Saturnus : Jurnal Teknologi dan Sistem Informasi
Volume. 3, Nomor. 2, April 2025
e-ISSN : 3031-9943; dan p-ISSN : 3031-9935; Hal. 10-22
DOI: https://doi.org/10.61132/saturnus.v3i1.756
Available online at: https://journal.arteii.or.id/index.php/Saturnus

Pengujian Perangkat Lunak untuk Aplikasi Sistem Pengelolaan Data Penduduk Berbasis Web di Desa Penarukan, Kecamatan Dusun Utara, Kabupaten Barito Selatan

Angelina Estevani[1], Cindy Margaretha[2*], Magdalene Isma[3]
[1,2,3] Mahasiswa Teknik Informatika, Fakultas Teknik, Universitas Palangka Raya, Kalimantan Tengah, Indonesia
angelinaestevani@gmail.com [1], *cmargaretha07@gmail.com* [2*], *daleneisma5@gmail.com* [3]

Alamat: Alamat: Jalan Yos Sudarso, Palangka Raya, KalimantanTengah
Korespondensi penulis: cmargaretha07@gmail.com

Abstract*. This study examines software testing on a Web-Based Population Data Management System in Penarukan Village. The goal is to ensure system reliability and functionality before implementation. The testing process includes Unit Testing, Integration Testing, Validation Testing, and System Testing. Unit Testing verifies individual functions, Integration Testing ensures module interaction, Validation Testing assesses compliance with user needs, and System Testing evaluates overall system performance. The results confirm that key features such as data management, birth and death registration, and population movement tracking function correctly. Comprehensive testing minimizes errors, making the system stable, secure, and user-oriented.*

Keywords: *Integration Testing, Software Testing, System Testing, Unit Testing, Validation Testing*

Abstrak. Pengujian perangkat lunak pada Sistem Pengelolaan Data Penduduk Berbasis Web di Desa Penarukan. Tujuannya adalah untuk memastikan keandalan dan fungsionalitas sistem sebelum diterapkan. Pengujian mencakup Unit Testing, Integration Testing, Validation Testing, dan System Testing. Unit Testing menguji fungsi individu, Integration Testing memastikan interaksi antar modul, Validation Testing menilai kesesuaian dengan kebutuhan pengguna, dan System Testing mengevaluasi kinerja keseluruhan sistem. Hasil menunjukkan bahwa fitur utama seperti pengelolaan data, pencatatan kelahiran dan kematian, serta perpindahan penduduk berfungsi dengan baik. Pengujian menyeluruh mengurangi risiko kesalahan, menjadikan sistem lebih stabil, aman, dan sesuai kebutuhan pengguna.

Kata kunci: Pengujian Integrasi, Pengujian Perangkat Lunak, Pengujian Sistem, Pengujian Unit, Pengujian Validasi

1. LATAR BELAKANG

Perkembangan teknologi informasi telah membawa perubahan signifikan dalam pengelolaan data kependudukan. Sistem berbasis web menjadi solusi untuk meningkatkan ketelitian dan ketepatan administrasi kependudukan, khususnya di tingkat desa (Lawencon, 2024). Salah satu tantangan utama dalam implementasi sistem tersebut adalah memastikan keandalan dan keamanan aplikasi sebelum digunakan secara luas.

Penelitian ini berfokus pada pengujian perangkat lunak untuk Sistem Pengelolaan Data Penduduk Berbasis Web di Desa Penarukan, Kecamatan Dusun Utara, Kabupaten Barito Selatan. Pengujian ini penting untuk mengidentifikasi dan mengatasi potensi kesalahan yang dapat mengganggu fungsionalitas sistem. Menurut Sommerville (2015), pengujian perangkat lunak yang sistematis dapat meningkatkan keandalan dan kualitas perangkat lunak, sehingga mengurangi risiko kegagalan saat implementasi.

Received Maret 22, 2025; Revised April 06, 2025; Accepted April 20, 2025; Published April 22, 2025

Rumusan masalah dalam penelitian ini adalah bagaimana pengujian perangkat lunak dapat memastikan bahwa sistem berjalan sesuai dengan spesifikasi yang telah ditetapkan. Untuk menjawab pertanyaan tersebut, penelitian ini menggunakan pendekatan pengujian yang mencakup Unit Testing, Integration Testing, Validation Testing, dan System Testing. Metode ini sejalan dengan pendekatan yang telah digunakan dalam berbagai studi pengujian perangkat lunak sebelumnya (Pressman, 2020).

