

LAPORAN PRAKTIK KERJA LAPANGAN (PKL)

INSTANSI/PERUSAHAAN

DINAS KOMUNIKASI DAN INFORMATIKA KABUPATEN GRESIK

**PENERAPAN FRAMEWORK CODEIGNITER DALAM PERANCANGAN SISTEM
INFORMASI PENCATATAN PASIEN COVID-19 (SIPCOP) PADA RUMAH SAKIT DARURAT
GELORA JOKO SAMUDRO GRESIK BERBASIS WEB**

Diajukan untuk memenuhi sebagian persyaratan Kurikulum Sarjana



Disusun Oleh :

Chairunisa Dwinanda Asti	175150701111015
Velia Wilda Ifanah	175150707111007
Ya'Qub Al - Kindi	175150707111032

**PROGRAM STUDI TEKNOLOGI INFORMASI
JURUSAN SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2020**

PERSETUJUAN

LAPORAN PRAKTIK KERJA LAPANGAN (PKL)
INSTANSI/PERUSAHAAN
DINAS KOMUNIKASI DAN INFORMATIKA KABUPATEN GRESIK

PENERAPAN FRAMEWORK CODEIGNITER DALAM PERANCANGAN SISTEM
INFORMASI PENCATATAN PASIEN COVID-19 PADA RUMAH SAKIT DARURAT
GELORA JOKO SAMUDRO GRESIK BERBASIS WEB

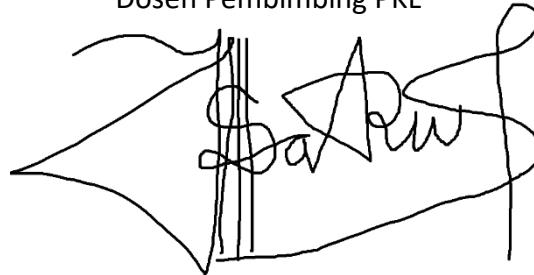
Diajukan untuk memenuhi sebagian persyaratan Kurikulum Sarjana
Program Studi Teknologi Informasi

Disusun Oleh :

Chairunisa Dwinanda Asti	175150701111015
Velia Wilda Ifanah	175150707111007
Ya'Qub Al - Kindi	175150707111032

Praktik Kerja Lapangan ini dilaksanakan pada 22 Juni 2020 sampai dengan 22
Agustus 2020 telah diperiksa dan disetujui oleh :

Dosen Pembimbing PKL



Issa Arwani, S.Kom., M.Sc.
NIP. 19830922 201212 1 003

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam laporan PKL ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain dalam kegiatan akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam laporan PKL ini terbukti terdapat unsur-unsur plagiasi, saya bersedia PKL ini digugurkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 19 Oktober 2020

A handwritten signature in black ink, appearing to read 'Velia' followed by a stylized surname.

Velia Wilda Ifanah
NIM. 175150707111007

KATA PENGANTAR

Puji syukur kepada Allah SWT yang telah melimpahkan rahmat dan karunia-Nya kepada penulis, sehingga penulis dapat menyelesaikan Praktik Kerja Lapangan dan penulisan laporan hasil Praktik Kerja Lapangan yang berjudul “Penerapan *Framework* Codeigniter dalam Perancangan Sistem Informasi Pencatatan Pasien *COVID-19* pada Rumah Sakit Darurat Gelora Joko Samudro Gresik Berbasis Web”.

Penulis menyadari bahwa penulisan ini tidak bisa terselesaikan tanpa pihak-pihak yang mendukung baik secara moril dan materil. Maka, penulis menyampaikan banyak terimakasih kepada pihak-pihak yang telah membantu penulis dalam penulisan Laporan Praktik Kerja Lapangan terutama kepada :

1. Orang tua yang selalu memberikan dukungan, bimbingan, nasihat serta doa selama penulis melakukan Praktik Kerja Lapangan.
2. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya yang telah memberikan kesempatan untuk melaksanakan Praktik Kerja Lapangan.
3. Bapak Issa Arwani, S.Kom., M.Sc selaku Ketua Jurusan Sistem Informasi Fakultas Ilmu Komputer Universitas Brawijaya sekaligus selaku dosen pembimbing PKL yang senantiasa meluangkan waktu dan memberikan bimbingan dalam melakukan Praktik Kerja Lapangan
4. Bapak Widhy Hayuhardika Nugraha Putra, S.Kom., M.Kom. selaku Kepala Program Studi Teknologi Informasi Fakultas Ilmu Komputer Universitas Brawijaya.
5. Ibu Anik Nur Kholifah dan Bapak Jeffry Nasri Faruki selaku pembimbing dari Seksi Aplikasi dan Pengembangan Informatika selama Praktik Kerja Lapangan berlangsung.
6. Segenap karyawan Dinas Komunikasi dan Informatika Kabupaten Gresik, Radio Announcer Suara Gresik, beserta semua pihak yang telah ikut membantu dalam pelaksanaan kegiatan Praktik Kerja Lapangan yang tidak dapat disebutkan satu persatu.

Penulis menyadari bahwa laporan ini masih jauh dari kata sempurna, sehingga penulis mengharapkan kritik dan saran bagi penulis. Akhir kata, semoga laporan ini dapat memberikan manfaat bagi penulis dan orang lain.

Malang, 19 Oktober 2020

Penulis

Chairunisa Dwinanda Asti

Velia Wilda Ifanah

Ya'Qub Al – Kindi

ABSTRAK

Pada akhir tahun 2019, di Kota Wuhan, provinsi Hubei, China ditemukan Virus bernama COVID-19, yaitu penyakit menular yang disebabkan oleh virus corona. Virus COVID-19 dapat menyebar dengan mudah. Hingga pada Maret 2020, Presiden Jokowi mengonfirmasi adanya kasus positif terjangkit virus corona (COVID-19) di Indonesia. Dan pada bulan Juni Jawa Timur menduduki peringkat tertinggi sehingga 18 rumah sakit rujukan pasien COVID-19 di Kabupaten Gresik sudah tidak mampu menampung pasien positif. Gubernur Jawa Timur meminta Pemerintah Kabupaten Gresik untuk menjadikan Stadion Gelora Joko Samudro (GJS) menjadi Rumah Sakit darurat. Sehingga untuk membantu Pemerintah Kabupaten Gresik dalam memberikan fasilitas dan memudahkan pencatatan maka kita membuat sistem informasi pencatatan COVID-19 (SIPCOP) untuk memudahkan pencatatan masuk dan keluarnya pasien pada rumah sakit darurat Gelora Joko Samudra (GJS).

Sistem Informasi Pencatatan COVID-19 (SIPCOP) dikembangkan menggunakan metode *Waterfall* dengan menggunakan bahasa pemrograman PHP dan basis data MySQL. Implementasi *website* menggunakan *framework* Codeigniter 3.0, sedangkan untuk implementasi tampilan menggunakan *framework* Bootstrap 4.0. Sistem Informasi ini juga terintegrasi dengan API (Application Programming Interface) dari Dinas Kependudukan dan Pencatatan Sipil Kabupaten Gresik, dimana sistem dapat menampilkan data lengkap melalui NIK pasien. Setelah itu dilakukan pengujian terhadap sistem menggunakan *Black Box testing* untuk mengetahui kebutuhan sistem. Hasil dari pengujian *Black Box Testing* didapatkan bahwa setiap fungsionalitas sistem telah terpenuhi, yakni proses Login admin dan petugas, input petugas, mengelola ruang rawat, mengelola status, input pasien, melihat grafik, melihat tracking riwayat pasien. Sedangkan untuk hasil pengujian Compatibility Testing dibantu dengan tools SortSite dan didapatkan bahwa sistem telah memenuhi kriteria compatibility.

Kata Kunci : Sistem Informasi Pencatatan *Covid*, *waterfall method*, PHP, MySQL, *Framework* Codeigniter 3.0, *Black Box testing*

ABSTRACT

At the end of 2019, in Wuhan City, Hubei Province, China a virus named COVID-19 was found, which is an infectious disease caused by the corona virus. The COVID-19 virus can spread easily. Until March 2020, President Jokowi confirmed a positive case of contracting the corona virus (COVID-19) in Indonesia. And in June East Java was in the highest ranking so that 18 referral hospitals for COVID-19 patients in Gresik Regency were unable to accommodate positive patients. The Governor of East Java asked the Gresik Regency Government to turn the Gelora Joko Samudro Stadium (GJS) into an emergency hospital. So that in order to help the Gresik Regency Government in providing facilities and facilitating recording, we have created a COVID-19 recording information system (SIPCOP) to facilitate recording the entry and exit of patients at the Gelora Joko Samudra (GJS) emergency hospital.

The Covid Recording Information System (SIPCOP) was developed using the Waterfall method using the PHP programming language and MySQL database. The website implementation uses the Codeigniter 3.0 framework, while the display implementation uses the Bootstrap 4.0 framework. This information system is also integrated with the API (Application Programming Interface) of the Gresik Regency population and civil registration office, where the system can display complete data through the patient's NIK. After that, the system is tested using black box testing to determine system requirements. The results of the Black Box Testing show that every system functionality has been fulfilled, namely the admin and staff login process, input of officers, managing the ward, managing status, patient input, viewing charts, viewing patient history tracking. Meanwhile, the results of the Compatibility Testing were assisted by the SortSite tool and it was found that the system met the compatibility criteria.

Keywords : Covid Recording Information System, *waterfall method* , PHP, MySQL, *Framework* CodeIgniter 3.0, *Black Box testing*

DAFTAR ISI

PERSETUJUAN.....	ii
PERNYATAAN ORISINALITAS	iii
KATA PENGANTAR	iv
ABSTRAK.....	v
ABSTRACT	vi
DAFTAR ISI	vii
DAFTAR GAMBAR	xii
DAFTAR TABEL	xiv
DAFTAR LAMPIRAN.....	xvi
BAB 1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	1
1.3 Tujuan	3
1.4 Batasan Masalah	3
1.5 Manfaat.....	3
1.6 Waktu dan Tempat Pelaksanaan.....	3
1.7 Sistematika Penulisan	4
BAB 2 PROFIL OBJEK PKL.....	6
2.1 Profil Dinas Komunikasi dan Informatika Kabupaten Gresik	6
2.2 Visi dan Misi Dinas Komunikasi dan Informatika Kabupaten Gresik.....	6
2.3 Struktur Organisasi Dinas Komunikasi dan Informatika Gresik	6
2.3.1 Seksi Aplikasi dan Pengembangan Informatika.....	7
BAB 3 TINJAUAN PUSTAKA	9
3.1 Sistem Informasi.....	9
3.1.1 Pengembangan Sistem Informasi	9
3.1.2 Rekayasa Kebutuhan.....	10
3.1.3 <i>Unified Modelling Language</i>	11
3.2 Teknologi Pengembangan Sistem	16
3.2.1 Metode Pengembangan Waterfall	16
3.2.2 Aplikasi Berbasis Web	18

3.2.3	Bahasa Pemrograman	18
3.2.4	<i>Software</i> Pendukung.....	19
3.2.5	<i>Framework</i>	19
3.3	Pengujian Perangkat Lunak.....	21
3.3.1	Black Box Testing	21
3.3.2	Compatibility Testing	22
BAB 4 METODOLOGI.....		23
4.1	Diskusi dengan Pembimbing Lapangan	24
4.2	Studi Kepustakaan	24
4.3	Analisis Kebutuhan	24
4.4	Perancangan Sistem	24
4.5	Implementasi	25
4.6	Pengujian Sistem	25
4.7	Kesimpulan dan Saran	25
BAB 5 ANALISIS KEBUTUHAN		26
5.1	Deskripsi Umum Sistem	26
5.2	Identifikasi Aktor	26
5.3	Analisis Kebutuhan Fungsional	27
5.4	Analisis Kebutuhan Non Fungsional	29
5.5	<i>Use Case Diagram</i>	29
5.6	Skenario Use Case	31
5.7	<i>Activity Diagram</i>	43
5.7.1	<i>Activity Diagram Login Admin</i>	43
5.7.2	<i>Activity Diagram Login Petugas</i>	44
5.7.3	<i>Activity Diagram Input Data Petugas</i>	45
5.7.4	<i>Activity Diagram Edit Data Petugas</i>	46
5.7.5	<i>Activity Diagram Hapus data Petugas</i>	47
5.7.6	<i>Activity Diagram Mengelola daftar status</i>	47
5.7.7	<i>Activity Diagram Mengelola Daftar Ruang Rawat</i>	49
5.7.8	<i>Activity Diagram Input data pasien</i>	50
5.7.9	<i>Activity Diagram Edit data pasien</i>	52
5.7.10	<i>Activity Diagram Hapus data pasien</i>	52

5.7.11	<i>Activity Diagram</i> Melihat grafik harian dan bulanan	53
5.7.12	<i>Activity Diagram</i> Tracking Riwayat pasien	54
BAB 6 PERANCANGAN SISTEM.....		56
6.1	Arsitektur Sistem	56
6.2	Pemodelan <i>Sequence Diagram</i>	57
6.2.1	<i>Sequence Diagram</i> Login Admin.....	57
6.2.2	<i>Sequence Diagram</i> Login petugas.....	57
6.2.3	<i>Sequence Diagram</i> Input data petugas.....	58
6.2.4	<i>Sequence Diagram</i> Input data pasien	59
6.2.5	<i>Sequence Diagram</i> Input ruang rawat	59
6.2.6	<i>Sequence Diagram</i> Input status pasien.....	60
6.2.7	<i>Sequence Diagram</i> Tracking riwayat pasien	61
6.3	Pemodelan <i>Class Diagram</i>	62
6.4	Pemodelan PDM (Physical Data Model).....	63
6.4.1	Perancangan Tabel Admin.....	63
6.4.2	Perancangan Tabel Pasien.....	63
6.4.3	Perancangan Tabel Petugas	64
6.4.4	Perancangan Tabel Ruang rawat	65
6.4.5	Perancangan Tabel Status pasien	65
6.4.6	Perancangan Tabel Log riwayat pasien.....	65
6.4.7	Perancangan Tabel Tracking status	65
6.5	Perancangan <i>User Interface</i>	66
6.5.1	User Interface Login Admin dan Login Petugas.....	66
6.5.2	<i>User Interface</i> Halaman Dashboard Admin (Input Petugas).....	67
6.5.3	<i>User Interface</i> halaman Ruang Rawat.....	68
6.5.4	<i>User Interface</i> halaman Status	69
6.5.5	<i>User Interface</i> halaman Dashboard petugas (Grafik)	70
6.5.6	<i>User Interface</i> halaman Input pasien.....	70
6.5.7	<i>User Interface</i> halaman Tracking riwayat pasien	71
BAB 7 IMPLEMENTASI		72
7.1	Implementasi <i>User Interface</i>	72
7.1.1	<i>User Interface</i> Login Admin	72

7.1.2	<i>User Interface</i> Halaman Input data petugas	73
7.1.3	<i>User Interface</i> Halaman Ruang rawat	74
7.1.4	<i>User Interface</i> Halaman Status	74
7.1.5	<i>User Interface</i> Login Petugas	75
7.1.6	<i>User Interface</i> Halaman Dashboard Petugas (Grafik)	76
7.1.7	<i>User Interface</i> Halaman Daftar Pasien	77
7.1.8	<i>User Interface</i> Halaman Tracking riwayat pasien	78
7.2	Implementasi <i>Database</i>	78
7.2.1	Implementasi Table Admin	79
7.2.2	Implementasi Tabel log riwayat pasien	79
7.2.3	Implementasi Tabel pasien	80
7.2.4	Implementasi Tabel Petugas	80
7.2.5	Implementasi Tabel Ruang rawat	81
7.2.6	Implementasi Tabel Status pasien	81
7.2.7	Implementasi Tabel tracking riwayat	81
7.3.	Implementasi Kode Program	82
7.3.1.	Implementasi <i>Login</i> Admin	82
7.3.2.	Implementasi <i>Login</i> Petugas	84
7.3.3.	Implementasi <i>Dashboard</i>	87
7.3.4.	Implementasi Tracking riwayat Pasien	89
7.3.5.	Implementasi Mengelola Pasien	90
7.3.6.	Implementasi Mengelola Petugas	100
7.3.7.	Implementasi Mengelola Ruang Rawat	104
7.3.8.	Implementasi Mengelola Status Pasien	107
7.3.9.	Implementasi log riwayat Pasien	110
BAB 8	PENGUJIAN	112
8.1	Hasil Pengujian Menggunakan <i>Black Box Testing</i>	112
8.2	Hasil Pengujian Menggunakan <i>Browser Compatibility Testing</i>	123
BAB 9	PENUTUP	125
9.1	Kesimpulan	125
9.2	Saran	126
DAFTAR PUSTAKA	127

LAMPIRAN	130
----------------	-----

DAFTAR GAMBAR

Gambar 2. 1 Struktur Organisasi Dinas Komunikasi dan Informatika Kabupaten Gresik	7
Gambar 3. 1 Metode Pengembangan waterfall	9
Gambar 3. 2 Metode Pengembangan waterfall	17
Gambar 3. 3 Alur kerja pada MVC	20
Gambar 4. 1 Diagram Alir pengembangan Sistem Informasi Pencatatan COVID-19 (SIPCOP)	23
Gambar 5. 1 Use Case Diagram Sistem Informasi Pencatatan COVID-19 (SIPCOP)	30
Gambar 5. 2 Activity Diagram Login Admin	43
Gambar 5. 3 Activity Diagram Login petugas	44
Gambar 5. 4 Activity Diagram Input data petugas	45
Gambar 5. 5 Activity Diagram edit data petugas.....	46
Gambar 5. 6 Activity Diagram hapus data petugas	47
Gambar 5. 7 Activity Diagram mengelola daftar status.....	48
Gambar 5. 8 Activity Diagram mengelola daftar ruang rawat	50
Gambar 5. 9 Activity Diagram input data pasien.....	51
Gambar 5. 10 Activity Diagram edit data pasien	52
Gambar 5. 11 Activity Diagram hapus data pasien.....	53
Gambar 5. 12 Activity Diagram melihat grafik harian dan bulanan	54
Gambar 5. 13 Activity Diagram Tracking riwayat pasien	55
Gambar 6. 1 Arsitektur Sistem Informasi Pencatatan COVID-19	56
Gambar 6. 2 Sequence Diagram Login Admin	57
Gambar 6. 3 Sequence Diagram Login Petugas.....	58
Gambar 6. 4 Sequence Diagram input data petugas.....	58
Gambar 6. 5 Sequence Diagram input data pasien	59
Gambar 6. 6 Sequence Diagram input ruang rawat	60
Gambar 6. 7 Sequence Diagram input status pasien.....	60
Gambar 6. 8 Sequence Diagram Tracking riwayat pasien.....	61
Gambar 6. 9 Class Diagram Sistem Informasi Pencatatan COVID-19	62
Gambar 6. 10 Physical Data Model Sistem Informasi Pencatatan COVID-19	63
Gambar 6. 11 User Interface Login Admin	66
Gambar 6. 12 User Interface Login petugas	66
Gambar 6. 13 User Interface Halaman Dashboard Admin.....	67
Gambar 6. 14 User Interface halaman Ruang Rawat.....	68
Gambar 6. 15 User Interface halaman Status	69
Gambar 6. 16 User Interface halaman Dashboard petugas (Grafik)	70
Gambar 6. 17 User Interface halaman Input pasien.....	71
Gambar 6. 18 User Interface halaman Tracking riwayat pasien	71
Gambar 7. 1 User Interface Login Admin	72
Gambar 7. 2 User Interface Halaman daftar petugas.....	73
Gambar 7. 3 User Interface Halaman Input data petugas	73
Gambar 7. 4 User Interface Halaman daftar Ruang rawat.....	74
Gambar 7. 5 User Interface Halaman input Ruang rawat.....	74
Gambar 7. 6 User Interface Halaman status	75

