perangkat Lunak Metode Black Box Berbasis Equivalence Partitions Pada Aplikasi Sistem Informasi Sekolah. *Jurnal Mikrotik*, 6(3), 1-8.
- Mustaqbal, M., Firdaus, R., & Rahmadi, H. (2015). engujian Aplikasi Menggunakan Black Box Testing Boundary Value Analysis (Studi Kasus: Aplikasi Prediksi Kelulusan SNMPTN). *Jurnal Ilmiah Teknologi Informasi Terapan*, 1(3), 31-36.
- Rosalina, A., Rassi, A. A., Hadi, G. Y., Ubaidillah, R., & Desyani, T. (2020). Pengujian Black Box pada Sistem Informasi Penjualan HI Shoe Store Menggunakan Teknik Equivalence Partitions. *Jurnal Informatika Universitas Pamulang*, 5, 26-29. doi:10.32493/informatika.v5i1.3964
- Setiyani, L. (2019). Pengujian Sistem Informasi Inventory Pada Perusahaan Distributor Farmasi Menggunakan Metode Black Box Testing. *Jurnal Ilmu Komputer & Teknologi Informasi*, 4(1), 1-9.
- Shi, M. (2010). Software Functional Testing from the Perspective of Business Practice. *Computer and Information Science*, 3(4), 49 - 52.