Penelitian ini juga mengacu pada teori pengujian perangkat lunak yang menekankan pentingnya validasi dan verifikasi dalam proses pengembangan sistem (Myers, 2011). Dengan mengimplementasikan pengujian menyeluruh, sistem diharapkan dapat memberikan layanan yang lebih andal dan sesuai dengan kebutuhan pengguna.

Dengan demikian, penelitian ini bertujuan untuk mengkaji bagaimana berbagai metode pengujian perangkat lunak dapat memastikan kualitas sistem, serta mengidentifikasi faktor-faktor yang dapat mempengaruhi kinerja sistem dalam penggunaan nyata.

2. KAJIAN TEORITIS

Teori dalam penelitian ini mengacu pada berbagai literatur dan penelitian sebelumnya yang membahas tentang pengujian perangkat lunak, metode yang digunakan, serta pentingnya pengujian dalam menjamin kualitas sistem.

Konsep Pengujian Perangkat Lunak

Menurut Pressman (2010), pengujian perangkat lunak adalah proses yang bertujuan untuk mengidentifikasi kesalahan dalam perangkat lunak dan memastikan bahwa perangkat lunak berfungsi sesuai dengan kebutuhan pengguna. Pengujian ini mencakup berbagai strategi seperti Unit Testing, Integration Testing, Validation Testing, dan System Testing yang bertujuan untuk menguji setiap aspek sistem secara bertahap.

Metode Pengujian Perangkat Lunak

Menurut Kartanti (2015), metode pengujian perangkat lunak dapat dilakukan dengan pendekatan White-Box Testing dan Black-Box Testing. White-Box Testing digunakan dalam Unit Testing untuk menguji struktur internal kode, sementara Black-Box Testing lebih banyak diterapkan dalam Validation Testing untuk memastikan bahwa perangkat lunak memenuhi kebutuhan pengguna.

Keamanan dalam Pengujian Perangkat Lunak

Menurut Williams (2006), aspek keamanan dalam pengujian perangkat lunak sangat penting mengingat meningkatnya ancaman siber. Pengujian keamanan seperti penetration testing diperlukan untuk memastikan bahwa sistem tidak memiliki celah keamanan yang dapat dieksploitasi oleh pihak yang tidak bertanggung jawab.

Pengujian Performa dan Skalabilitas

Menurut Lawencon (2020), pengujian performa seperti Stress Testing dan Load Testing harus dilakukan untuk memastikan sistem dapat beroperasi dengan baik dalam kondisi beban tinggi. Tanpa pengujian ini, sistem berisiko mengalami penurunan performa atau bahkan kegagalan ketika diakses oleh banyak pengguna.

3. METODE PENELITIAN

Penelitian ini menggunakan metode eksperimen dengan pendekatan pengujian perangkat lunak. Subjek penelitian adalah Sistem Pengelolaan Data Penduduk Berbasis Web yang dikembangkan untuk Desa Penarukan. Teknik pengumpulan data dilakukan melalui pengujian langsung terhadap sistem dengan menggunakan metode Unit Testing, Integration Testing, Validation Testing, dan System Testing.

4. HASIL DAN PEMBAHASAN

Pengujian menunjukkan bahwa seluruh fitur utama dalam sistem telah diuji secara menyeluruh.

Unit Testing

Unit testing adalah salah satu jenis pengujian perangkat lunak (*software*) yang berfokus pada pengujian unit-unit terkecil dalam sebuah sistem perangkat lunak. Biasanya, *unit testing* mencakup pengujian *function*, *method*, dan *class*.Umumnya, *unit testing* adalah pengujian paling awal yang dilakukan oleh developer sebelum melakukan pengujian lain, seperti *integration test*, *functional test*, dan *end-to-end test*. Dalam pengujian unit testing ini, menggunakan metode White Box Testing dengan menguji fungsi Tambah Data. Pengujian ini bertujuan untuk memastikan bahwa setiap operasi dalam fungsi tersebut berfungsi sesuai dengan spesifikasi yang telah ditetapkan. Berikut adalah penjelasan mengenai setiap bagian kode yang diuji dalam proses White Box Testing.