Gambar 7. 7 User Interface tambah status	75
Gambar 7. 8 User Interface Login Petugas	76
Gambar 7. 9 User Interface Halaman Dashboard Petugas (Grafik harian)	76
Gambar 7. 10 User Interface Halaman Dashboard Petugas (Grafik bulanan)	77
Gambar 7. 11 User Interface Halaman Daftar Pasien	77
Gambar 7. 12 User Interface Halaman tambah Pasien.....	78
Gambar 7. 13 User Interface Halaman Tracking riwayat pasien	78
Gambar 7. 14 Database pada Sistem Informasi Pencatatan COVID-19.....	79
Gambar 8. 1 Hasil Compatibility Testing	124

DAFTAR TABEL

Tabel 3. 1 Komponen pada Activity Diagram	11
Tabel 3. 2 Komponen pada Sequence Diagram.....	12
Tabel 3. 3 Komponen pada Usecase Diagram	14
Tabel 3. 4 Komponen pada Class Diagram	15
Tabel 5. 1 Identifikasi Aktor	26
Tabel 5. 2 Kebutuhan Fungsional.....	27
Tabel 5. 3 Kebutuhan Non Fungsional	29
Tabel 5. 4 Skenario Use Case Login Admin.....	31
Tabel 5. 5 Skenario Use Case Input data petugas.....	32
Tabel 5. 6 Skenario Use Case melihat data petugas	32
Tabel 5. 7 Skenario Use Case Edit data petugas	33
Tabel 5. 8 Skenario Use Case Delete Petugas.....	34
Tabel 5. 9 Skenario Use Case mengelola daftar status	34
Tabel 5. 10 Skenario Use Case mengelola daftar ruang rawat.....	35
Tabel 5. 11 Skenario Use Case Login petugas.....	37
Tabel 5. 12 Skenario Use Case Input data pasien	37
Tabel 5. 13 Skenario Use Case melihat data pasien	38
Tabel 5. 14 Skenario Use Case edit data pasien	39
Tabel 5. 15 Skenario Use Case delete pasien	40
Tabel 5. 16 Skenario Use Case Melihat Grafik harian dan bulanan.....	40
Tabel 5. 17 Skenario Use Case Tracking riwayat pasien	41
Tabel 5. 18 Skenario Use Case Melihat daftar status	41
Tabel 5. 19 Skenario Use Case edit status pasien.....	42
Tabel 7. 1 Kode Program membuat tabel admin.....	79
Tabel 7. 2 Kode Program membuat tabel log riwayat pasien	79
Tabel 7. 3 Kode Program membuat tabel pasien	80
Tabel 7. 4 Kode Program membuat tabel petugas	80
Tabel 7. 5 Kode Program membuat tabel ruang rawat	81
Tabel 7. 6 Kode Program membuat tabel status pasien	81
Tabel 7. 7 Kode Program membuat tabel tracking riwayat	82
Tabel 7. 8 Kode Program Controller Login Admin	82
Tabel 7. 9 Kode Program Model Login Admin	83
Tabel 7. 10 Kode Program Controller Login Petugas	85
Tabel 7. 11 Kode Program Model Login Petugas.....	86
Tabel 7. 12 Kode Program Controller dashboard	87
Tabel 7. 13 Kode Program Model tracking riwayat pasien	89
Tabel 7. 14 Kode Program Controller Pasien.....	91
Tabel 7. 15 Kode Program Model Pasien	98
Tabel 7. 16 Kode Program Controller Petugas	100
Tabel 7. 17 Kode Program Model Petugas	103
Tabel 7. 18 Kode Program Controller Ruang rawat	104
Tabel 7. 19 Kode Program Model Ruang rawat.....	106
Tabel 7. 20 Kode Program Controller Status pasien	107
Tabel 7. 21 Kode Program Model Status pasien.....	109

Tabel 7. 22 Kode Program Model Log riwayat pasien	110
Tabel 8. 1 Pengujian pada proses Login Admin	112
Tabel 8. 2 Pengujian pada proses Login Petugas.....	113
Tabel 8. 3 Pengujian pada proses Input data petugas	114
Tabel 8. 4 Pengujian pada proses Melihat data petugas	115
Tabel 8. 5 Pengujian pada proses edit data petugas	115
Tabel 8. 6 Pengujian pada proses delete data petugas	116
Tabel 8. 7 Pengujian pada proses mengelola daftar status	116
Tabel 8. 8 Pengujian pada proses mengelola daftar ruang rawat.....	117
Tabel 8. 9 Pengujian pada proses input data pasien	118
Tabel 8. 10 Pengujian pada proses melihat data pasien.....	119
Tabel 8. 11 Pengujian pada proses edit data pasien	120
Tabel 8. 12 Pengujian pada proses delete pasien	120
Tabel 8. 13 Pengujian pada proses melihat grafik harian dan bulanan.....	121
Tabel 8. 14 Pengujian pada proses tracking riwayat pasien	121
Tabel 8. 15 Pengujian pada proses melihat daftar status.....	122
Tabel 8. 16 Pengujian pada proses edit status pasien	123
Tabel 8. 17 Browser Compatibility Testing	123

DAFTAR LAMPIRAN

Lampiran 1. 1 Foto kegiatan Praktik Kerja Lapangan bersama Pembimbing Lapangan	130
Lampiran 1. 2 Foto kegiatan bersama Kepala Diskominfo Kabupaten Gresik	130
Lampiran 2. 1 Form Validasi User Acceptance Testing	131

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Mengutip World Health Organization (WHO 2020), COVID-19 adalah penyakit menular yang disebabkan oleh virus corona yang baru ditemukan. Virus Corona sendiri adalah sebuah keluarga virus yang ditemukan pada manusia dan hewan, sebagian virusnya dapat menginfeksi manusia serta menyebabkan berbagai penyakit, mulai dari penyakit umum seperti flu, hingga penyakit-penyakit yang lebih fatal, seperti Middle East Respiratory Syndrome (MERS) dan Severe Acute Respiratory Syndrome (SARS). Kasus ini pertama kali ditemukan di Kota Wuhan, provinsi Hubei, China. Virus COVID-19 dapat menyebar melalui tetesan air liur, cairan hidung saat bersin, dan batuk. Orang-orang yang hidup di dan melakukan perjalanan ke daerah-daerah dimana virus ini menyebar berpotensi terjangkit virus corona. WHO pun menginformasikan bahwa saat ini belum ada vaksin atau perawatan khusus untuk COVID-19.

Penambahan jumlah kasus COVID-19 berlangsung sangat cepat dan sudah terjadi penyebaran ke luar wilayah Wuhan dan negara lain. Hingga pada tanggal 2 Maret 2020, Presiden Jokowi mengonfirmasi adanya dua orang di Indonesia yang positif terjangkit virus corona (CNN 2020). Jumlah kasus positif COVID-19 di Indonesia pun semakin menyebar dengan cepat, hingga pada tanggal 26 Juni 2020 Provinsi Jawa Timur menjadi provinsi dengan kasus positif paling tinggi dengan tambahan 356 kasus terkonfirmasi positif COVID-19, sehingga jumlah total pasien mencapai 10.901 angka tersebut melebihi jumlah kasus di Jakarta yang selama ini menjadi daerah dengan kasus tertinggi di Indonesia (Putra 2020). Pada Provinsi Jawa Timur, Kabupaten Gresik sendiri menduduki peringkat tertinggi ketiga setelah Surabaya dan Sidoarjo dengan jumlah pertambahan kasus COVID-19 mencapai 39 orang perhari yang semakin menambah jumlah kasus terkonfirmasi (Liputan6 2020).

Karena tingginya angka penderita COVID-19 di Kabupaten Gresik, sehingga 18 Rumah Sakit yang menjadi Rumah Sakit rujukan pasien COVID-19 di Kabupaten Gresik sudah tidak mampu lagi menampung pasien positif (Mubyarsah 2020). Oleh karena itu Gubernur Jawa Timur Khofifah Indar Parawansa, meminta Pemerintah Kabupaten Gresik untuk menjadikan Stadion Gelora Joko Samudro (GJS) menjadi Rumah Sakit darurat karena melihat dari keberhasilan RS Darurat Lapangan di Surabaya yang memberikan dampak baik dengan tingkat kesembuhan pasien positif COVID-19 yang cukup tinggi. Bupati Gresik pun melakukan berbagai persiapan fasilitas yang akan digunakan pada Rumah Sakit darurat ini yang akan menampung sebanyak 140 pasien dengan rincian 80 orang di zona merah, 40 orang di zona kuning, dan 20 orang di zona hijau (Saptiyulda 2020).

Untuk membantu Pemerintah Kabupaten Gresik menyiapkan kebutuhan yang akan digunakan, maka sesuai dengan tugas dan fungsinya Dinas Komunikasi dan Informatika Kabupaten Gresik sebagai penyelenggaraan urusan pemerintah bidang Komunikasi dan Informatika untuk daerah Kabupaten Gresik, membantu untuk mengembangkan Sistem Informasi

Pencatatan COVID-19 (SIPCOP). Oleh karena itu Kepala Dinas Komunikasi dan Informatika Kabupaten Gresik memberikan amanat kepada kelompok PKL Kami untuk membangun sistem ini. Kemudian di lakukan rapat yang diadakan dengan kepala Dinas Komunikasi dan Informatika Kabupaten Gresik, Kepala Seksi Pengelolaan dan Pelayanan Informasi, Kepala Seksi Aplikasi dan Pengembangan Informatika, Subbagian Umum dan Kepegawaian, dan Pembimbing lapangan selama praktik kerja lapangan untuk membahas lebih dalam mengenai kebutuhan dari Sistem Informasi yang akan dibangun agar mendapatkan hasil akhir aplikasi yang sesuai dengan kebutuhan yang diinginkan pada Rumah Sakit Darurat Gelora Joko Samudro. Sistem Informasi ini digunakan untuk memudahkan pencatatan masuk dan keluarnya pasien pada Rumah Sakit Darurat Gelora Joko Samudra (GJS). Sistem Informasi ini akan dikembangkan menggunakan metode *waterfall* yang terdiri dari lima tahapan perancangan mulai dari analisis kebutuhan, desain sistem, implementasi, pengujian dan kemudian dilakukan uji coba dengan metode Black box serta dilakukan perbaikan jika ada kesalahan pada sistem yang dibuat. Sistem informasi ini diimplementasikan menggunakan bahasa pemrograman PHP dengan menggunakan *framework* Codeigniter 3.0, sedangkan untuk implementasi tampilan agar dapat mudah digunakan oleh petugas pencatatan maka tampilan sistem informasi ini dirancang menggunakan *framework* Bootstrap 4.0. Sistem Informasi ini juga terintegrasi dengan API (Application Programming Interface) dari Dinas Kependudukan Dan Pencatatan Sipil Kabupaten Gresik. Dengan adanya aplikasi ini diharapkan dapat membantu untuk memudahkan pencatatan pasien di Rumah Sakit Darurat Gelora Joko Samudra (GJS) dan dapat memonitoring jumlah pasien dengan melihat sehingga dapat menghentikan rantai penyebaran COVID-19 pada Kabupaten Gresik, dan dapat menekan angka kasus terkonfirmasi positif COVID-19 agar semakin menurun.

1.2 Rumusan Masalah

Dari latar belakang sebelumnya maka dapat dibuat rumusan masalah sebagai berikut :

1. Bagaimana analisis kebutuhan dari Sistem Informasi pencatatan COVID-19 (SIPCOP) pada Rumah Sakit darurat Gelora Joko Samudro?
2. Bagaimana rancangan dan implementasi Sistem Informasi berbasis website pencatatan COVID-19 (SIPCOP) pada Rumah Sakit darurat Gelora Joko Samudro menggunakan framework Codeigniter ?
3. Bagaimana hasil pengujian fungsional dan non-fungsional dari pembangunan website Sistem Informasi Pencatatan COVID-19 (SIPCOP) pada Rumah Sakit darurat Gelora Joko Samudro?

1.3 Tujuan

Berdasarkan rumusan masalah di atas, maka tujuan penulisan laporan ini adalah sebagai berikut :

1. Mengetahui analisis kebutuhan dari Sistem Informasi berbasis website pencatatan Covid-19 (SIPCOP) pada Rumah Sakit darurat Gelora Joko Samudro.
2. Menghasilkan rancangan dan implementasi Sistem Informasi pencatatan Covid-19 (SIPCOP) pada Rumah Sakit darurat Gelora Joko Samudro.
3. Mengetahui hasil pengujian fungsional dan non-fungsional dari pembangunan website Sistem Informasi Pencatatan COVID-19 (SIPCOP) pada Rumah Sakit darurat Gelora Joko Samudro

1.4 Batasan Masalah

Adapun hal-hal yang menjadi batasan masalah agar analisis, perancangan, dan implementasi sesuai yang diinginkan, yaitu :

1. Sistem Informasi pencatatan COVID-19 (SIPCOP) merupakan aplikasi berbasis web yang dikembangkan menggunakan *framework* Codeigniter versi 3.0
2. Sistem Informasi pencatatan COVID-19 (SIPCOP) hanya digunakan pada Rumah Sakit darurat Gelora Joko Samudro.

1.5 Manfaat

Manfaat perancangan Sistem Informasi pencatatan COVID-19 (SIPCOP), yaitu sebagai berikut :

1. Untuk menghasilkan Sistem Informasi berbasis website yang dapat digunakan di Rumah Sakit Darurat Gelora Joko Samudro.
2. Untuk memudahkan pemantauan area mana saja pada Kabupaten Gresik yang intensitas penyebarannya sangat cepat.
3. Untuk mengurangi rantai penyebaran COVID-19 pada daerah yang rentan tersebar virus dengan memantau grafik pertumbuhan berdasarkan status pasien pada Kabupaten Gresik.

1.6 Waktu dan Tempat Pelaksanaan

Praktik Kerja Lapangan (PKL) ini dilaksanakan pada tanggal 22 Juni 2020 – 22 Agustus 2020 di Dinas Komunikasi dan Informatika Kabupaten Gresik yang berlokasi di Jl. Dr.Wahidin S.H. No. 60, Randuagung, Kebomas, Kabupaten Gresik, Jawa Timur 61121.

1.7 Sistematika Penulisan

BAB 1 Pendahuluan

Bab ini membahas mengenai latar belakang, rumusan masalah, tujuan, batasan masalah, manfaat, dan waktu dan tempat pelaksanaan Praktik Kerja Lapangan serta sistematika penulisan yang digunakan dalam penulisan Laporan.

BAB 2 Profil Objek PKL

Bab ini menjelaskan gambaran profil instansi, visi misi, dan struktur organisasi pada Dinas Komunikasi dan Informatika Kabupaten Gresik.

BAB 3 Tinjauan Pustaka

Bab ini akan menjelaskan dasar dasar berbagai teori yang membantu pengerjaan Sistem Informasi pencatatan COVID-19 (SIPCOP).

BAB 4 Metodologi

Bab ini akan menjelaskan tahapan-tahapan metodologi penulisan yang digunakan dalam penyusunan laporan dan pengerjaan Sistem Informasi pencatatan COVID-19 (SIPCOP).

BAB 5 Perencanaan Kebutuhan

Bab ini akan mengidentifikasi apa saja kebutuhan sistem dan identifikasi aktor dalam pengembangan Sistem Informasi pencatatan COVID-19 (SIPCOP).

BAB 6 Perancangan

Bab ini akan menjelaskan rancangan keseluruhan arsitektur Sistem Informasi pencatatan COVID-19 (SIPCOP) dengan menggunakan tools Unified Modeling Language (UML).

BAB 7 Implementasi

Bab ini membahas tentang implementasi dari perancangan sistem informasi dalam bentuk desain *user interface*, kode program, dan *database*.

BAB 8 Pengujian

Bab ini akan dipaparkan hasil pengujian yang dilakukan pada sistem yang telah dibuat. Pengujian dilakukan dengan metode *Black Box Testing* untuk pengujian kebutuhan fungsional dan metode *Compatibility Testing* untuk pengujian kebutuhan *Compatibility Browser*.

BAB 9

Penutup

Bab ini akan menguraikan kesimpulan dan saran berdasarkan hasil pengembangan Sistem Informasi Pencatatan COVID-19 (SIPCOP).

BAB 2 PROFIL OBJEK PKL

2.1 Profil Dinas Komunikasi dan Informatika Kabupaten Gresik

Dinas Komunikasi dan Informatika Kabupaten Gresik merupakan institusi dibawah naungan pemerintah dan berada dalam ruang lingkup Kementerian Komunikasi dan Informatika (Kemkominfo). Sesuai Undang-Undang Nomor 39 Tahun 2008 tentang Kementerian Negara, Kementerian Kominfo merupakan perangkat Pemerintah Republik Indonesia ini membidangi urusan yang ruang lingkupnya disebutkan dalam Undang-Undang Dasar Negara Republik Indonesia Tahun 1945, yaitu informasi dan komunikasi. Kementerian Komunikasi dan Informatika mempunyai tugas menyelenggarakan urusan pemerintahan di bidang komunikasi dan informatika untuk membantu Presiden dalam menyelenggarakan pemerintahan negara. Dengan demikian, Dinas Komunikasi dan Informatika yang berada di bawah naungan pemerintah kabupaten Gresik memiliki tugas membantu Bupati dalam melaksanakan urusan pemerintahan bidang Komunikasi dan Informatika, urusan pemerintahan bidang Statistik dan urusan pemerintahan bidang persandian.

2.2 Visi dan Misi Dinas Komunikasi dan Informatika Kabupaten Gresik

Visi :

"mewujudkan teknologi informasi dan komunikasi untuk masa depan yang lebih baik dan kehidupan yang berkualitas "

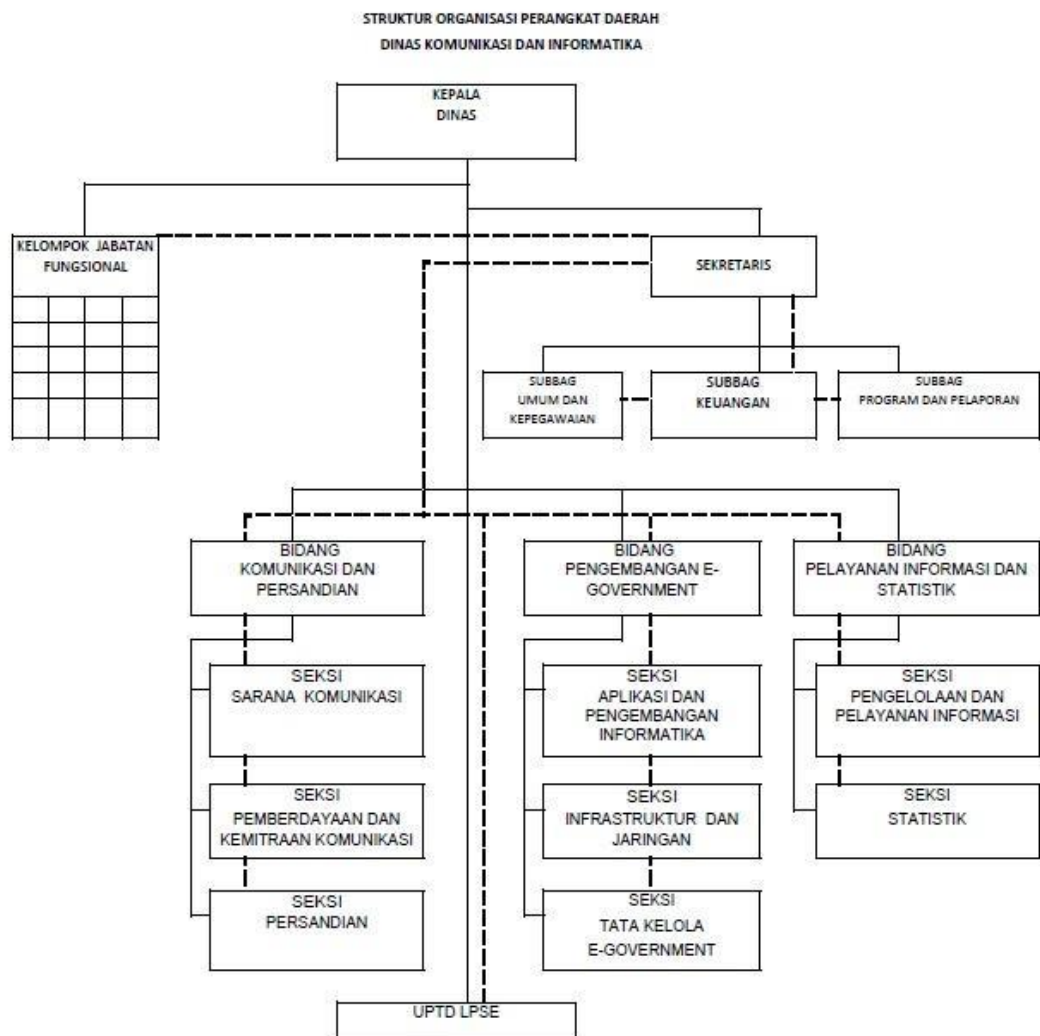
"information communication and technology to be better future for better life "

Misi :

1. Menyediakan Pelayanan Publik Berbasis E- Government
2. Meningkatkan Pengelolaan dan Pelayanan Informasi dengan Dukungan Data yang Valid, Akurat dan Uptodate
3. Meningkatkan Komunikasi yang Efektif dengan Dukungan Sarana dan Prasarana yang Memadai

2.3 Struktur Organisasi Dinas Komunikasi dan Informatika Gresik

Selama kami melakukan Praktik Kerja Lapangan, kami berada di seksi aplikasi dan pengembangan informatika yang dikepalai oleh unit bidang pengembangan E-Government. Berikut adalah struktur organisasi pada Dinas Komunikasi dan Informatika Kabupaten Gresik.



Gambar 2. 1 Struktur Organisasi Dinas Komunikasi dan Informatika Kabupaten Gresik
Sumber: Website Dinas Komunikasi dan Informatika Kabupaten Gresik (2020)

2.3.1 Seksi Aplikasi dan Pengembangan Informatika

Seksi aplikasi dan pengembangan Informatika merupakan salah satu divisi yang ada di Dinas Komunikasi dan Informatika Kabupaten Gresik yang berada di bawah Unit bidang pengembangan E-Government. Seksi aplikasi dan pengembangan informatika memiliki tugas untuk merencanakan, menyiapkan bahan pelaksanaan dan mengkoordinasikan e-Government dan pemberdayaan Teknologi Informasi dan Komunikasi (TIK), Pengembangan Aplikasi serta persandian dan keamanan informasi. Seksi Aplikasi dan pengembangan memiliki beberapa fungsi diantaranya yaitu memfasilitasi integrasi pelayanan publik e-Government, melaksanakan pembinaan dan pengembangan perangkat lunak, serta melaksanakan monitoring, evaluasi, dan pelaporan aplikasi informatika. Salah satunya proyek yang diterima oleh Diskominfo Gresik yaitu mengembangkan aplikasi berbasis web untuk Rumah Sakit darurat Gelora Joko Samudro yang bertujuan untuk mempermudah petugas dalam pencatatan

keluar dan masuknya pasien Covid-19, tracking status pasien, dan update jumlah pasien Covid-19 yang berada dalam Rumah sakit Gelora Joko Samudro.

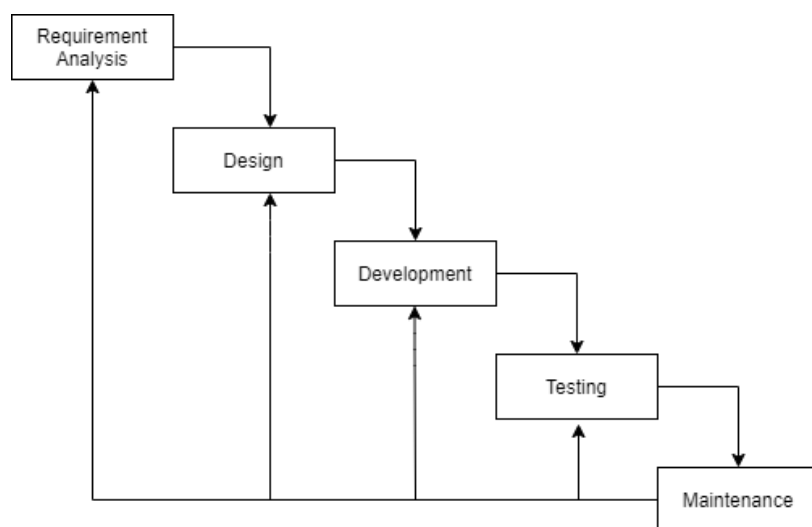
BAB 3 TINJAUAN PUSTAKA

3.1 Sistem Informasi

Sistem adalah sekumpulan komponen yang saling terkait dengan batas yang jelas, bekerja bersama untuk mencapai tujuan dengan menerima input dan menghasilkan output dalam proses transformasi terorganisir (O'Brien & Marakas, 2008). Sedangkan pengertian informasi merupakan kumpulan fakta yang terorganisir sehingga memiliki nilai tambah. Dari pengertian-pengertian tersebut, dapat ditarik kesimpulan bahwa sistem informasi merupakan komponen yang saling bekerja sama untuk mengumpulkan, mengolah, menyimpan dan menyebarkan informasi untuk pendukung pengambilan keputusan dalam sebuah organisasi.

3.1.1 Pengembangan Sistem Informasi

Pengembangan sistem informasi merupakan proses untuk merencanakan, mengembangkan, dan mengimplementasikan sistem informasi dengan menggunakan metode, teknik dan alat bantu pengembangan tertentu (Dezaneru 2016). Salah satu pendekatan metode pengembangan sistem informasi adalah metode *waterfall*. Metode *waterfall* merupakan suatu metode dalam pengembangan software dimana pengerjaannya harus dilakukan secara berurutan dimulai dari analisis kebutuhan, pemodelan (design), implementasi, testing dan maintenance (Pressman 2012).



Gambar 3. 1 Metode Pengembangan *waterfall*

Sumber : (Pressman 2012)

Dari gambar 2.2 diatas, dapat diketahui bahwa metode *waterfall* terdiri dari 5 fase. Fase pertama adalah *Requirement analysis*. Fase ini pengembang sistem diperlukan komunikasi untuk memahami perangkat lunak yang diharapkan oleh pengguna. Fase kedua yaitu *System Design*. Pada fase ini mempelajari spesifikasi kebutuhan dari tahap sebelumnya dan desain sistem akan disiapkan. Fase ketiga yaitu *implementation*. Pada fase ini terjadi proses menerjemahkan perancangan desain ke bentuk yang dimengerti oleh mesin dengan menggunakan kode bahasa pemrograman. Fase keempat yaitu *Testing*, dimana akan dilakukan pengujian untuk mengecek setiap kegagalan maupun kesalahan pada perangkat lunak. Fase terakhir yaitu *Maintenance* dimana perangkat lunak yang sudah jadi akan dijalankan serta dilakukan pemeliharaan termasuk memperbaiki kesalahan yang ditemukan pada langkah sebelumnya.

3.1.2 Rekayasa Kebutuhan

Rekayasa kebutuhan digunakan sebagai pembangun jembatan menuju desain dan pembangunan perangkat lunak dengan menyediakan mekanisme yang tepat untuk memahami apa yang diinginkan user, menganalisis kebutuhan, menguji kelayakan, menetapkan solusi dari kedua belah pihak dan mengelola kebutuhan yang akan diwujudkan ke sistem operasional (Proboyekti 2007).

Proses Rekayasa Kebutuhan terdiri dari 7 fungsi yaitu :

a. *Inception* (Permulaan)

Inception atau permulaan, merupakan awal dari terjadinya pembicaraan tentang kebutuhan akan software. Permulaan ini dapat terjadi karena dari pembicaraan biasa, kebutuhan bisnis yang dirasakan, adanya pasar potensial, atau munculnya layanan potensial yang dapat dilakukan oleh software.

b. *Elicitation* (Pengungkapan)

Aktifitas dilakukan dalam bentuk pertemuan, dimana setiap stakeholder yang diundang diminta untuk membuat daftar kebutuhan software yang disertai penjelasan tentang karakteristik atau detail dari kemampuan tersebut.

c. *Elaboration* (Penjelasan)

Fokus pada pemodelan fungsi, fitur dan batasan dari perangkat lunak. Dalam kasus OOP, maka class, atribut dan hubungan antar class diidentifikasi pada aktifitas ini.

d. *Negotiation*

Jika terjadi konflik kepentingan karena perbedaan kebutuhan antara bagian pada klien, atau antar end-user, maka aktifitas negosiasi diperlukan untuk membuat prioritas kebutuhan, mengevaluasi tiap kebutuhan untuk

mengurangi atau mengubah sesuai dengan yang dimaksud pihak-pihak yang berkonflik.

e. *Spesification*

Spesifikasi dapat berupa dokumen, model grafik, model matematika, skenario atau prototype yang merupakan produk akhir dari rekayasa kebutuhan. Apa yang disajikan sebagai spesifikasi merupakan hasil identifikasi kebutuhan melalui aktifitas-aktifitas sebelumnya. Spesifikasi menggambarkan fungsi dan kinerja dari perangkat lunak dan batasan-batasan yang ditentukan.

f. *Validation* (Validasi)

Validasi yaitu menguji kualitas dari spesifikasi untuk memastikan kebutuhan yang dinyatakan dapat diterima atau sepaham, konsisten, lengkap dan bebas kesalahan. Mekanisme yang dapat dilakukan adalah formal technical review atau pertemuan evaluasi teknis.

g. *Management*

Manajemen yaitu mengelola kebutuhan dengan identifikasi, kontrol dan mengikuti perkembangan kebutuhan software yang dikerjakan selama proyek dan perubahan-perubahan yang terjadi.

3.1.3 *Unified Modelling Language*


Unified Modelling Language (UML) adalah suatu metode atau bahasa standar secara visual sebagai sarana perancangan sistem berorientasi objek (Sora 2015). Saat ini UML sudah menjadi bahasa standar dalam penulisan *blue print* software sebab berisi tentang informasi detail mengenai kebutuhan-kebutuhan yang akan diimplementasikan. Jenis-jenis diagram UML untuk menggambarkan sistem yaitu *activity diagram*, *sequence diagram*, *use case diagram*, dan *class diagram*. Berikut penjelasan terkait model-model *diagram* UML.




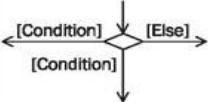
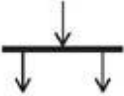
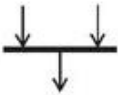
a. *Activity Diagram*

Activity Diagram merupakan diagram menjelaskan tentang alir kegiatan dalam program yang sedang dirancang, bagaimana proses alir berawal, keputusan yang mungkin terjadi, dan bagaimana sistem akan berakhir. Activity diagram juga dapat menjelaskan metode parallel yang mungkin terjadi pada beberapa eksekusi (Ansori 2020).

Simbol-simbol yang digunakan pada *activity diagram* akan dijelaskan pada Tabel 3.1

Tabel 3. 1 Komponen pada Activity Diagram

Simbol	Nama	Keterangan
	<i>Start Point (Initial Node)</i>	<i>Activity</i> dimulai

	<i>End Point (Activity Final Node)</i>	Activity berakhir
	<i>Activity</i>	Memperlihatkan bagaimana kelas-kelas antarmuka berinteraksi satu sama lain
	<i>Edge (Control Flow)</i>	Menghubungkan satu simbol dengan simbol lainnya.
	<i>Decision</i>	Percabangan seleksi-kondisi
	<i>Fork</i>	Membuat cabang dari satu <i>activity</i>
	<i>Join</i>	Menggabungkan dua <i>activity</i>


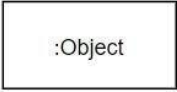


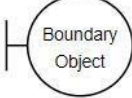
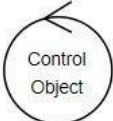
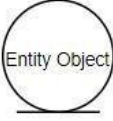



Sumber: (Ansori 2020)

b. Sequence Diagram

Sequence Diagram merupakan suatu diagram yang menjelaskan interaksi objek dan menunjukkan (memberi tanda atau petunjuk) komunikasi diantara objek-objek tersebut. Sequence diagram digunakan untuk menjelaskan perilaku pada sebuah scenario dan menggambarkan bagaimana entitas dan sistem berinteraksi, termasuk pesan yang dipakai saat berinteraksi (Ansori 2020). Penjelasan tentang komponen-komponen pada *Sequence Diagram* akan dijelaskan pada Tabel 3.2

Tabel 3. 2 Komponen pada Sequence Diagram

Simbol	Nama	Keterangan
--------	------	------------

	<i>Actor</i>	Subjek yang berinteraksi dengan sistem
	<i>Object</i>	Representasi dari class/object. Menunjukkan cara objek berperilaku pada sebuah sistem
	<i>Lifeline</i>	Umur hidup sebuah object
	<i>Activation</i>	Menunjukkan bahwa object sedang berinteraksi dengan object lain
	<i>Boundary Object</i>	Representasi dari batasan sistem (<i>boundary</i>). Biasanya berupa <i>user interface</i>
	<i>Control Object</i>	Mengatur aliran informasi dari sebuah skenario
	<i>Entity Object</i>	Penyimpanan data atau informasi
	<i>Message Entry</i>	Menunjukkan <i>object</i> sedang berinteraksi dengan <i>object</i> lain
	<i>Message to Self/Return</i>	Mengembalikan <i>object</i> setelah berinteraksi dengan <i>object</i> lain
	<i>Self-Message</i>	Pesan yang mewakili jenis pesan yang hidup di <i>lifetime</i> yang sama

Sumber: (Ansori 2020)


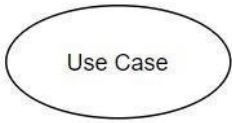
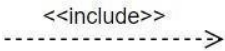
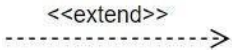
Objek pada *Sequence Diagram* diurutkan dari kiri ke kanan. *Sequence Diagram* terdiri dari 2 dimensi, yaitu vertikal dan horizontal. Dimensi vertikal

merepresentasikan waktu, sedangkan dimensi horizontal merepresentasikan objek-objek yang ada pada sebuah *Sequence Diagram* .

c. *Use Case Diagram*

Use Case Diagram merupakan diagram yang membantu dalam menyusun requirement sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua fitur yang ada pada sistem (Riadi 2013). Pada Tabel 3.3 akan dijelaskan komponen-komponen pada *Use Case Diagram*.

Tabel 3. 3 Komponen pada Usecase Diagram

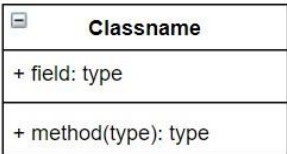


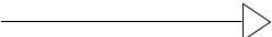
Simbol	Nama	Keterangan
 Actor	Actor	Subjek yang berinteraksi dengan sistem
 Use Case	Use Case	Gambaran fungsional sistem yang akan dibuat
	Include	Proses yang harus terpenuhi agar proses selanjutnya dapat dijalankan
	Extend	Proses perluasan yang dapat dipanggil jika kondisi atau syarat terpenuhi




Sumber: (Riadi 2013)

d. *Class Diagram*

Class Diagram merupakan diagram yang menggambarkan struktur dari sebuah sistem dengan cara mendefinisikan kelas-kelas yang dibangun dari sebuah sistem (Dini 2017). Kelas-kelas tersebut merepresentasikan kumpulan fungsi yang sesuai dengan kebutuhan sistem. Class diagram juga menunjukkan property dan operasi sebuah class dan Batasan-batasan yang terdapat dalam hubungan objek tersebut. Pada Tabel 3.4 akan dijelaskan komponen-komponen dari *Class Diagram*.

Tabel 3. 4 Komponen pada Class Diagram

Simbol	Nama	Keterangan
	<i>Class</i>	Representasi dari objek yang mendefinisikan atribut dan method
	<i>Association</i>	Hubungan antar kelas yang mereferensikan kelas yang lain, menggambarkan interaksi yang mungkin terjadi antara 1 kelas dengan kelas yang lain selama kelas tersebut tidak saling memiliki atau bukan bagian dari kelas tersebut
	<i>Directed Association</i>	Hubungan antar kelas dimana kelas yang satu digunakan oleh kelas yang lain
	<i>Generalization</i>	Hubungan antar kelas dimana sebuah kelas adalah kelas <i>child</i> yang lebih spesifik terhadap kelas <i>parent</i> nya. Kelas <i>child</i> memiliki kelas <i>parent</i> tetapi kelas <i>parent</i> tidak memiliki apa yang hanya dimiliki oleh kelas <i>child</i> . Jika hubungan di balik maka disebut <i>Inheritance</i> .