a. Tambah Data Penduduk

```
application > controllers > Penduduk.php
4   class Penduduk extends CI_Controller {
24      public function tambah() {
25          $data['title'] = "Tambah Penduduk - Desa Penarukan";
26
27          $this->load->view('header', $data);
28          $this->load->view('penduduk/tambah_penduduk');
29          $this->load->view('footer');
30      }
31
32      public function proses_tambah() {
33          $nik = $this->input->post('nik');
34          $no_kk = $this->input->post('no_kk');
35          $tempat_lahir = $this->input->post('tempat_lahir');
36          $tanggal_lahir = $this->input->post('tanggal_lahir');
37          $jenis_kelamin = $this->input->post('jenis_kelamin');
38          $nama = $this->input->post('nama');
39          $alamat = $this->input->post('alamat');
40          $rt = $this->input->post('rt');
41          $rw = $this->input->post('rw');
42          $agama = $this->input->post('agama');
43          $pekerjaan = $this->input->post('pekerjaan');
44          $status_perkawinan = $this->input->post('status_perkawinan');
45
46          // Validasi panjang NIK dan No KK
47          if (strlen($nik) != 16 || strlen($no_kk) != 16) {
48              $this->session->set_flashdata('error', 'NIK dan No KK harus 16 digit.');
49              redirect(base_url('penduduk/tambah'));
50          }
51
52          // Cek apakah NIK sudah ada di database
53          if ($this->m_penduduk->cek_nik($nik)) {
54              $this->session->set_flashdata('error', 'NIK ' . $nik . ' sudah terdaftar.');
55              redirect(base_url('penduduk/tambah'));
56          }
57
58          $data = array(
59              'nik' => $nik,
60              'no_kk' => $no_kk,
61              'nama' => ucwords($nama),
62              'tempat_lahir' => ucwords($tempat_lahir),
63              'tanggal_lahir' => $tanggal_lahir,
64              'jenis_kelamin' => $jenis_kelamin,
65              'alamat' => ucwords($alamat),
66              'rt' => $rt,
67              'rw' => $rw,
```

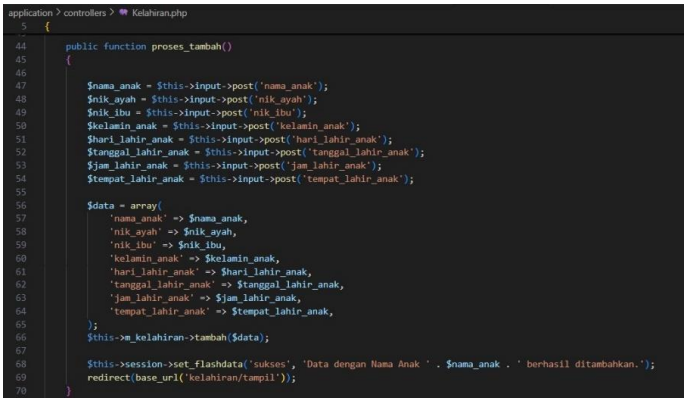
Gambar 1 Tambah data Penduduk

Pada gambar 1 diatas bagian ini menunjukan Controller dalam framework CodeIgniter yang berfungsi untuk menangani proses penambahan data penduduk di Desa Penarukan. Fungsi tambah() digunakan untuk menampilkan halaman formulir penambahan penduduk dengan memuat tampilan header, formulir input, dan footer. Sementara itu, fungsi proses_tambah() bertanggung jawab untuk mengambil data dari formulir yang dikirim oleh pengguna, kemudian melakukan validasi terhadap NIK dan No KK agar memiliki panjang 16 digit. Jika validasi gagal, sistem akan menampilkan pesan error menggunakan flashdata dan mengarahkan kembali ke halaman formulir. Selain itu, sistem juga memeriksa apakah NIK yang dimasukkan sudah terdaftar dalam database dengan memanggil metode cek_nik() dari model m_penduduk. Jika NIK sudah ada, pengguna akan mendapatkan notifikasi bahwa data tidak bisa didaftarkan ulang. Jika semua validasi berhasil, data penduduk akan disimpan ke dalam database melalui metode tambah() pada model, lalu sistem akan memberikan notifikasi sukses dan mengarahkan pengguna ke halaman daftar penduduk.