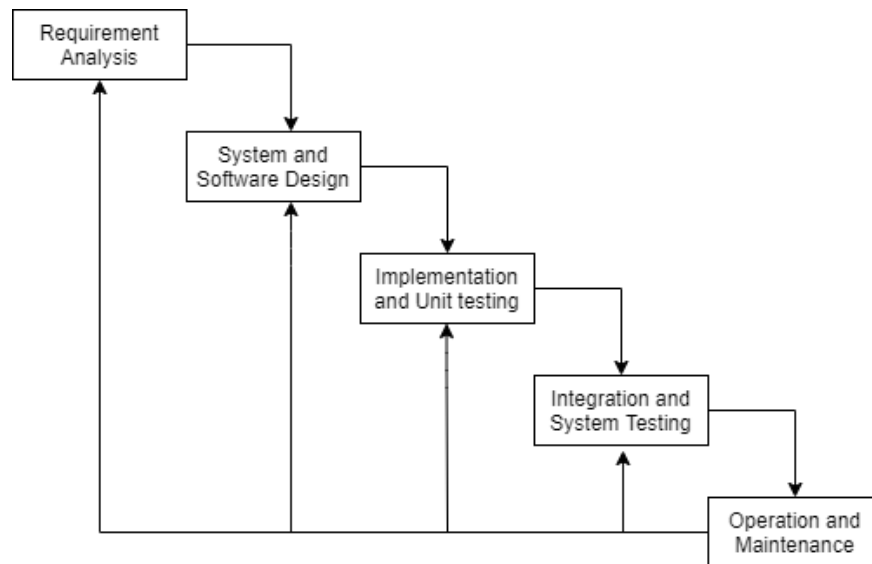
	<i>Aggregation</i>	Hubungan antar kelas dimana suatu kelas merupakan bagian dari kelas yang lain namun bersifat tidak wajib. Relasi ini juga menyatakan bahwa suatu kelas yang menjadi bagian dari kelas yang lain tidak akan dihapus meskipun kelas yang memilikinya dihapus
	<i>Composition</i>	Hubungan antar kelas yang saling bergantung, dimana suatu kelas merupakan bagian dari kelas yang lain dan bersifat wajib. Relasi ini juga mengindikasikan bahwa suatu kelas yang menjadi bagian kelas yang lain akan terhapus ketika kelas yang memilikinya dihapus
	<i>Dependency</i>	Hubungan antar kelas yang mengindikasikan ketergantungan sebuah kelas terhadap kelas yang lain

Sumber: (Dini 2017)

3.2 Teknologi Pengembangan Sistem

3.2.1 Metode Pengembangan Waterfall

Metode pengembangan *Waterfall* yang sering dinamakan siklus hidup klasik (*classic life cycle*) dimana hal ini menggambarkan pendekatan yang sistematis dan pengerjaannya harus dilakukan secara berurutan, dimulai dengan spesifikasi kebutuhan pengguna, perancangan desain, implementasi, pengujian dan pemeliharaan.



Gambar 3. 2 Metode Pengembangan waterfall

Sumber : (Rizky 2019)

Tahapan-tahapan pada waterfall yaitu

1. Requirement Analysis (Analisis Kebutuhan)

Pada tahap ini pengembang sistem diperlukan suatu komunikasi yang bertujuan untuk memahami software yang dibutuhkan pengguna dan batasan software. Informasi ini biasanya dapat diperoleh melalui wawancara, survey atau diskusi.

2. System Design (Perancangan Sistem)

Pada proses desain, dilakukan penerjemahan syarat kebutuhan ke sebuah perancangan desain perangkat lunak yang dapat diperkirakan sebelum dibuatnya proses pengkodean (coding). Proses ini berfokus pada struktur data, arsitektur perangkat lunak, representasi interface, dan detail algoritma prosedural.

3. Implementation (Pengembangan)

Pada tahap ini terjadi proses menerjemahkan perancangan desain ke bentuk yang dapat dimengerti oleh mesin, dengan menggunakan kode kode bahasa pemrograman. Kode program yang dihasilkan masih berupa modul-modul kecil yang nantinya akan digabungkan pada tahap berikutnya.

4. Integration & Testing (Pengujian)

Di tahap ini dilakukan penggabungan modul-modul yang sudah dibuat dan dilakukan pengujian ini dilakukan untuk mengetahui apakah software yang dibuat telah sesuai dengan desainnya dan fungsi pada software terdapat kesalahan atau tidak.

5. Operation & Maintenance (Pemeliharaan dan perbaikan)

Ini merupakan tahap terakhir dalam model waterfall. Software yang sudah jadi dijalankan serta dilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya. Perbaikan implementasi unit sistem dan peningkatan jasa sistem sebagai kebutuhan baru.

3.2.2 Aplikasi Berbasis Web

Aplikasi berbasis web adalah program yang tersimpan pada server kemudian dikirim melalui internet dan diakses dengan antar muka atau interface berupa web browser (Trifaris 2020). Aplikasi berbasis web juga dapat diakses melalui PC atau Laptop maupun mobile. Aplikasi web dapat dibangun menggunakan HTML, CSS, dan bahasa pemrograman yang mendukung pengembangan aplikasi web seperti PHP, Ruby, Python, dan lain-lain.

3.2.3 Bahasa Pemrograman

3.2.3.1 HTML

HTML (*Hypertext Markup Language*) merupakan Bahasa pemrograman yang digunakan untuk menampilkan dokumen pada browser dalam sebuah web. HTML bertujuan untuk mendefinisikan struktur dokumen web dan tata letak tampilan (Jayanti and Siska 2014). HTML juga merupakan bahasa pemrograman yang fleksibel dan dapat digabungkan dengan bahasa pemrograman lain seperti PHP, Javascript, Ruby, maupun Python. HTML menggunakan beragam tag dan atribut. Sebuah dokumen HTML ditandai dengan tag awal <HTML> dan diakhiri dengan tag </HTML>. Selain itu ada tag lain yang penting yaitu link yang mengandung URL (*Uniform Resource Locator*) yang berfungsi untuk merujuk pada lokasi dokumen lain di server yang sama atau computer lain yang berada di jaringan internet.

3.2.3.2 PHP

PHP merupakan bahasa pemrograman berbasis web yang memiliki kemampuan untuk memproses data dinamis. PHP dikatakan sebagai sebuah *server-side embedded script language* artinya sintaks-sintaks dan perintah yang kita berikan akan sepenuhnya dijalankan oleh server tetapi disertakan pada halaman HTML biasa (Usada, Yuniarsyah, and Rifani 2012). Pada awalnya PHP merupakan singkatan dari *Personal Home Page*, namun sekarang lebih dikenal dengan *Hypertext Preprocessor*.

3.2.3.3 CSS

CSS (*Cascading Style Sheet*) merupakan *stylesheet language* yang digunakan untuk menjelaskan tampilan sebuah halaman situs web dalam *markup language* (Jayan 2010). CSS merupakan *script* yang berguna untuk membuat halaman *web* menjadi bentuk *web* yang lebih indah dan menarik.

3.2.3.4 Javascript

Javascript adalah bahasa naskah berorientasi objek yang digunakan pada web browser dengan menambahkan fungsi interaktif pada halaman web (William 2011). memberikan fitur tambahan diluar kemampuan HTML dan CSS yang dapat menambah interaksi antara halaman web dengan pengguna

3.2.4 Software Pendukung

3.2.4.1 Visual Studio Code

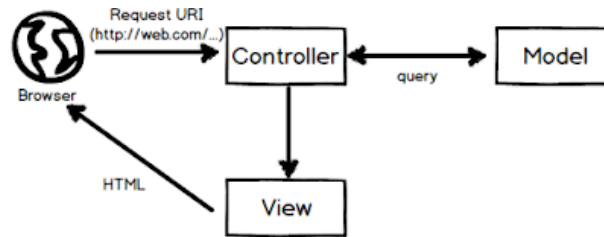
Visual Studio Code merupakan aplikasi *source-code editor* yang dikembangkan oleh Microsoft yang bisa digunakan pada berbagai platform seperti Windows, Linux, dan macOS. Visual Studio Code mendukung berbagai hampir semua bahasa pemrograman dengan berbagai fitur yang dimilikinya (Alfandi 2019). Visual Studio Code memiliki fitur *extensions* yang berfungsi untuk menambah fungsionalitas dari Visual Studio Code agar memudahkan pengembangan perangkat lunak dan penulisan kode.

3.2.5 Framework

Framework adalah sebuah kerangka kerja yang berisi kumpulan fungsi-fungsi dasar atau perintah yang biasa digunakan dalam mengembangkan suatu software agar software yang dibangun dapat dikerjakan lebih cepat dan terstruktur (Syafitri 2019). Berdasarkan pengertian tersebut sudah jelas bahwa *framework* memiliki fungsi utama untuk membantu dan memudahkan para developer dalam menyelesaikan suatu proyek pengembangan software.

3.2.5.1 Model-View-Controller

Model-View-Controller (MVC) adalah pola arsitektur yang memisahkan aplikasi dalam tiga komponen utama yaitu Model, View dan Controller. Masing-masing komponen ini dibangun untuk menangani aspek-aspek tertentu pembangunan aplikasi. MVC adalah salah satu kerangka pembangunan web standar industri paling sering digunakan untuk menciptakan proyek yang terukur an besar dan extensible (Jogianto 1999). Alur kerja sebuah MVC dipaparkan pada Gambar 3.3



Gambar 3. 3 Alur kerja pada MVC

Sumber: (Setiawan 2019)

Berikut adalah penjelasan dari Model, View, dan Controller (Setiawan 2019)

1. Model

Model adalah bagian yang berhubungan dengan manipulasi data didalam database misalnya insert,create, update dan delete dan lainnya. Model ini dihubungkan oleh kontrol aplikasi ke interface user.

2. View

View merupakan bagian yang menangani terkait tampilan user interface sebuah aplikasi. didalam aplikasi web biasanya pasti akan berhubungan dengan HTML dan CSS.

3. Controller

Controller bisa dikatakan sebagai otak dari sistem. karena controller yang menjadi penghubung antara bagian model dan view. Controller berfungsi untuk menerima request dan data dari user kemudian diproses dengan menghubungkan bagian model dan view sehingga bisa di terima oleh user.

3.2.5.2 CodeIgniter

CodeIgniter merupakan *framework* PHP yang dibuat berdasarkan *Model View Controller* (MVC). CI memiliki *library* yang lengkap untuk mengerjakan operasi-operasi yang umum dibutuhkan oleh aplikasi berbasis web misalnya mengakses database, memvalidasi form sehingga sistem yang dikembangkan lebih mudah dikerjakan. CI juga menjadi satu-satunya *framework* dengan dokumentasi yang lengkap dan jelas. Source code CI yang dilengkapi dengan comment didalamnya sehingga lebih memperjelas fungsi sebuah kode program dan CI yang dihasilkan sangat Bersih (clean) dan search Engine Friendly (SEF). Codeigniter juga dapat memudahkan developer dalam membuat aplikasi web berbasis PHP, karena framework sudah memiliki kerangka kerja sehingga tidak perlu menulis semua kode program dari awal. Selain itu, struktur dan susunan logis dari codeigniter membuat aplikasi menjadi semakin teratur dan dapat fokus pada fitur-fitur apa yang akan dibutuhkan dalam pembuatan aplikasi tersebut (Destiningrum and Adrian 2017).

3.2.5.3 Bootstrap

Bootstrap adalah sebuah framework yang dibuat dengan menggunakan bahasa dari HTML dan CSS, namun juga menyediakan efek javascript yang dibangun dengan menggunakan jquery. Bootstrap telah menyediakan kumpulan komponen class interface dasar yang telah dirancang

sedemikian rupa untuk menciptakan tampilan yang menarik, bersih dan ringan. Selain itu, bootstrap juga memiliki fitur grid yang berfungsi untuk mengatur layout yang bisa digunakan dengan sangat mudah dan cepat (Sanjaya and Hesinto 2018). Kita juga diberi keleluasaan dalam mengembangkan tampilan website yang menggunakan bootstrap yaitu dengan mengubah tampilan bootstrap dengan menambahkan class dan CSS sendiri.

3.3 Pengujian Perangkat Lunak

Pengujian perangkat lunak adalah proses mengeksekusi program atau aplikasi dengan maksud untuk menemukan bug dari suatu perangkat lunak yang dibuat. Hal ini juga dapat dinyatakan sebagai proses validasi dan verifikasi bahwa program perangkat lunak atau aplikasi yang telah dibuat telah memenuhi persyaratan teknis, dapat bekerja seperti yang diharapkan serta dapat diimplementasikan sesuai dengan harapan (Suhartono 2016).

Pengujian perangkat lunak memiliki tujuan yang berbeda dan objektif. Tujuan utama dari pengujian perangkat lunak adalah sebagai berikut:

- a. Menemukan cacat yang mungkin bisa dibuat oleh programmer ketika mengembangkan perangkat lunak.
- b. Mendapatkan kepercayaan dan memberikan informasi tentang tingkat kualitas.
- c. Untuk memastikan bahwa hasil akhir memenuhi bisnis dan kebutuhan pengguna.
- d. Untuk memastikan bahwa itu memenuhi BRS yang Spesifikasi Kebutuhan Bisnis dan SRS yang Kebutuhan Sistem Spesifikasi.
- e. Untuk mendapatkan kepercayaan dari pelanggan dengan menyediakan produk yang berkualitas.

Software pengujian membantu menyelesaikan aplikasi perangkat lunak atau produk terhadap bisnis dan kebutuhan pengguna. Hal ini sangat penting untuk memiliki cakupan tes yang baik untuk menguji aplikasi perangkat lunak sepenuhnya dan membuatnya yakin bahwa itu bekerja dengan baik dan sesuai dengan spesifikasinya (Suhartono 2016).

Pengujian perangkat lunak dapat dibedakan menjadi dua jenis, yaitu :

1. *White Box* – metode untuk melakukan pengujian kemampuan sistem dan pemrograman
2. *Black Box* – metode untuk menguji software tanpa mengetahui struktur internal kode atau program

3.3.1 Black Box Testing

Black Box Testing adalah metode pengujian perangkat lunak (biasa disebut dengan pengujian fungsional) untuk menguji perangkat lunak tanpa mengetahui struktur internal kode atau program. Dalam pengujian

ini, penguji hanya mengetahui fungsional dari sebuah sistem tanpa mengetahui bagaimana isi kode program sistem (Kurniawati 2018).

Kelebihan Black Box Testing yaitu:

- a. Efisien untuk segmen kode besar
- b. Akses kode tidak diperlukan
- c. Pemisahan antara perspektif pengguna dan pengembang

Kelemahan Black Box Testing yaitu:

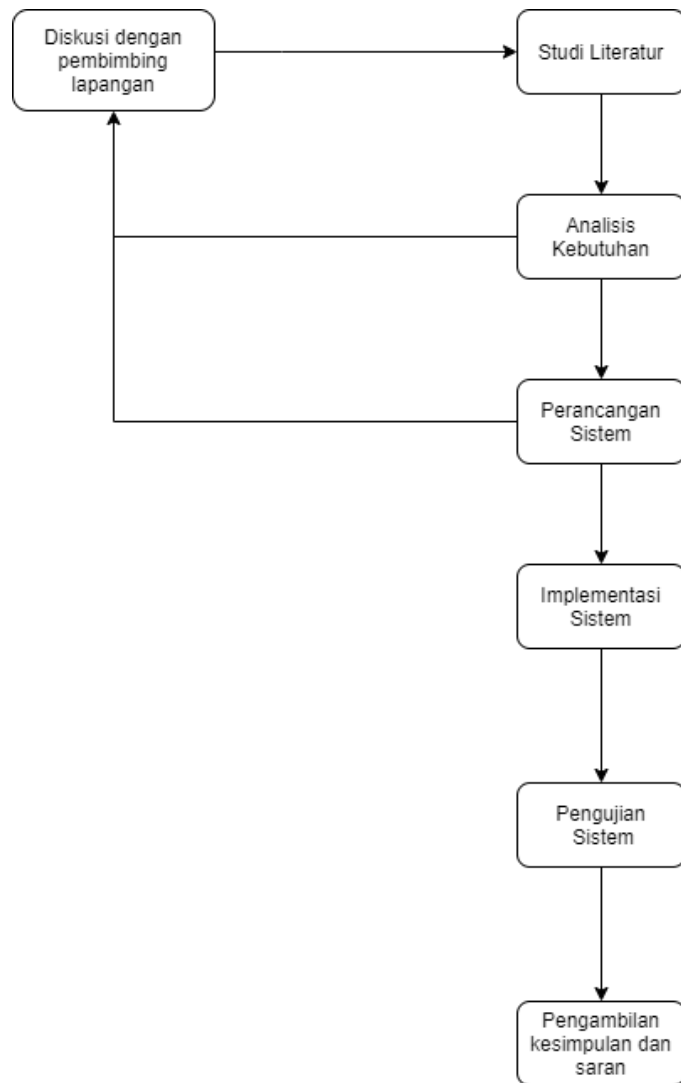
- a. Cakupan terbatas karena hanya sebagian kecil dari skenario pengujian yang dilakukan
- b. Pengujian tidak efisien karena keberuntungan tester dari pengetahuan tentang perangkat lunak internal

3.3.2 Compatibility Testing

Compatibility testing merupakan pengujian yang digunakan untuk memeriksa apakah perangkat lunak yang dikembangkan mampu berjalan pada hardware, sistem operasi, aplikasi, ataupun lingkungan jaringan yang berbeda. Pengujian kompatibilitas berfungsi untuk menentukan set lingkungan yang diharapkan agar dapat menjalankan aplikasi yang dikembangkan. Semakin banyak aplikasi dapat berjalan di jenis perangkat yang berbeda, maka semakin baik aspek kompatibilitasnya (Fatimah 2013).

BAB 4 METODOLOGI

Pada bab ini berisi uraian tentang tahap-tahap yang dilakukan secara sistematis dalam perancangan Sistem Informasi Pencatatan COVID-19 (SIPCOP). Tahap-tahap yang dilakukan diantaranya adalah diskusi dengan pembimbing lapangan, studi kepustakaan, perencanaan kebutuhan, perancangan sistem, implementasi, serta penarikan kesimpulan dan saran. Berikut dipaparkan diagram alir metode penelitian pada Gambar 4.1.



Gambar 4. 1 Diagram Alir pengembangan Sistem Informasi Pencatatan COVID-19 (SIPCOP)

4.1 Diskusi dengan Pembimbing Lapangan

Tahap pertama yang dilakukan yaitu diskusi dengan pembimbing praktik kerja lapangan, dengan cara mengadakan rapat yang diadakan dengan kepala Dinas Komunikasi dan Informatika Kabupaten Gresik, Kepala Seksi Pengelolaan dan Pelayanan Informasi, Kepala Seksi Aplikasi dan Pengembangan Informatika, Subbagian Umum dan Kepegawaian, dan Pembimbing lapangan selama praktik kerja lapangan mengenai apa saja yang dilakukan pada hari kedua pelaksanaan Praktik Kerja Lapangan. Untuk setiap harinya juga dilakukan pelaporan progres dan kendala yang dihadapi untuk setiap masalah yang ada. Jika ada hal yang perlu ditanyakan, atau saran mengenai perancangan sistem yang akan dibuat maka akan dilakukan diskusi kecil dengan pembimbing lapangan selama jam kantor.

4.2 Studi Kepustakaan

Studi Kepustakaan merupakan tahap mengumpulkan seluruh informasi yang relevan dengan topik bahasan baik dari jurnal, buku, karya ilmiah, internet, dan sumber-sumber lain untuk dipelajari dan dipahami. Teori-teori tersebut nantinya akan dijadikan sebagai acuan dasar penelitian serta dapat mempermudah untuk menyelesaikan permasalahan yang akan dibahas dalam penelitian.

4.3 Analisis Kebutuhan

Dalam tahap analisis kebutuhan ini akan dilakukan identifikasi pengguna sistem sehingga dapat diketahui apa saja yang menjadi kebutuhan sistem yaitu dengan cara mengidentifikasi kebutuhan informasi, tujuan dari sistem, batasan-batasan sistem, masalah yang dihadapi untuk menentukan tujuan, dan juga alternatif pemecahan masalah. Dari hasil identifikasi, akan diambil hasil analisis yang akan digunakan untuk mengetahui perilaku sistem dan juga untuk mengetahui aktivitas apa saja yang ada dalam sistem tersebut. Tahap ini akan dilakukan perancangan kebutuhan fungsional, dan non-fungsional kemudian digambarkan dalam *use case diagram* dan *activity diagram* yang dibuat melalui Draw.io

4.4 Perancangan Sistem

Setelah semua kebutuhan dalam pengembangan telah didapatkan dari tahap analisis kebutuhan, selanjutnya adalah tahap perancangan sistem. Perancangan sistem terdiri dari beberapa tahap yaitu menggambarkan sistem dengan *sequence diagram*, *class diagram*, *User Interface*, dan *Perancangan Physical Data Model (PDM)*. Tools yang digunakan dalam menggambar diagram

yaitu Draw.io, Figma (untuk merancang User Interface), dan Power Designer untuk merancang *Physical Data Model* (PDM).

4.5 Implementasi

Pada tahap ini akan dimulai pembangunan sistem dengan menerapkan rancangan sistem yang telah dibuat sebelumnya. Pada tahap ini terdapat banyak aktivitas yang dilakukan, hasil desain dimasukkan ke dalam bentuk bahasa pemrograman yang digunakan agar dapat dijalankan. Implementasi pada Sistem Informasi *pencatatan COVID-19 (SIPCOP)* meliputi implementasi *user interface* menggunakan HTML, CSS dan Javascript dengan *framework* Bootstrap 4.0, implementasi database menggunakan MySQL, dan implementasi kode program menggunakan bahasa pemrograman PHP dengan *framework* CodeIgniter versi 3.0. Implementasi pada SIPCOP meliputi implementasi *User Interface*, implementasi database, dan implementasi kode program.

4.6 Pengujian Sistem

Tahap pengujian dilakukan untuk menguji apakah semua kebutuhan telah sesuai dengan permintaan user dan ditemukannya kesalahan pada sistem yang telah dibuat. Bentuk pengujian yang akan dilakukan adalah pengujian validasi dengan menggunakan *Black Box Testing*. Sedangkan untuk pengujian Compatibility testing menggunakan SortSite. Jika selama pengujian terdapat kesalahan maupun tidak berjalan sebagaimana mestinya maka akan dilakukan perbaikan ulang setelah itu dilakukan pengujian kembali.

4.7 Kesimpulan dan Saran

Setelah semua tahapan pada metodologi penelitian telah selesai maka akan diambil kesimpulan. Kesimpulan tersebut diambil dari hasil analisis dari sistem informasi yang telah selesai dibuat dan merupakan jawaban dari rumusan masalah yang sudah dirumuskan sebelumnya. Selanjutnya juga diberikan saran sebagai bahan masukan agar dapat memperbaiki kekurangan dari pengembangan sistem maupun cara penulisan bagi penelitian selanjutnya.

BAB 5 ANALISIS KEBUTUHAN

Dalam tahap analisis kebutuhan ini akan dilakukan identifikasi sehingga dapat diketahui apa saja yang menjadi kebutuhan sistem yaitu dengan cara mengidentifikasi kebutuhan informasi. Dari hasil identifikasi tersebut akan diambil hasil analisis untuk mengetahui perilaku sistem dan aktivitas apa saja yang ada pada sistem. Tahap ini akan dilakukan identifikasi aktor sesuai kebutuhan sistem, perencanaan kebutuhan fungsional dan non-fungsional kemudian digambarkan dalam *use case diagram* dan *activity diagram*.

Spesifikasi kebutuhan ditulis dengan format F_Aktor_Fungsi untuk mendefinisikan kebutuhan fungsional. Format NF_SIP_COMP mendefinisikan kebutuhan non fungsional. F adalah singkatan untuk Fungsional, Aktor yaitu ADM untuk admin dan PTGS untuk petugas. NF adalah singkatan untuk Non Fungsional, SIP adalah singkatan untuk Sistem Informasi Pencatatan.

5.1 Deskripsi Umum Sistem

Sistem Informasi Pencatatan *COVID-19* (SIPCOP) adalah sistem informasi yang berfungsi untuk manajemen data pasien *COVID-19* dan memonitoring kasus riwayat pasien. Sistem informasi ini akan digunakan pada Gelora Joko Samudro sebagai rumah sakit darurat pada Kabupaten Gresik.

User dalam Sistem Informasi Pencatatan *COVID-19* (SIPCOP) memiliki 2 kewenangan berbeda, yaitu Administrator (ADM) dan Petugas (PTGS). Administrator (ADM) adalah satu orang yang bertugas untuk manajemen kewenangan dari petugas pencatatan, mengelola data petugas serta melakukan update pada keterangan status *COVID* dan jenis ruang perawatan. Sedangkan petugas pencatatan adalah orang yang berwenang untuk mengelola dan memonitoring data pasien *COVID-19*. Selain itu petugas juga dapat melihat tracking riwayat pasien *COVID-19* dan memantau jumlah pasien dalam bentuk grafik harian dan bulanan.

5.2 Identifikasi Aktor

Karakteristik setiap aktor akan dipaparkan pada Tabel 5.1

Tabel 5. 1 Identifikasi Aktor

No.	Identifikasi Aktor	Karakteristik
-----	--------------------	---------------

1.	Administrator (ADM)	Orang yang dapat mengakses sistem dengan login terlebih dahulu agar dapat memiliki hak akses fitur yang ada pada sistem. Memiliki kewenangan untuk manajemen kewenangan dari petugas pencatatan, mengelola data petugas serta melakukan update pada keterangan status COVID dan jenis ruang perawatan.
2.	Petugas (PTGS)	Orang yang dapat mengakses sistem dengan login terlebih dahulu agar dapat memiliki hak akses fitur yang ada pada sistem. Memiliki kewenangan untuk mengelola dan memonitoring data pasien <i>COVID-19</i> yang masuk di Rumah Sakit Gelora Joko Samudro. Selain itu juga dapat melakukan tracking kasus <i>COVID-19</i> dari data pasien serta memantau grafik harian dan bulanan dari jumlah pasien yang terdaftar di Rumah Sakit tersebut.

5.3 Analisis Kebutuhan Fungsional

Analisis Kebutuhan fungsional bertujuan untuk menganalisis fungsi-fungsi apa saja yang harus dilakukan oleh sistem yang dikembangkan. Kebutuhan fungsional pada sistem ini dijelaskan pada Tabel 5.2

Tabel 5. 2 Kebutuhan Fungsional

No.	Kode Fungsi	Nama Fungsi	Deskripsi
1.	F_ADM_LOGIN	<i>Login</i> Admin	Admin dapat masuk ke sistem menggunakan <i>username</i> dan <i>password</i> yang telah dimiliki oleh 1 orang sebagai super admin.
2.	F_PTGS_LOGIN	<i>Login</i> petugas	Petugas dapat masuk ke sistem menggunakan <i>username</i> dan <i>password</i> yang telah terdaftar pada halaman web super admin
3.	F_ADM_INPUT	Input Data petugas	Admin dapat melakukan input data petugas baru di halaman admin pada sub menu petugas agar petugas dapat memiliki hak

			akses untuk membuka halaman sistem.
4.	F_ADM_VIEW	Melihat data petugas	Admin dapat melihat daftar petugas yang telah ditambahkan pada sub menu petugas
5.	F_ADM_EDIT	Edit data petugas	Admin dapat melakukan update data petugas (username dan password) yang ada dalam daftar petugas pada sub menu petugas
6.	F_ADM_DELETE	Delete petugas	Admin dapat menghapus data petugas yang ada dalam daftar petugas pada sub menu petugas
7.	F_ADM_STATUS	Mengelola daftar Status	Admin dapat mengelola daftar status <i>COVID-19</i> yang terdaftar sesuai dengan peraturan kesehatan
8.	F_ADM_RUANG	Mengelola daftar ruang rawat	Admin dapat mengelola daftar ruang rawat yang tersedia di Rumah Sakit
9.	F_PTGS_INPUT	Input data pasien	Petugas dapat melakukan input data pasien baru di halaman petugas pada sub menu pasien
10.	F_PTGS_VIEW	Melihat data pasien	Petugas dapat melihat data pasien yang telah terdaftar dengan menekan tombol 'Detail' pada sub menu pasien
11.	F_PTGS_EDIT	Edit data pasien	Petugas dapat melakukan update data pasien pada sub menu pasien
12.	F_PTGS_DELETE	Delete data pasien	Petugas dapat menghapus data pasien pada sub menu pasien dengan menekan tombol 'Keluar'
13.	F_PTGS_DASHBOARD	Melihat grafik harian dan bulanan	Petugas dapat melihat grafik jumlah pasien yang terdaftar berupa grafik batang untuk bulanan dan diagram lingkaran

			untuk harian pada halaman <i>dashboard</i> .
14.	F_PTGS_TRACKING	Tracking riwayat pasien	Petugas dapat melacak pasien yang telah terdaftar dengan memasukkan no KTP pasien
15.	F_PTGS_VIEWSTATUS	Melihat daftar status	Petugas dapat melihat daftar status COVID-19 pada halaman petugas sub menu pasien dengan menekan tombol 'Daftar Status'
16.	F_PTGS_EDITSTATUS	Edit status pasien	Petugas dapat mengubah status pasien ketika diperlukan dengan menekan tombol 'Ubah Status'

5.4 Analisis Kebutuhan Non Fungsional

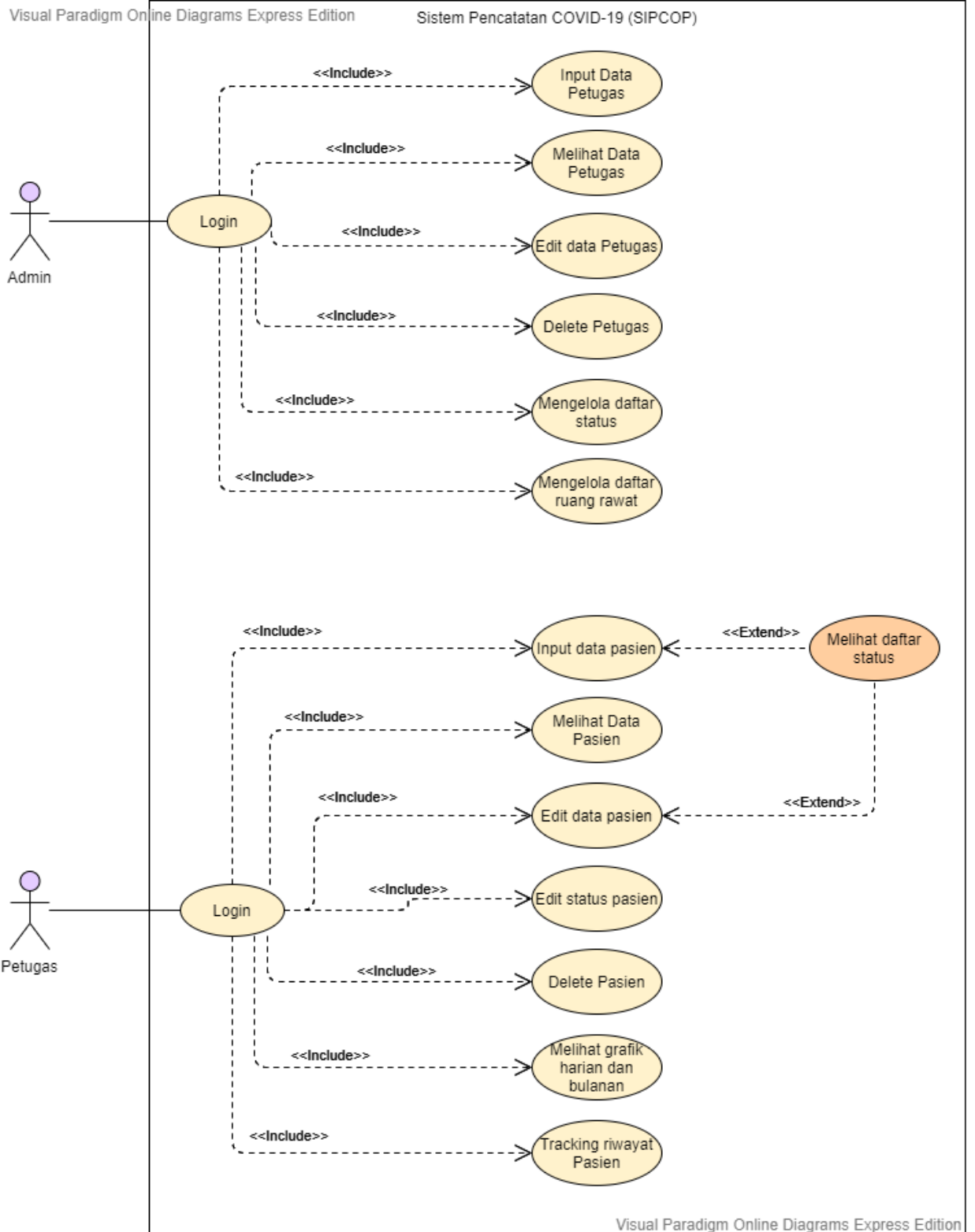
Kebutuhan Non Fungsional adalah kebutuhan yang menitikberatkan pada properti perilaku yang dimiliki oleh sistem. Kebutuhan non fungsional pada sistem ini dijelaskan pada Tabel 5.3

Tabel 5. 3 Kebutuhan Non Fungsional

No.	Kode Fungsi	Nama Fungsi	Deskripsi
1.	NF_SIP_COMP	<i>Compatibility Browser</i>	Sistem dapat berjalan pada aplikasi <i>web browser</i> yang berbeda

5.5 Use Case Diagram

Use case diagram merupakan gambaran scenario dari interaksi antara pengguna dengan sistem. Didalam *use case* ini akan diketahui fungsi-fungsi apa saja yang berada pada sistem yang dibuat (Maulana and Rachmawati 2017). Diagram Use Case dari Sistem Informasi Pencatatan COVID-19 (SIPCOP) digambarkan seperti pada Gambar 5.1.



Gambar 5. 1 Use Case Diagram Sistem Informasi Pencatatan COVID-19 (SIPCOP)

5.6 Skenario Use Case

Pada bagian ini terdapat tabel-tabel yang menjelaskan skenario dari setiap kebutuhan yang telah dipaparkan pada analisis kebutuhan fungsional dan non fungsional.

a. Admin

Tabel 5. 4 Skenario Use Case Login Admin

Nama Use Case	<i>Login Admin</i>
Kode Kebutuhan	F_ADM_LOGIN
Aktor	Admin
Tujuan	Aktor masuk ke sistem sebagai Super Admin menggunakan <i>username</i> dan <i>password</i> yang telah dimiliki oleh 1 orang sebagai super admin.
Precondition	Aktor telah berada di halaman <i>login</i> Superadmin
Main Flow	<ol style="list-style-type: none">1. Sistem menampilkan halaman <i>login</i>2. Sistem menampilkan form <i>login</i> berupa <i>username</i> dan <i>password</i>3. Aktor memasukkan <i>username</i> dan <i>password</i>4. Aktor menekan tombol 'Login'5. Sistem mengecek <i>username</i> dan <i>password</i> yang dimasukkan oleh aktor6. Sistem menampilkan halaman <i>dashboard</i>
Postcondition	Aktor berhasil masuk ke dalam sistem dan tampil halaman beranda
Alternative Flow	<ol style="list-style-type: none">1. Jika aktor memasukkan <i>username</i> atau <i>password</i> yang salah atau tidak terdaftar dalam sistem, maka sistem akan mengarahkan aktor pada halaman <i>login</i> kembali2. Jika aktor hanya mengisi salah satu form diantara <i>username</i> dan <i>password</i>, maka sistem akan memberikan peringatan pada form yang kosong

Tabel 5. 5 Skenario Use Case Input data petugas

Nama Use Case	Input Data Petugas
Kode Kebutuhan	F_ADM_INPUT
Aktor	Admin
Tujuan	Menginput data petugas baru oleh aktor pada halaman input data petugas
Precondition	Aktor telah berada di halaman admin
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman admin pada side bar input petugas 2. Sistem menampilkan daftar petugas yang telah terdaftar 3. Aktor menekan tombol 'Tambah' 4. Sistem menampilkan form tambah petugas 5. Aktor memasukkan data petugas pada form yang tersedia 6. Aktor menekan tombol 'Simpan' 7. Sistem menampilkan semua daftar petugas
Postcondition	Aktor berhasil melakukan input data pada petugas baru pada halaman input petugas
Alternative Flow	<ol style="list-style-type: none"> 1. Jika aktor tidak melakukan input <i>username</i> atau <i>password</i> baru pada form tambah petugas, maka sistem akan memberikan peringatan untuk mengisi form yang kosong.