Untuk menguji fungsionalitas kode ini menggunakan PHPUnit, dapat dilakukan beberapa pengujian, seperti validasi panjang NIK dan No KK, pengecekan apakah NIK sudah terdaftar, serta memastikan bahwa data penduduk dapat ditambahkan dengan

benar. Pengujian dilakukan dengan membuat unit test yang memanfaatkan teknik mocking untuk menghindari manipulasi langsung terhadap database. Dalam pengujian validasi NIK, misalnya, dilakukan pengecekan apakah NIK yang kurang dari 16 digit akan menghasilkan error. Selain itu, pengujian terhadap metode cek_nik() dilakukan dengan mensimulasikan skenario di mana NIK sudah ada dalam database, yang seharusnya menghasilkan pesan error dan mencegah data disimpan. Terakhir, pengujian juga dilakukan untuk memastikan bahwa data penduduk yang valid dapat tersimpan dengan benar menggunakan metode tambah(). Dengan adanya pengujian ini, diharapkan sistem dapat berjalan dengan lebih andal dan mengurangi kemungkinan terjadinya kesalahan saat pengguna memasukkan data baru.

b. Tambah Data kelahiran



```
application > controllers > Kelahiran.php
5   {
44      public function proses_tambah()
45      {
46
47          $nama_anak = $this->input->post('nama_anak');
48          $nik_ayah = $this->input->post('nik_ayah');
49          $nik_ibu = $this->input->post('nik_ibu');
50          $kelamin_anak = $this->input->post('kelamin_anak');
51          $hari_lahir_anak = $this->input->post('hari_lahir_anak');
52          $tanggal_lahir_anak = $this->input->post('tanggal_lahir_anak');
53          $jam_lahir_anak = $this->input->post('jam_lahir_anak');
54          $tempat_lahir_anak = $this->input->post('tempat_lahir_anak');
55
56          $data = array(
57              'nama_anak' => $nama_anak,
58              'nik_ayah' => $nik_ayah,
59              'nik_ibu' => $nik_ibu,
60              'kelamin_anak' => $kelamin_anak,
61              'hari_lahir_anak' => $hari_lahir_anak,
62              'tanggal_lahir_anak' => $tanggal_lahir_anak,
63              'jam_lahir_anak' => $jam_lahir_anak,
64              'tempat_lahir_anak' => $tempat_lahir_anak,
65          );
66          $this->m_kelahiran->tambah($data);
67
68          $this->session->set_flashdata('sukses', 'Data dengan Nama Anak ' . $nama_anak . ' berhasil ditambahkan.');
69          redirect(base_url('kelahiran/tampil'));
70      }
```

Gambar 2 Tambah Data Kelahiran

Pada gambar 2 diatas bagian ini merupakan bagian dari Controller dalam framework CodeIgniter yang mengelola data kelahiran di Desa Penarukan. Terdapat dua fungsi utama, yaitu proses_tambah() dan edit($id). Fungsi proses_tambah() digunakan untuk menangani penambahan data kelahiran yang dikirim melalui form input. Fungsi ini mengambil data dari request POST, seperti nama anak, NIK ayah dan ibu, jenis kelamin anak, hari, tanggal, serta jam lahir anak, dan tempat lahir anak. Setelah data dikumpulkan dalam array $data, fungsi ini memanggil metode tambah($data) pada model m_kelahiran untuk menyimpan data ke dalam database. Setelah proses penyimpanan berhasil, sistem memberikan notifikasi sukses menggunakan set_flashdata() dan mengarahkan pengguna ke halaman daftar kelahiran (kelahiran/tampil).

Sementara itu, fungsi edit($id) digunakan untuk menampilkan halaman edit data kelahiran berdasarkan ID yang diberikan. Fungsi ini mengambil informasi kelahiran yang akan diedit melalui metode edit($id) dari model m_kelahiran. Selain itu, data

penduduk juga diambil melalui metode get_penduduk() untuk keperluan referensi saat melakukan pengeditan. Data yang diperoleh kemudian dikirim ke tampilan (view) dengan memuat file header, form edit kelahiran, dan footer. Untuk menguji fungsi ini menggunakan PHPUnit, pengujian dapat dilakukan dengan memastikan bahwa: Data yang dikirim melalui proses_tambah() benar-benar tersimpan di database. Fungsi edit($id) mengembalikan data kelahiran yang benar berdasarkan ID yang diberikan. Flash message muncul setelah berhasil menambahkan data. Redirect ke halaman yang benar setelah operasi selesai.

Pengujian dapat dilakukan dengan mocking model m_kelahiran agar tidak perlu berinteraksi langsung dengan database selama pengujian. Dengan demikian, kode ini dapat diuji secara lebih efisien untuk memastikan bahwa fungsinya berjalan dengan benar.

c. Tambah Data Kematian

```
application > controllers > Kematian.php
5   {
26      public function tambah()
27      {
28          $data['title'] = "Tambah Kematian - Desa Penarukan";
29          $data['penduduk'] = $this->m_kematian->get_penduduk();
30
31          $this->load->view('header', $data);
32          $this->load->view('kematian/tambah_kematian', $data);
33          $this->load->view('footer');
34      }
35
36      public function proses_tambah()
37      {
38
39          $nik = $this->input->post('nik');
40          $hari_wafat = $this->input->post('hari_wafat');
41          $tanggal_wafat = $this->input->post('tanggal_wafat');
42          $pukul = $this->input->post('pukul');
43          $sebab_wafat = $this->input->post('sebab_wafat');
44          $tempat = $this->input->post('tempat');
45          $keterangan = $this->input->post('keterangan');
46
47          $data = array(
48              'nik' => $nik,
49              'hari_wafat' => $hari_wafat,
50              'tanggal_wafat' => $tanggal_wafat,
51              'pukul' => $pukul,
52              'sebab_wafat' => $sebab_wafat,
53              'tempat' => $tempat,
54              'keterangan' => $keterangan,
55          );
56          $this->m_kematian->tambah($data);
57
58          $this->session->set_flashdata('sukses', 'Data dengan NIK ' . $nik . ' berhasil ditambahkan.');
59          redirect(base_url('kematian/tampil'));
60      }
```

Gambar 3 Tambah Data Kematian

Pada gambar 3 diatas bagian ini merupakan merupakan bagian dari Controller dalam framework CodeIgniter yang menangani pengelolaan data kematian di Desa Penarukan. Fungsi tambah() digunakan untuk menampilkan halaman formulir penambahan data kematian dengan mengambil daftar penduduk dari model m_kematian melalui metode get_penduduk(), yang kemungkinan digunakan untuk memilih individu yang akan dicatat kematiannya. Data ini kemudian diteruskan ke tampilan yang terdiri dari header, form tambah kematian, dan footer.

Sementara itu, fungsi proses_tambah() bertanggung jawab untuk menangani data yang dikirim melalui formulir input, termasuk NIK, hari dan tanggal wafat, waktu kematian, penyebab, tempat wafat, serta keterangan tambahan. Setelah data

dikumpulkan dalam bentuk array, fungsi ini memanggil metode tambah($data) dari model m_kematian untuk menyimpan informasi ke dalam database. Setelah proses penyimpanan berhasil, sistem memberikan notifikasi sukses menggunakan flashdata dan mengarahkan pengguna ke halaman daftar kematian (kematian/tampil). Untuk memastikan fungsionalitasnya, pengujian dengan PHPUnit dapat dilakukan dengan memeriksa apakah data berhasil disimpan, memastikan formulir ditampilkan dengan benar, serta memverifikasi bahwa sistem memberikan notifikasi dan melakukan redirect ke halaman yang sesuai setelah proses selesai. Pengujian ini dapat dilakukan dengan teknik mocking model agar tidak perlu berinteraksi langsung dengan database selama pengujian.

d. Tambah Data Pindah

```
application > controllers > Pindah.php
5   {
25      public function tambah()
26      {
27          $data['title'] = "Tambah Pindah  - Desa Penarukan";
28          $data['penduduk'] = $this->m_penduduk->tampil();
29
30          $this->load->view('header', $data);
31          $this->load->view('pindah/tambah_pindah');
32          $this->load->view('footer');
33      }
34
35      public function proses_tambah()
36      {
37          $nik = $this->input->post('nik');
38          $alasan_pindah = $this->input->post('alasan_pindah');
39          $tanggal_pindah = $this->input->post('tanggal_pindah');
40          $alamat_tujuan = $this->input->post('alamat_tujuan');
41          $klasifikasi_pindah = $this->input->post('klasifikasi_pindah');
42
43          $data = array(
44              'nik' => $nik,
45              'alasan_pindah' => ucwords($alasan_pindah),
46              'tanggal_pindah' => $tanggal_pindah,
47              'alamat_tujuan' => $alamat_tujuan,
48              'klasifikasi_pindah' => $klasifikasi_pindah,
49          );
50          $this->m_pindah->tambah($data);
51
52          $this->session->set_flashdata('sukses', 'Data dengan NO KK ' . $nik . ' berhasil ditambahkan.');
53          redirect(base_url('pindah/tampil'));
54      }
```