Tabel 5. 6 Skenario Use Case melihat data petugas

Nama Use Case	Melihat data petugas
Kode Kebutuhan	F_ADM_VIEW
Aktor	Admin
Tujuan	Aktor dapat melihat daftar petugas yang sudah ditambahkan maupun yang baru saja ditambahkan

Precondition	Aktor telah berada di halaman admin pada <i>sidebar</i> petugas
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman admin pada <i>sidebar</i> petugas 2. Sistem menampilkan daftar petugas yang baru saja ditambahkan maupun yang sudah ditambahkan sebelumnya
Postcondition	Aktor berhasil melihat daftar petugas
Alternative Flow	-

Tabel 5. 7 Skenario Use Case Edit data petugas

Nama Use Case	Edit data petugas
Kode Kebutuhan	F_ADM_EDIT
Aktor	Admin
Tujuan	Aktor dapat mengedit data petugas pada halaman admin <i>sidebar</i> petugas
Precondition	Aktor telah berada di halaman admin pada <i>sidebar</i> petugas
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman admin pada <i>sidebar</i> petugas 2. Sistem menampilkan daftar petugas 3. Aktor menekan tombol 'Ubah' pada petugas yang ingin diubah datanya 4. Sistem menampilkan form ubah petugas 5. Aktor melakukan edit data petugas pada form yang ingin diubah 6. Aktor menekan tombol 'Simpan' 7. Sistem menampilkan daftar petugas dengan data petugas yang baru saja diedit.
Postcondition	Aktor berhasil melakukan edit data petugas pada halaman admin <i>sidebar</i> petugas

Alternative Flow	1. Jika aktor tidak memasukkan <i>username</i> atau <i>password</i> pada form Ubah petugas, maka sistem akan memberikan peringatan untuk mengisi form yang kosong.
-------------------------	--

Tabel 5. 8 Skenario Use Case Delete Petugas

Nama Use Case	Delete Petugas
Kode Kebutuhan	F_ADM_DELETE
Aktor	Admin
Tujuan	Aktor dapat melakukan hapus data petugas
Precondition	Aktor telah berada di halaman admin pada <i>sidebar</i> petugas
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman admin pada <i>sidebar</i> petugas 2. Sistem menampilkan daftar petugas 3. Aktor menekan tombol 'Hapus' 4. Sistem akan memproses hapus data petugas
Postcondition	Aktor berhasil menghapus data petugas pada halaman admin
Alternative Flow	-

Tabel 5. 9 Skenario Use Case mengelola daftar status

Nama Use Case	Mengelola daftar status
Kode Kebutuhan	F_ADM_STATUS
Aktor	Admin
Tujuan	Aktor dapat mengelola daftar status pasien
Precondition	Aktor telah berada di halaman super admin pada sub menu status pasien

Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu Status Pasien 4. Aktor menekan tombol 'Tambah' untuk menginput daftar status yang ingin ditambahkan 5. Aktor memasukkan Nama Status 6. Aktor memasukkan Keterangan dari Status tersebut 7. Aktor menekan tombol 'Simpan' 8. Sistem menampilkan Pemberitahuan bahwa penambahan data status pasien berhasil ditambahkan 9. Sistem menampilkan daftar Status Pasien yang terdaftar sesuai dengan peraturan kesehatan 10. Jika ingin melakukan perubahan, Aktor dapat menekan tombol 'Edit' 11. Aktor melakukan perubahan nama maupun keterangan status 12. Aktor menekan tombol 'Simpan' 13. Sistem menampilkan pemberitahuan bahwa perubahan data status pasien berhasil 14. Jika ingin menghapus, Aktor dapat menekan tombol 'Delete' untuk menghapus status yang dipilih dari daftar 15. Sistem akan menampilkan konfirmasi untuk hapus kepada aktor berupa <i>pop up</i> 16. Aktor menekan tombol 'Oke' jika data tersebut yakin dihapus 17. Sistem menampilkan pemberitahuan bahwa data status pasien yang dipilih berhasil terhapus
Postcondition	Aktor berhasil mengelola daftar status pasien pada sub menu status
Alternative Flow	<ol style="list-style-type: none"> 1. Jika aktor tidak menginputkan nama status saat melakukan input maupun edit, maka sistem akan menampilkan pesan peringatan untuk melakukan input nama status. 2. Jika aktor menekan tombol 'Batal' pada <i>pop up</i> saat melakukan penghapusan, maka data tidak jadi terhapus dari daftar status

Tabel 5. 10 Skenario Use Case mengelola daftar ruang rawat

Nama Use Case	Mengelola daftar ruang rawat
Kode Kebutuhan	F_ADM_RUANG

Aktor	Admin
Tujuan	Aktor dapat mengelola daftar ruang rawat
Precondition	Aktor telah berada di halaman super admin pada sub menu Ruang Rawat
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu Ruang Rawat 4. Aktor menekan tombol 'Tambah' untuk menginput nama ruang rawat yang ingin ditambahkan 5. Aktor memasukkan nama Zona Ruang 6. Aktor memasukkan Keterangan dari ruang tersebut 7. Aktor menekan tombol 'Simpan' 8. Sistem menampilkan Pemberitahuan bahwa penambahan data ruang rawat berhasil ditambahkan 9. Sistem menampilkan daftar Ruang Rawat yang digunakan pada rumah sakit darurat 10. Jika ingin melakukan perubahan, Aktor dapat menekan tombol 'Edit' 11. Aktor melakukan perubahan nama maupun keterangan ruang rawat 12. Aktor menekan tombol 'Simpan' 13. Sistem menampilkan Pemberitahuan bahwa perubahan data ruang rawat berhasil 14. Jika ingin menghapus, Aktor dapat menekan tombol 'Delete' untuk menghapus ruang rawat yang dipilih dari daftar 15. Sistem akan menampilkan konfirmasi kepada aktor berupa <i>pop up</i> 16. Aktor menekan tombol 'Oke' pada <i>pop up</i> 17. Sistem menampilkan Pemberitahuan bahwa ruang rawat yang dipilih berhasil terhapus
Postcondition	Aktor berhasil mengelola daftar ruang rawat
Alternative Flow	<ol style="list-style-type: none"> 1. Jika aktor tidak menginputkan nama zona ruang saat melakukan input maupun edit, maka sistem akan menampilkan pesan peringatan untuk melakukan input nama zona ruang. 2. Jika aktor menekan tombol 'Batal' pada <i>pop up</i> saat melakukan penghapusan, maka data tidak jadi terhapus dari daftar ruang rawat.

b. Petugas

Tabel 5. 11 Skenario Use Case Login petugas

Nama Use Case	<i>Login</i> petugas
Kode Kebutuhan	F_PTGS_INPUT
Aktor	Petugas
Tujuan	Aktor dapat masuk ke sistem menggunakan <i>username</i> dan <i>password</i> yang telah terdaftar pada halaman web super admin
Precondition	Aktor telah berada di halaman <i>login</i> Petugas
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>login</i> 2. Sistem menampilkan form <i>login</i> berupa <i>username</i> dan <i>password</i> 3. Aktor memasukkan <i>username</i> dan <i>password</i> 4. Aktor menekan tombol 'Login' 5. Sistem mengecek <i>username</i> dan <i>password</i> yang dimasukkan oleh aktor 6. Sistem menampilkan halaman <i>dashboard</i>
Postcondition	Aktor berhasil masuk ke dalam sistem dan tampil halaman dashboard
Alternative Flow	<ol style="list-style-type: none"> 1. Jika aktor memasukkan <i>username</i> atau <i>password</i> yang salah atau tidak terdaftar dalam sistem, maka sistem akan mengarahkan aktor pada halaman <i>login</i> kembali 2. Jika aktor hanya mengisi salah satu form diantara <i>username</i> dan <i>password</i>, maka sistem akan memberikan peringatan pada form yang kosong

Tabel 5. 12 Skenario Use Case Input data pasien

Nama Use Case	Input data pasien
Kode Kebutuhan	F_PTGS_INPUT

Aktor	Petugas
Tujuan	Menginput data pasien baru oleh aktor pada halaman petugas sub menu pasien
Precondition	Aktor telah berada di halaman petugas sub menu pasien
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman petugas pada sub menu pasien 2. Sistem menampilkan data pasien yang telah terdaftar 3. Aktor menekan tombol 'Tambah' 4. Sistem menampilkan form tambah pasien 5. Aktor memasukkan data pasien pada form yang tersedia 6. Aktor menekan tombol 'Simpan' 7. Sistem menampilkan semua daftar pasien
Postcondition	Aktor berhasil melakukan input data pada pasien baru pada halaman pasien
Alternative Flow	<ol style="list-style-type: none"> 1. Jika aktor tidak memasukkan salah satu kolom yang wajib diisi pada form tambah pasien, maka sistem akan memberikan peringatan untuk mengisi form yang kosong.

Tabel 5. 13 Skenario Use Case melihat data pasien

Nama Use Case	Melihat data pasien
Kode Kebutuhan	F_PTGS_VIEW
Aktor	Petugas
Tujuan	Aktor dapat melihat daftar pasien yang baru saja ditambahkan maupun yang sudah ditambahkan sebelumnya
Precondition	Aktor telah berada di halaman petugas pada sub menu pasien

Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman petugas pada sub menu pasien 2. Sistem menampilkan daftar pasien yang baru saja ditambahkan maupun yang sudah ditambahkan sebelumnya
Postcondition	Aktor berhasil melihat daftar pasien pada sub menu pasien
Alternative Flow	-

Tabel 5. 14 Skenario Use Case edit data pasien

Nama Use Case	Edit data pasien
Kode Kebutuhan	F_PTGS_EDIT
Aktor	Petugas
Tujuan	Aktor dapat mengedit data pasien pada halaman petugas sub menu pasien
Precondition	Aktor telah berada di halaman petugas pada sub menu pasien
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan daftar pasien 2. Aktor menekan tombol 'Ubah' pada pasien yang akan diubah maupun update datanya. 3. Sistem menampilkan form ubah pasien 4. Aktor melakukan edit data pasien pada form yang ingin diubah maupun <i>update</i> 5. Aktor menekan tombol 'Simpan' 6. Sistem menampilkan daftar petugas yang baru saja diedit.
Postcondition	Aktor berhasil melakukan edit data pasien pada halaman petugas sub menu pasien
Alternative Flow	Jika aktor tidak memasukkan salah satu kolom yang wajib diisi pada form tambah pasien, maka sistem akan memberikan peringatan untuk mengisi form yang kosong.

Tabel 5. 15 Skenario Use Case delete pasien

Nama Use Case	Delete pasien
Kode Kebutuhan	F_PTGS_DELETE
Aktor	Petugas
Tujuan	Aktor dapat melakukan hapus data pasien
Precondition	Aktor telah berada di halaman petugas pada submenu pasien
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan daftar pasien 2. Aktor dapat menghapus data pasien dengan menekan tombol 'Keluar' 3. Sistem akan menampilkan daftar pasien yang masih berada di Rumah Sakit
Postcondition	Aktor berhasil menghapus data pasien pada halaman petugas sub menu pasien
Alternative Flow	-

Tabel 5. 16 Skenario Use Case Melihat Grafik harian dan bulanan

Nama Use Case	Melihat grafik harian dan bulanan
Kode Kebutuhan	F_PTGS_DASHBOARD
Aktor	Petugas
Tujuan	Aktor dapat melihat grafik harian dan bulanan pada halaman petugas
Precondition	Aktor telah berada di halaman petugas pada sub menu <i>dashboard</i>

Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan grafik dalam 1 hari dan grafik dalam 1 tahun terakhir yang dilihat pada setiap bulan
Postcondition	Aktor berhasil melihat grafik harian dan bulanan pada halaman petugas
Alternative Flow	-

Tabel 5. 17 Skenario Use Case Tracking riwayat pasien

Nama Use Case	Tracking riwayat Pasien
Kode Kebutuhan	F_PTGS_TRACKING
Aktor	Petugas
Tujuan	Aktor dapat melakukan tracking riwayat pasien
Precondition	Aktor telah berada di halaman petugas pada sub menu tracking riwayat pasien
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan sub menu tracking riwayat pasien 2. Aktor melakukan input No. KTP pada kolom yang tersedia 3. Aktor menekan tombol 'Cari' 4. Sistem menampilkan data pasien yang nomor KTP nya telah diinputkan
Postcondition	Aktor berhasil melakukan tracking riwayat pasien
Alternative Flow	<ol style="list-style-type: none"> 1. Jika nomor KTP yang diinputkan tidak terdaftar pada sistem maka sistem tidak menampilkan data apapun.

Tabel 5. 18 Skenario Use Case Melihat daftar status

Nama Use Case	Melihat daftar status
Kode Kebutuhan	F_PTGS_VIEWSTATUS

Aktor	Petugas
Tujuan	Aktor dapat melihat daftar status beserta keterangannya
Precondition	Aktor telah berada di halaman petugas pada sub menu pasien
Main Flow	<ol style="list-style-type: none"> 1. Aktor menampilkan halaman pada sub menu pasien 2. Aktor menekan tombol 'Daftar Status' 3. Sistem akan menampilkan daftar status COVID beserta keterangannya
Postcondition	Aktor berhasil menampilkan daftar status beserta keterangannya
Alternative Flow	-

Tabel 5. 19 Skenario Use Case edit status pasien

Nama Use Case	Edit status pasien
Kode Kebutuhan	F_PTGS_EDITSTATUS
Aktor	Petugas
Tujuan	Aktor dapat melakukan edit status pasien
Precondition	Aktor telah berada di halaman petugas pada sub menu pasien
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan sub menu pasien 2. Aktor menekan tombol 'Ubah Status' yang terletak dibawah status pasien saat ini pada kolom status 3. Aktor menekan tombol 'Simpan' ketika sudah melakukan perubahan
Postcondition	Aktor berhasil melakukan edit pada status pasien

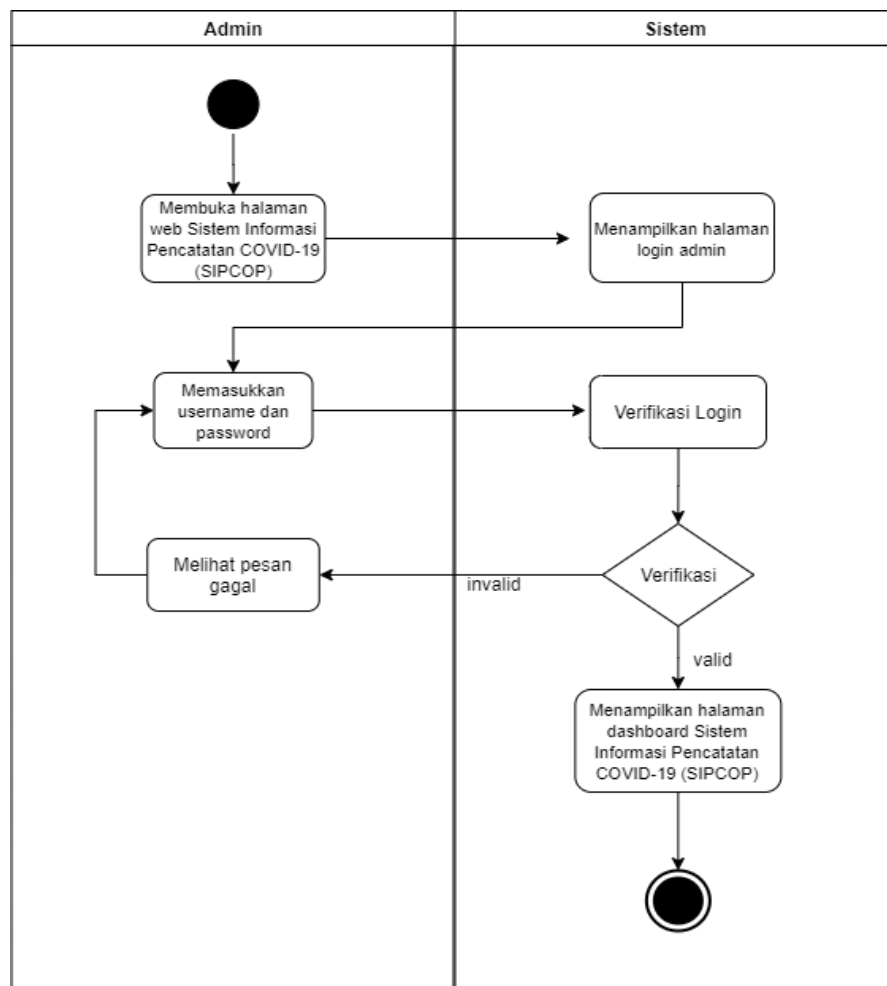
Alternative Flow	1. Jika aktor melakukan edit status pasien “Selesai isolasi” maka aktor tidak dapat lagi melakukan perubahan pada status pasien.
-------------------------	--

5.7 Activity Diagram

Activity Diagram merupakan diagram yang menjelaskan tentang alir kegiatan dalam program yang sedang dirancang, bagaimana proses alir berawal, keputusan yang mungkin terjadi, dan bagaimana sistem akan berakhir. Berikut adalah *activity diagram* dari Sistem Informasi Pencatatan COVID-19 (SIPCOP).

5.7.1 Activity Diagram Login Admin

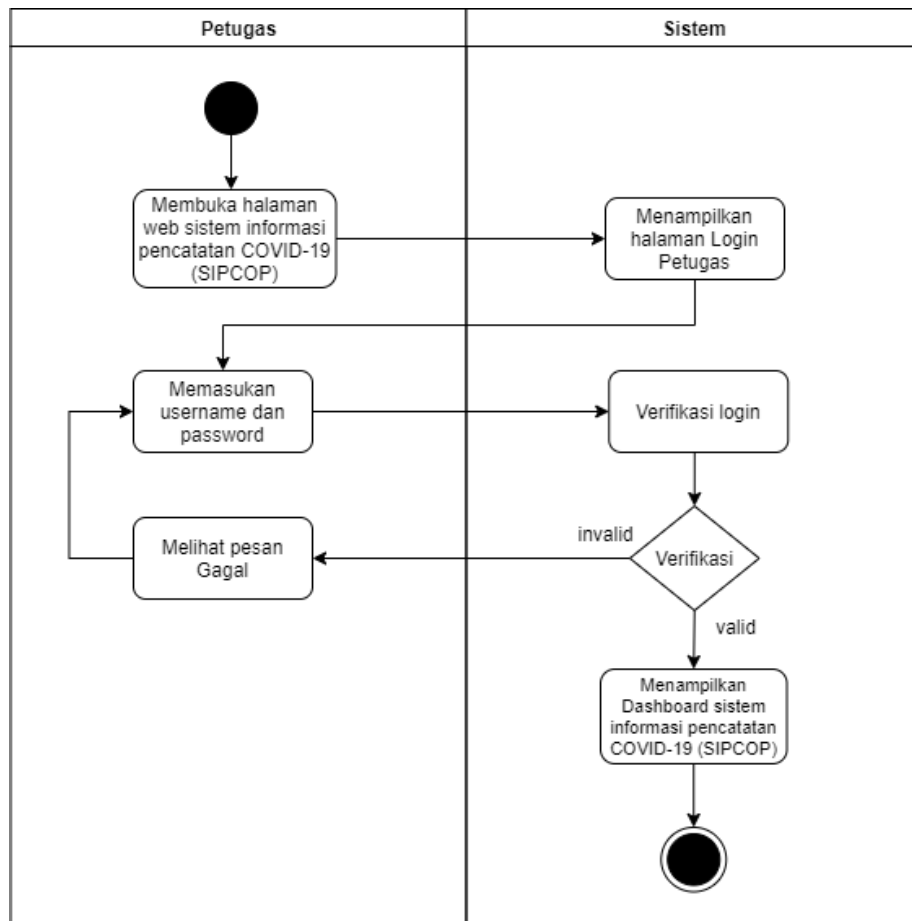
Berikut adalah *Activity Diagram* Login yang akan dijelaskan pada gambar 5.2. *User* melakukan *login* dengan menggunakan *username* dan *password* yang telah dimiliki oleh 1 orang sebagai super admin. Jika *username* dan *password* sesuai, *user* akan menemui halaman *dashboard* daftar petugas. Jika tidak sesuai, *user* akan kembali menemui halaman *login* admin.