Gambar 4 Tambah Data Pindah

Pada gambar 4 diatas bagian ini merupakan bagian dari Controller dalam framework CodeIgniter yang menangani pengelolaan data kepindahan penduduk di Desa Penarukan. Fungsi tambah() digunakan untuk menampilkan halaman formulir penambahan data pindah dengan mengambil daftar penduduk yang tersedia dari model m_penduduk melalui metode tampil(). Data ini kemudian dikirim ke tampilan yang memuat header, form tambah pindah, dan footer. Selanjutnya, fungsi proses_tambah() menangani data yang dikirim melalui formulir input, seperti NIK, alasan pindah, tanggal pindah, alamat tujuan, dan klasifikasi pindah. Setelah semua data dikumpulkan dalam bentuk array, sistem menyimpan informasi tersebut ke dalam database menggunakan metode tambah($data) dari model m_pindah.

Setelah proses penyimpanan selesai, sistem akan menampilkan notifikasi sukses menggunakan flashdata dan mengarahkan pengguna ke halaman daftar kepindahan (pindah/tampil). Untuk memastikan fungsionalitas kode ini, pengujian dengan PHPUnit

dapat dilakukan dengan memverifikasi bahwa data yang dimasukkan benar-benar tersimpan, memastikan bahwa formulir ditampilkan dengan data yang sesuai, serta memastikan bahwa setelah proses penyimpanan, pengguna mendapatkan notifikasi dan diarahkan ke halaman yang benar. Pengujian dapat menggunakan mocking model m_pindah untuk menghindari interaksi langsung dengan database selama pengujian.

e. Tambah Data Kedatangan

```
application > controllers > Kedatangan.php
5   {
24      public function tambah()
25      {
26          $data['title'] = "Tambah Kedatangan  - Desa Penarukan";
27          $data['penduduk'] = $this->m_kedatangan->get_penduduk();
28
29          $this->load->view('header', $data);
30          $this->load->view('kedatangan/tambah_kedatangan');
31          $this->load->view('footer');
32      }
33
34      public function proses_tambah()
35      {
36          $nik = $this->input->post('nik');
37          $alasan_datang = $this->input->post('alasan_datang');
38          $tanggal_datang = $this->input->post('tanggal_datang');
39          $alamat_tujuan = $this->input->post('alamat_tujuan');
40          $klasifikasi_datang = $this->input->post('klasifikasi_datang');
41
42          $data = array(
43              'nik' => $nik,
44              'alasan_datang' => ucwords($alasan_datang),
45              'tanggal_datang' => $tanggal_datang,
46              'alamat_tujuan' => $alamat_tujuan,
47              'klasifikasi_datang' => $klasifikasi_datang,
48          );
49          $this->m_kedatangan->tambah($data);
50
51          $this->session->set_flashdata('sukses', 'Data dengan NO NIK ' . $nik . ' berhasil ditambahkan.');
52          redirect(base_url('kedatangan/tampil'));
53      }
```

Gambar 5 Tambah Data Pindah

Pada gambar 4 diatas bagian ini merupakan bagian dari Controller dalam framework CodeIgniter yang digunakan untuk mengelola data kedatangan penduduk di Desa Penarukan. Fungsi tambah() bertugas untuk menampilkan halaman formulir penambahan data kedatangan dengan mengambil daftar penduduk yang tersedia dari model m_kedatangan melalui metode get_penduduk(). Data ini kemudian diteruskan ke tampilan yang terdiri dari header, form tambah kedatangan, dan footer. Sementara itu, fungsi proses_tambah() berperan dalam menangani data yang dikirim melalui formulir input, termasuk NIK, alasan kedatangan, tanggal kedatangan, alamat tujuan, dan klasifikasi kedatangan.

Setelah data dikumpulkan dalam bentuk array, sistem akan menyimpannya ke dalam database menggunakan metode tambah($data) dari model m_kedatangan. Setelah penyimpanan berhasil, sistem menampilkan pesan sukses menggunakan flashdata dan mengarahkan pengguna ke halaman daftar kedatangan (kedatangan/tampil). Untuk memastikan fungsi ini berjalan dengan baik, pengujian menggunakan PHPUnit dapat dilakukan dengan memverifikasi bahwa data berhasil disimpan, formulir ditampilkan dengan benar, serta memastikan bahwa setelah proses penyimpanan, pengguna mendapatkan notifikasi sukses dan diarahkan ke halaman yang sesuai. Pengujian ini

dapat dilakukan dengan teknik mocking model m_kedatangan agar pengujian dapat berjalan tanpa harus berinteraksi langsung dengan database.