Gambar 5. 2 Activity Diagram Login Admin

5.7.2 Activity Diagram Login Petugas

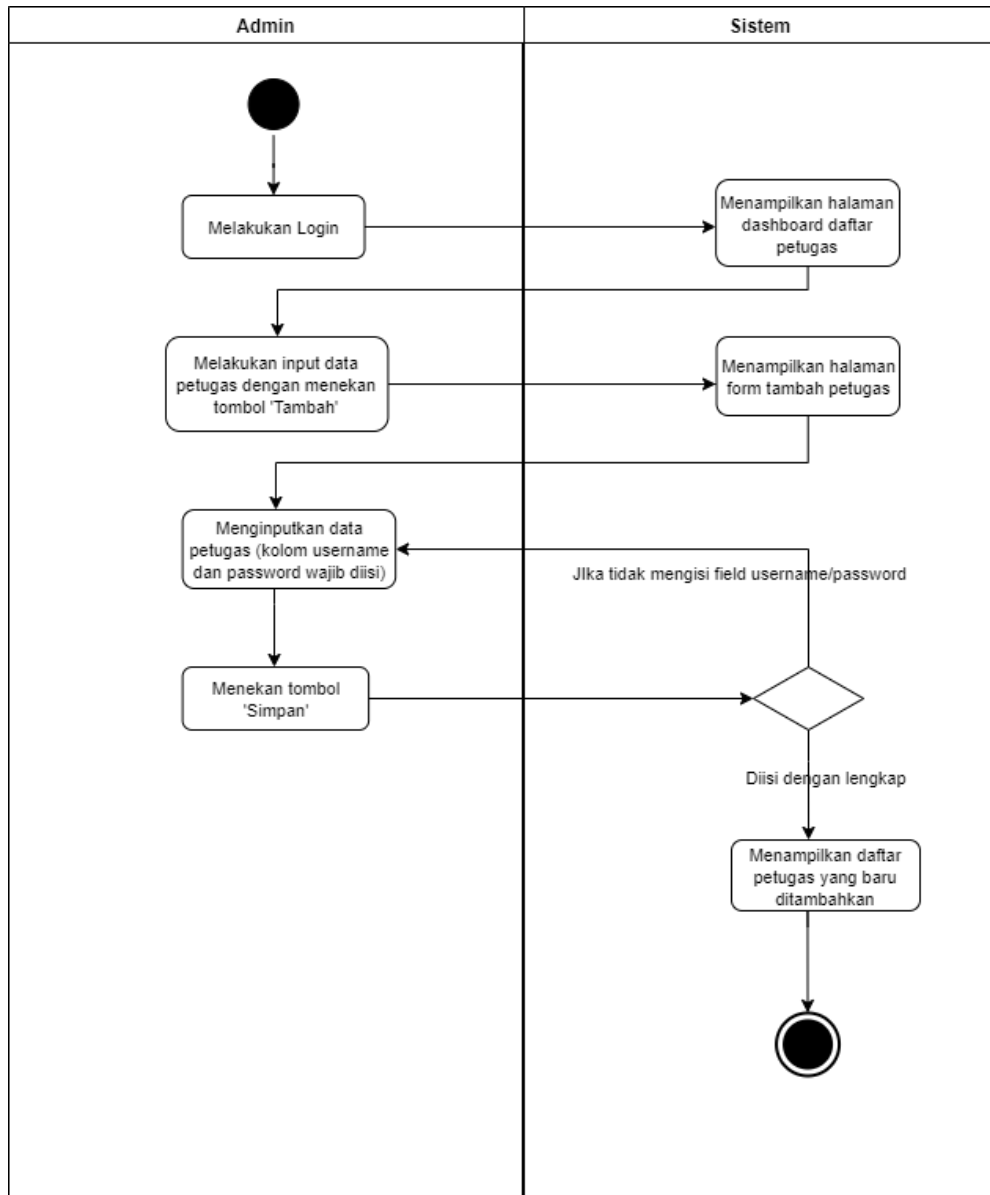
Berikut adalah *Activity Diagram Login Petugas* yang akan dijelaskan pada gambar 5.3. *User* melakukan *login* dengan menggunakan *username* dan *password* yang telah terdaftar pada halaman web admin. Jika *username* dan *password* sesuai, maka petugas dapat memasuki sistem yang menampilkan halaman dashboard Sistem Informasi pencatatan *COVID-19* (SIPCOP). Jika tidak sesuai, sistem akan menampilkan pesan gagal dan meminta petugas untuk memeriksa kembali agar menginputkan *username* dan *password* yang benar.



Gambar 5. 3 Activity Diagram Login petugas

5.7.3 Activity Diagram Input Data Petugas

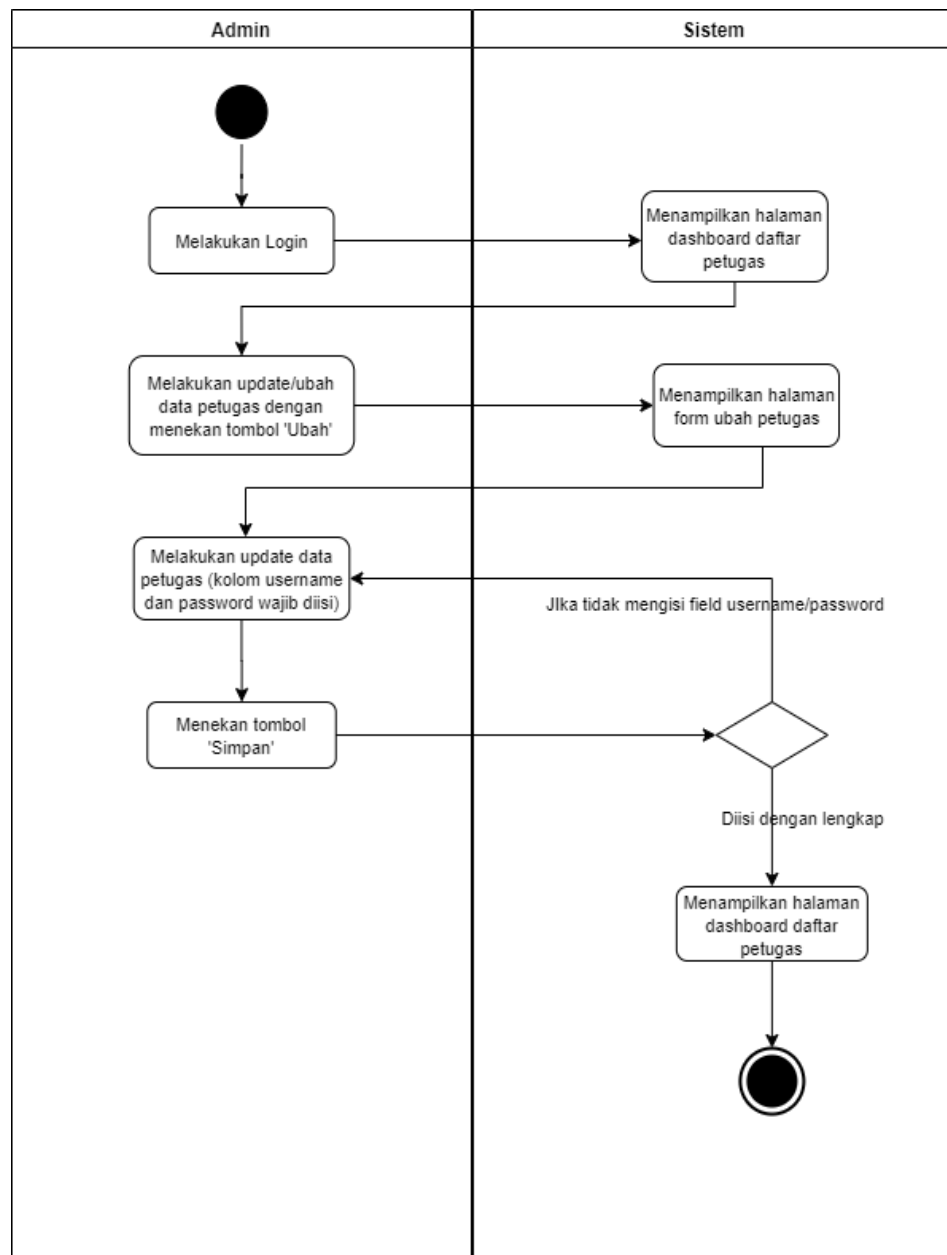
Berikut adalah *Activity Diagram Input Data Petugas* yang akan dijelaskan pada gambar 5.4. *User* terlebih dahulu melakukan *login*, kemudian sistem akan menampilkan halaman dashboard admin dimana halaman tersebut adalah halaman untuk menginputkan data petugas. Selanjutnya user akan menginputkan data petugas dengan menekan tombol 'Tambah'. Setelah ditampilkan form untuk input data petugas, user harus mengisi username dan password, jika salah satu kolom tidak diisi maka akan muncul pesan gagal simpan. Jika semua sudah diisi maka akan muncul pesan berhasil simpan.



Gambar 5. 4 Activity Diagram Input data petugas

5.7.4 Activity Diagram Edit Data Petugas

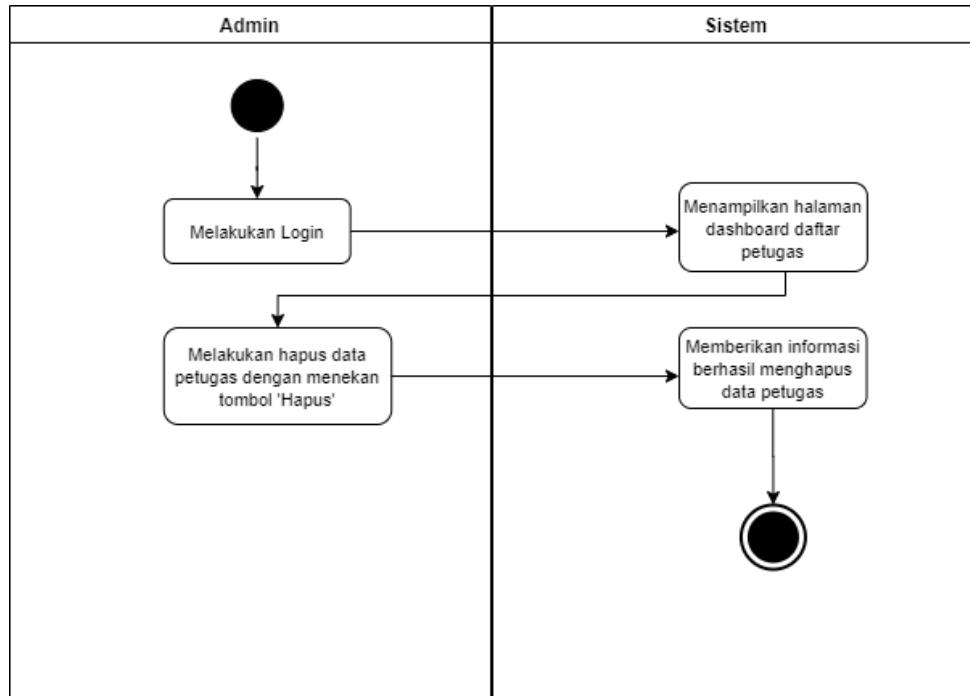
Berikut adalah *Activity Diagram* Edit data petugas yang akan dijelaskan pada gambar 5.5. *User* terlebih dahulu melakukan login, kemudian sistem akan menampilkan halaman dashboard daftar petugas. Selanjutnya user akan melakukan update data dengan menekan tombol 'Ubah'. Setelah ditampilkan halaman form ubah petugas, user dapat melakukan perubahan pada data petugas, dan diwajibkan untuk mengisi username dan password. Jika salah satu kolom tidak diisi maka akan muncul pesan gagal simpan.



Gambar 5. 5 Activity Diagram edit data petugas

5.7.5 Activity Diagram Hapus data Petugas

Berikut adalah *Activity Diagram* Hapus data Petugas yang akan dijelaskan pada gambar 5.6. *User* terlebih dahulu melakukan *login*, kemudian sistem akan menampilkan halaman dashboard daftar petugas. Selanjutnya user menekan tombol 'Hapus' untuk menghapus data petugas yang ada pada halaman dashboard. Jika berhasil menghapus sistem akan menampilkan informasi berhasil menghapus data petugas.

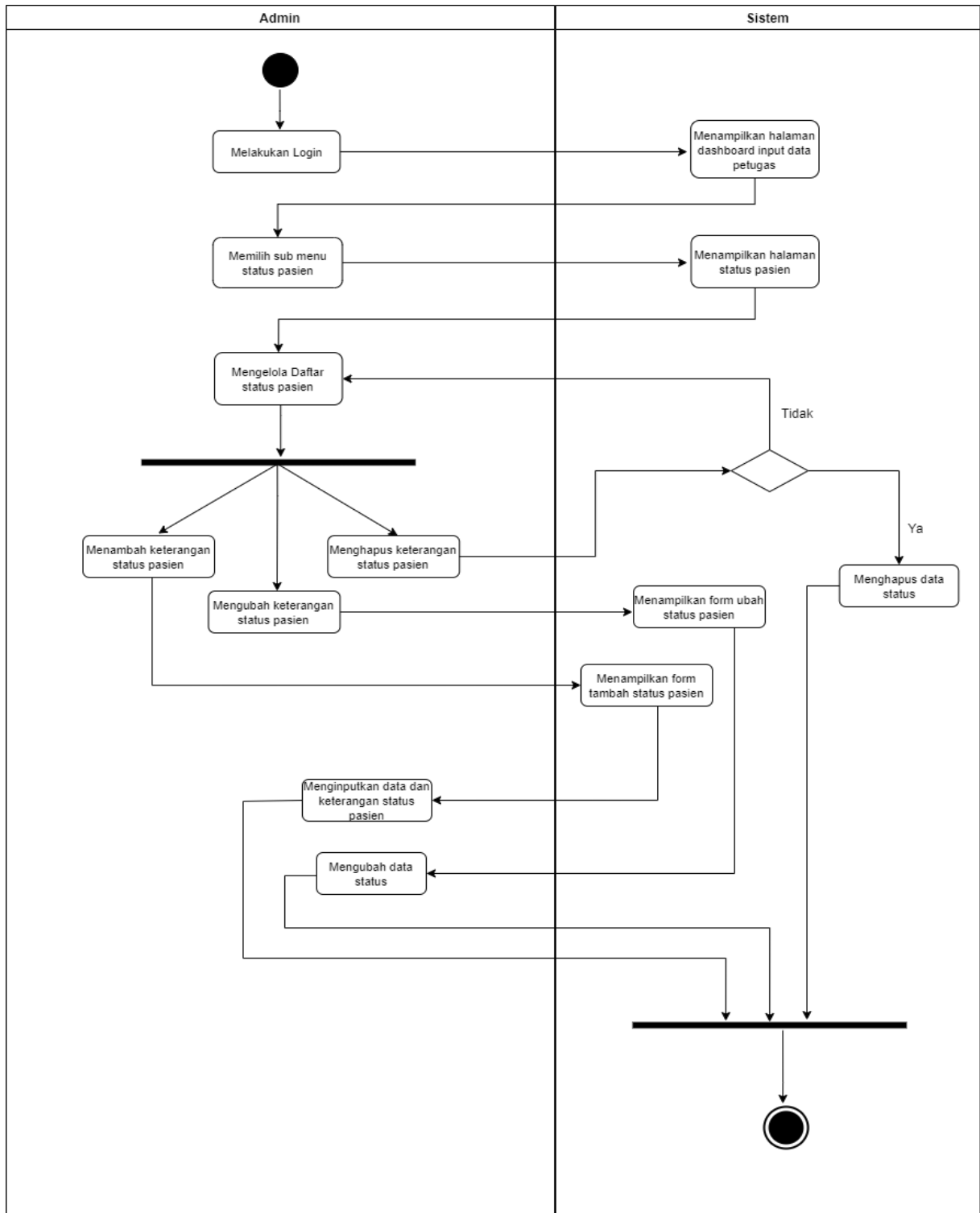


Gambar 5. 6 Activity Diagram hapus data petugas

5.7.6 Activity Diagram Mengelola daftar status

Berikut adalah *Activity Diagram* mengelola daftar status yang akan dijelaskan pada gambar 5.7. *User* terlebih dahulu melakukan *login*, kemudian sistem akan menampilkan halaman dashboard daftar petugas. Selanjutnya user memilih sub menu status pasien dan sistem akan menampilkan halaman status pasien. User dapat melakukan 3 aktivitas yaitu menambah atau membuat keterangan status pasien, mengubah keterangan dan menghapus keterangan status pasien. Aktivitas pertama yaitu tambah status, user dapat menekan tombol 'Tambah' dan nantinya akan ditampilkan form untuk tambah keterangan status. Aktivitas kedua yaitu Ubah status, user dapat menekan tombol 'Ubah' dan nantinya akan ditampilkan form untuk mengubah data keterangan status. Aktivitas ketiga yaitu hapus data, dimana ketika user menekan tombol 'Hapus' maka akan diberikan pilihan "Ya" atau "Tidak". Jika tidak yakin untuk menghapus maka user akan dikembalikan ke halaman status.