Integration Testing

Integration testing adalah proses pengujian antara dua aplikasi software, yang berfokus pada keakuratan, kecakapan, hingga kualitas antarmuka keduanya. Metode pengujian satu ini berguna untuk mengidentifikasi kesalahan interaksi pada dua unit software yang terintegrasi atau saling berhubungan. Tujuan integration testing secara keseluruhan adalah untuk mengidentifikasi masalah atau bug yang muncul saat berbagai komponen digabungkan dan berinteraksi satu sama lain. Selain itu dengan pengujian ini juga developer dapat mengetahui waktu interaksi yang dibutuhkan pada dua unit terintegrasi. Integration testing biasa dilakukan setelah unit testing dan sebelum sistem testing. Melalui pengujian ini juga, Softwer testing dapat mengidentifikasi dan menyelesaikan masalah integrasi di awal ssoklus pengembagan, yang pada akhirnya penerapan hal ini mampu mengurangi resiko yang lebih parah di masa depan.

Tabel 1

No	Skenario Uji	Data Input	Ekspetasi output	Status
1	Menambah data penduduk baru dan memastikan data tersimpan di database	NIK: No. KK: Nama: TTL: JK: Alamat: RT: RW: Agama: Pekerjaan: Status: Perkawinan:	Data tersimpan di database, notifikasi “Data Berhasil Ditambahkan”	Berhasil
2	Mencegah NIK input yang sudah ada	NIK: (Sudah Ada)	Notifikasi “NIK sudah terdaftar data tidak di simpan”	Berhasil
3	Mencegah NIK kurang dari 15 digit	NIk:	Notofikasi “NIK Harus 16 Digit”, data tidak terimpan	Berhasil
4	Mengedit data penduduk	NIK: Ubah alamat ke “”	Data di perbaharui di database notifikasi “data berhasil di perbaharui”	Berhasil
5	Menghapus data	NIK:	Data terhapus dari	Berhasil

	penduduk		database notifikasi "Data berhasil di hapus"	
6	Menampilkan daftar penduduk dari database	-	Semua data Penduduk muncul dengan benar ditampilan	Berhasil
7	Mencari penduduk berdasarkan NIK	NIK	Data penduduk sesuai muncul di hasil pencarian	Berhasil

Validation Testing

Validation testing adalah sebuah proses pengujian yang dilakukan untuk memastikan suatu produk, sistem, ataupun software dapat sesuai dengan kebutuhan serta harapan pengguna atau klien. Tujuan utama uji coba ini adalah untuk memastikan produk akhir berfungsi sesuai dengan apa yang diharapkan dan direncanakan sebelumnya. Validation testing biasanya dilakukan pada masa akhir pengembanga software, tepatnya sebelum produk tesebut diluncurkan secara massal ke pengguna. Selama proses penerapannya ini melibatkan banyak pengujian fitur mulai dari user interface, performa, dan lain sebagainya. Validation Testing bertujuan untuk memastikan bahwa fitur berfungsi sesuai dengan kebutuhan pengguna. Pengujian ini difokuskan pada validasi input serta respons sistem terhadap berbagai kondisi yang mungkin terjadi. pengujian dirangkum dalam tabel berikut:

Tabel 2

No	Skenario Uji	Data Input	Ekspetasi output	Status
1	Input data kelahiran dengan semua field valid	Nama anak: NIK Ayah: NIK Ibu: JK: TL:	Data berhasil di simpan	Berhasil
2	Input data dengan NIK Ayah kurang dari 16 digit	Nama anak: NIK Ayah: NIK Ibu:	Error : "NIK harus 16 digit"	Gagal
3	Input data kelahiran dengan format tgl salah	Nama anak: NIK Ayah: NIK Ibu: TL:	Error : "Format tgl tidak valid"	Gagal
4	Input data dengan karakter anggka di nama anak	Nama anak: NIK Ayah: NIK Ibu:	Error : "Nama hanya boleh berisi huruf"	Gagal
5	Input data dengan NIK ibu	Nama anak: NIK Ayah:	Error : "NIK ibu tidak di temukan"	Gagal

	tidak terdaftar di database	NIK Ibu:		
6	Input data tanpa mengisi nama anak	Nama Anak:	Error : "Nama anak wajib diis"	Gagal
7	Input data kematian dengan semua field valid	NIK: Hari Wafat: Tgl Wafat: Sebab Wafat:	Data berhasil di simpan	Berhasil
8	Input data kematian dengan NIK kurang dari 16 digit	NIK: Hari Wafat: Tgl Wafat:	Error : "NIK harus 16 digit"	Gagal
9	Input data kematian dengan tanggal wafat lebih awal dari tanggal lahir	NIK: TL: Tgl Wafat:	Error : "tanggal wafat tidak boleh lebih awal dari tanggal lahir"	Gagal
10	Input data kematian dengan penyebab wafat mengandung karakter angka	Sebab Wafat:	Error : "Sebab wafat hanya boleh berisi huruf"	Gagal
11	Input data kematian dengan NIK tidak terdaftar	NIK:	Error : "NIK tidak di temukan dalam database"	Gagal
12	Input data kematian dengan tempat format wafat kosong	Tempat:	Error : "Tempat wafat wajib diisi"	Gagal

System Testing

System testing adalah tahap pengujian dalam siklus pengembangan perangkat lunak yang bertujuan untuk menguji fungsi lengkap dari suatu sistem secara keseluruhan. Pengujian ini biasanya dilakukan dengan mensimulasikan kondisi nyata yang akan dialami perangkat lunak saat digunakan oleh pengguna. Fokus utamanya yakni pada pengujian *end-to-end* yang mencakup pengujian fungsionalitas, performa, keamanan, dan beberapa aspek lain dari perangkat lunak.

Tabel 3

No	Skenario Uji	Data Input	Ekspetasi output	Status
1	Login dengan	Username:	Redirect ke	Berhasil

	kredensial valid	Password:	halaman beranda	
2	Login dengan username salah	Username: Password:	Error : “Username atau Password salah”	Gagal
3	Login dengan Password salah	Username: Password:	Error : “Username atau Password salah”	Gagal
4	Login dengan Username dan password kosong	Username: Password:	Error : “Field tidak boleh kosong ”	Gagal
5	Login dengan akun yang tidak terdaftar	Username: Password:	Error : “Akun tidak di temukan”	Gagal
6	Akses beranda setelah login	-	Beranda di tampilkan dengan menu utama	Berhasil
7	Akses beranda tanpa login (Langsung ke URL)	-	Redirect ke halaman login	Gagal
8	Logout dari sistem	Klik tombol Logout	Redirect ke halaman login	Berhasil

Unit Testing memastikan bahwa setiap fungsi dasar bekerja dengan baik tanpa kesalahan. Integration Testing menunjukkan bahwa interaksi antar modul berjalan dengan lancar. Validation Testing membuktikan bahwa sistem memenuhi persyaratan yang telah ditentukan, sementara System Testing mengonfirmasi bahwa sistem berjalan stabil dalam lingkungan nyata.

Berdasarkan hasil pengujian, dapat disimpulkan bahwa metode pengujian yang diterapkan telah berhasil mengidentifikasi dan mengatasi potensi kesalahan dalam sistem. Pengujian menunjukkan bahwa sistem dapat mengelola data kependudukan dengan baik, termasuk pencatatan kelahiran, kematian, dan perpindahan penduduk. Tidak ditemukan kesalahan kritis dalam pengujian yang dilakukan, sehingga sistem dinilai layak untuk digunakan secara luas. Dengan pengujian yang menyeluruh, sistem ini dapat memberikan manfaat yang optimal dalam mendukung administrasi kependudukan berbasis digital.

DAFTAR REFERENSI

Dicoding Academy. (2023). *Penerapan pengujian menyeluruh dalam sistem berbasis web*. Dicoding Academy.

Kartanti, R. (2015). *Strategi pengujian perangkat lunak dan implementasinya*. Jakarta: Informatika Press.

Lawencon. (2020). *Mengenal integration testing dalam pengujian perangkat lunak*. Retrieved from https://www.lawencon.com *(catatan: tambahkan URL lengkap jika tersedia)*

Lawencon. (2024). *Strategi pengujian perangkat lunak untuk sistem berbasis web*. Lawencon Digital.

Pressman, R. S. (2010). *Software engineering: A practitioner's approach* (7th ed.). New York: McGraw-Hill.

Sommerville, I. (2015). *Software engineering* (10th ed.). Pearson.

Williams, L. (2006). Security testing for software systems: A comprehensive approach. *ACM*. *(catatan: tambahkan volume, issue, dan halaman jika tersedia)*