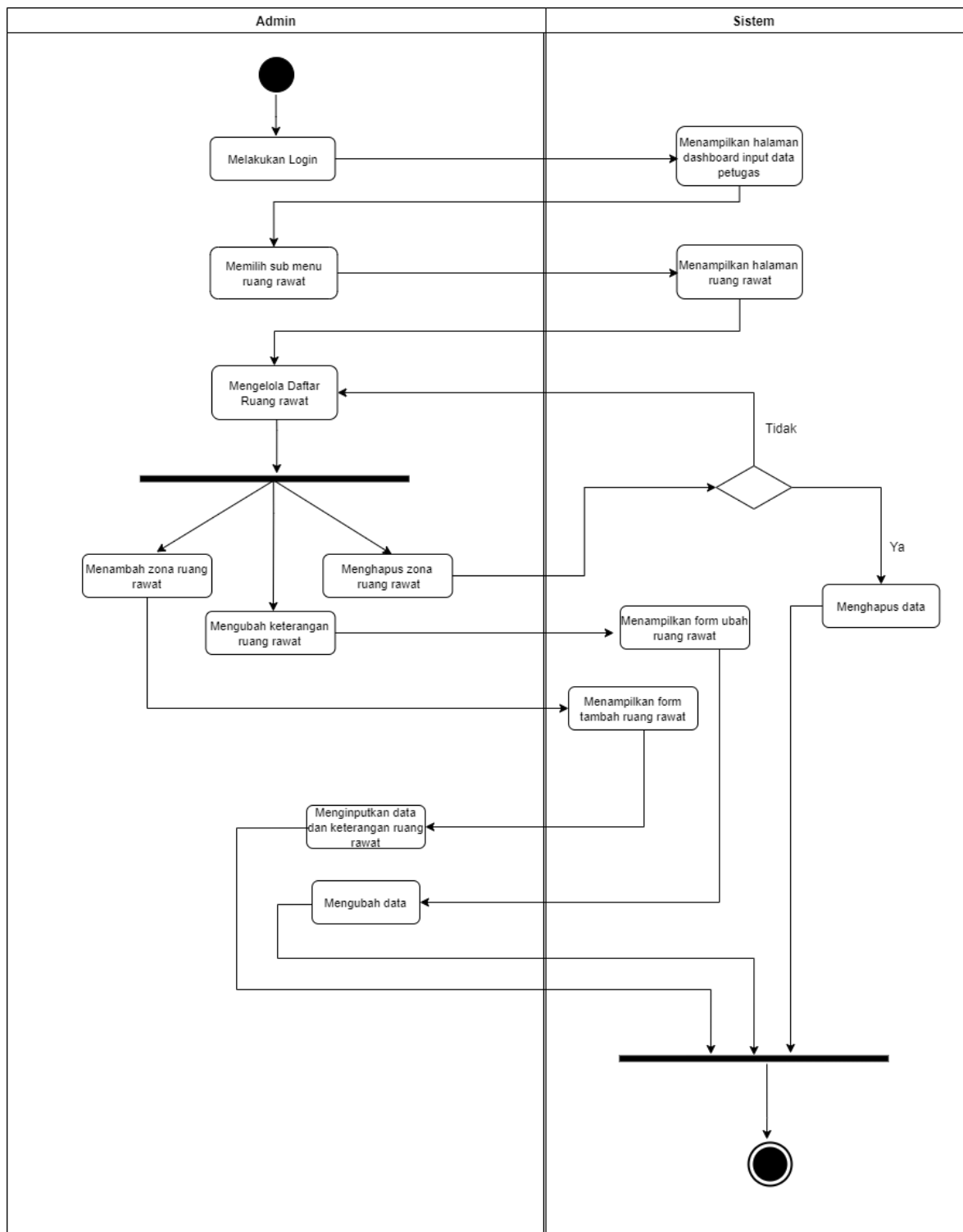
Jika yakin untuk menghapus maka data akan langsung terhapus dan proses selesai.



Gambar 5. 7 Activity Diagram mengelola daftar status

5.7.7 Activity Diagram Mengelola Daftar Ruang Rawat

Berikut adalah *Activity Diagram Approval Prospect* yang akan dijelaskan pada gambar 5.8. *User* terlebih dahulu melakukan *login*, kemudian sistem akan menampilkan halaman dashboard petugas. Selanjutnya *user* memilih sub menu ruang rawat dan sistem akan menampilkan halaman ruang rawat. *User* dapat melakukan 3 aktivitas yaitu menambah atau membuat keterangan ruang rawat, mengubah keterangan dan menghapus keterangan ruang rawat. Aktivitas pertama yaitu tambah ruang rawat, *user* dapat menekan tombol 'Tambah' dan nantinya akan ditampilkan form untuk tambah ruang rawat. Aktivitas kedua yaitu Ubah status, *user* dapat menekan tombol 'Ubah' dan nantinya akan ditampilkan form untuk mengubah keterangan ruang rawat. Aktivitas ketiga yaitu hapus data, dimana ketika *user* menekan tombol 'Hapus' maka akan diberikan pilihan "Ya" atau "Tidak". Jika tidak yakin untuk menghapus maka *user* akan dikembalikan ke halaman ruang rawat. Jika yakin untuk menghapus maka data akan langsung terhapus dan proses selesai.

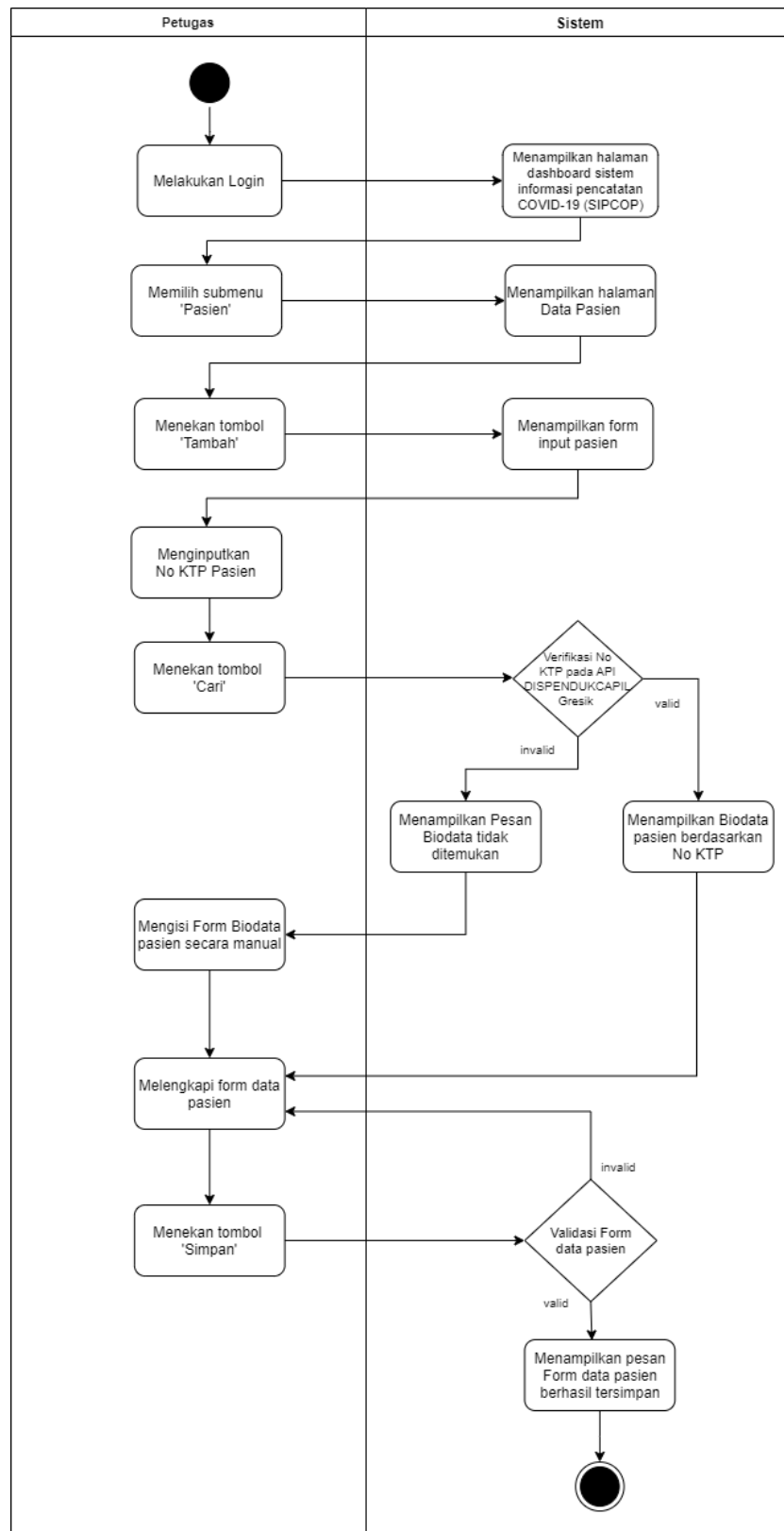


Gambar 5. 8 Activity Diagram mengelola daftar ruang rawat

5.7.8 Activity Diagram Input data pasien

Berikut adalah Activity Diagram menginputkan data pasien yang akan dijelaskan pada gambar 5.9. Petugas pencatatan harus melakukan *login* terlebih dahulu, kemudian petugas dapat menginputkan data pasien, disini petugas dapat memasukkan nomor KTP kemudian menekan tombol cari untuk mendapatkan data terisi secara otomatis dari API DISPENDUKAPIL

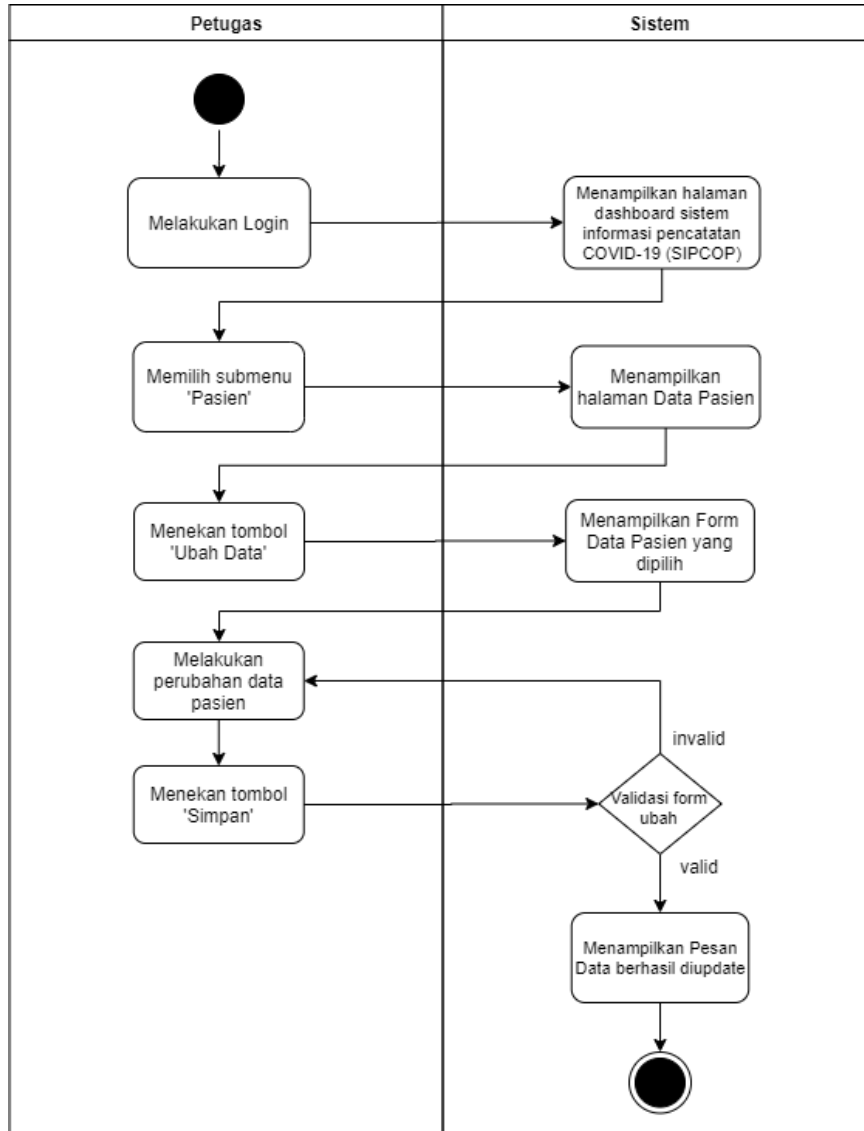
Kabupaten Gresik, namun jika nomor KTP tidak ditemukan, maka petugas dapat menginputkan data secara manual, kemudian melengkapi data pasien dan menyimpan data pasien tersebut.



Gambar 5. 9 Activity Diagram input data pasien

5.7.9 Activity Diagram Edit data pasien

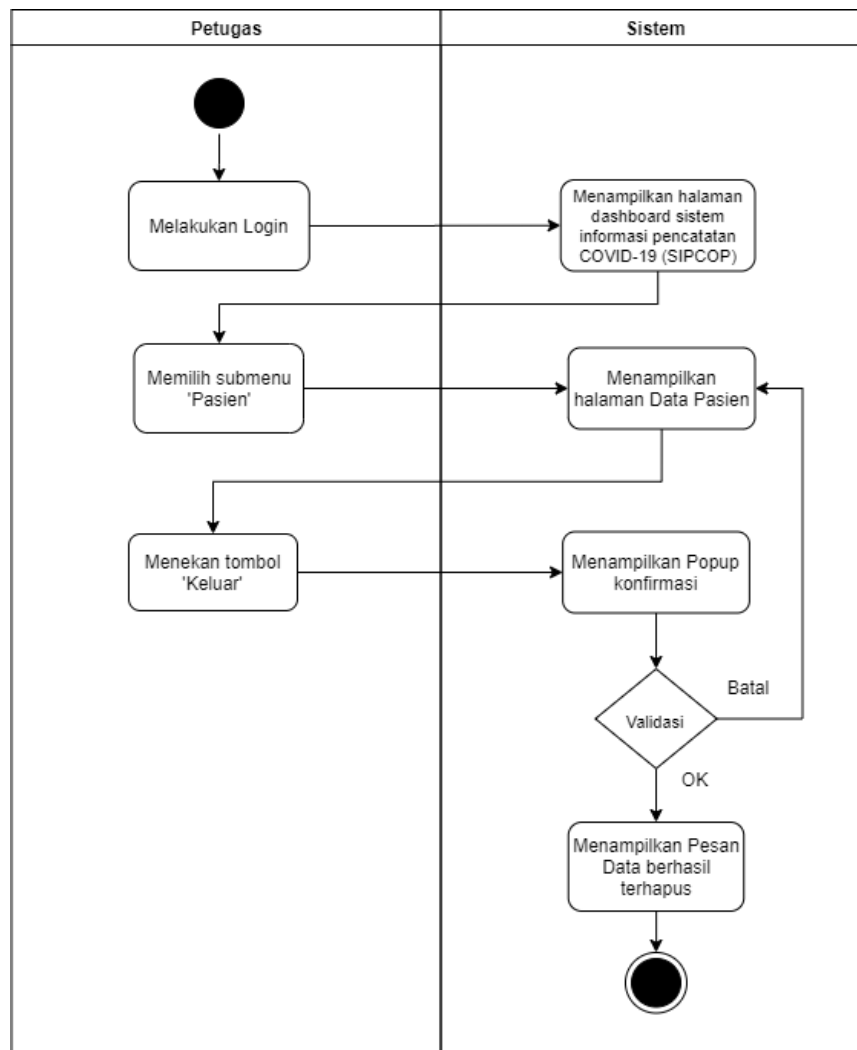
Berikut adalah Activity Diagram mengedit data pasien yang akan dijelaskan pada gambar 5.10. Petugas pencatatan harus melakukan *login* terlebih dahulu, kemudian petugas dapat mengedit data pasien dengan menekan ubah data, kemudian disimpan untuk memperbarui data pasien tersebut.



Gambar 5. 10 Activity Diagram edit data pasien

5.7.10 Activity Diagram Hapus data pasien

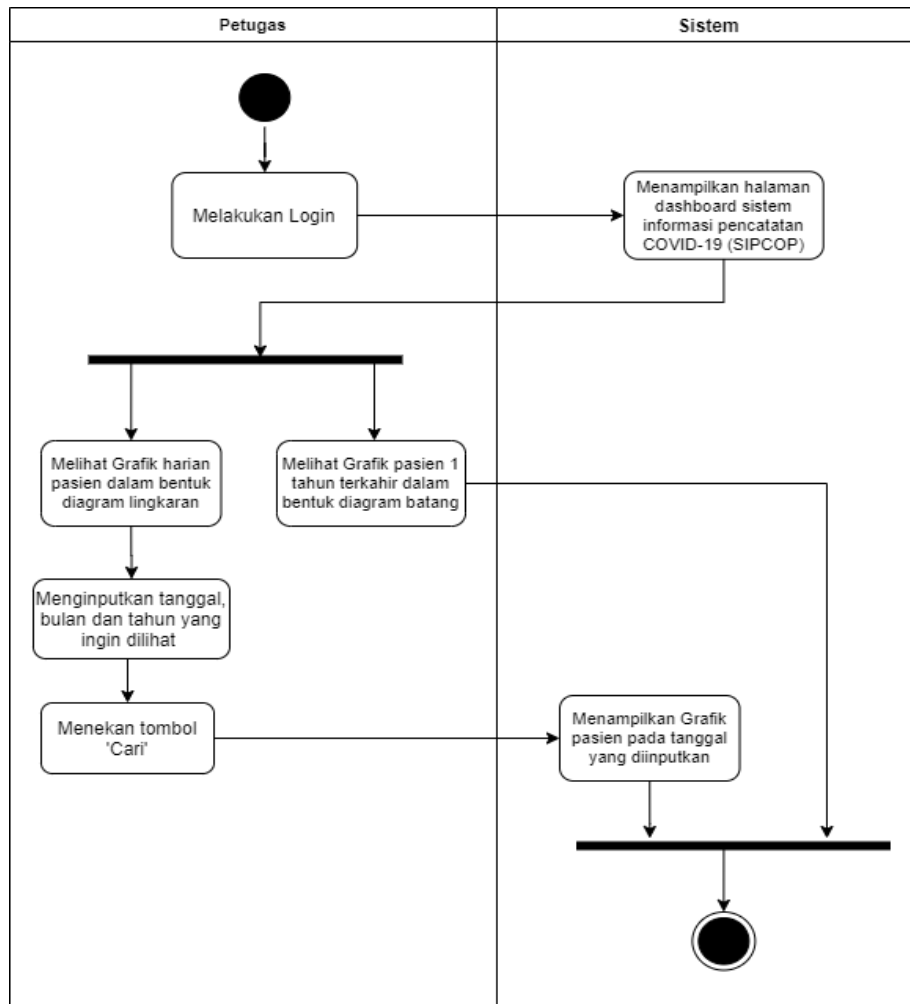
Berikut adalah Activity Diagram hapus data pasien yang akan dijelaskan pada gambar 5.11. Petugas pencatatan harus melakukan *login* terlebih dahulu, kemudian petugas dapat menghapus data pasien yang dipilih, dan pasien yang telah berpindah status menjadi 'selesai isolasi' dengan menekan tombol keluar.



Gambar 5. 11 Activity Diagram hapus data pasien

5.7.11 Activity Diagram Melihat grafik harian dan bulanan

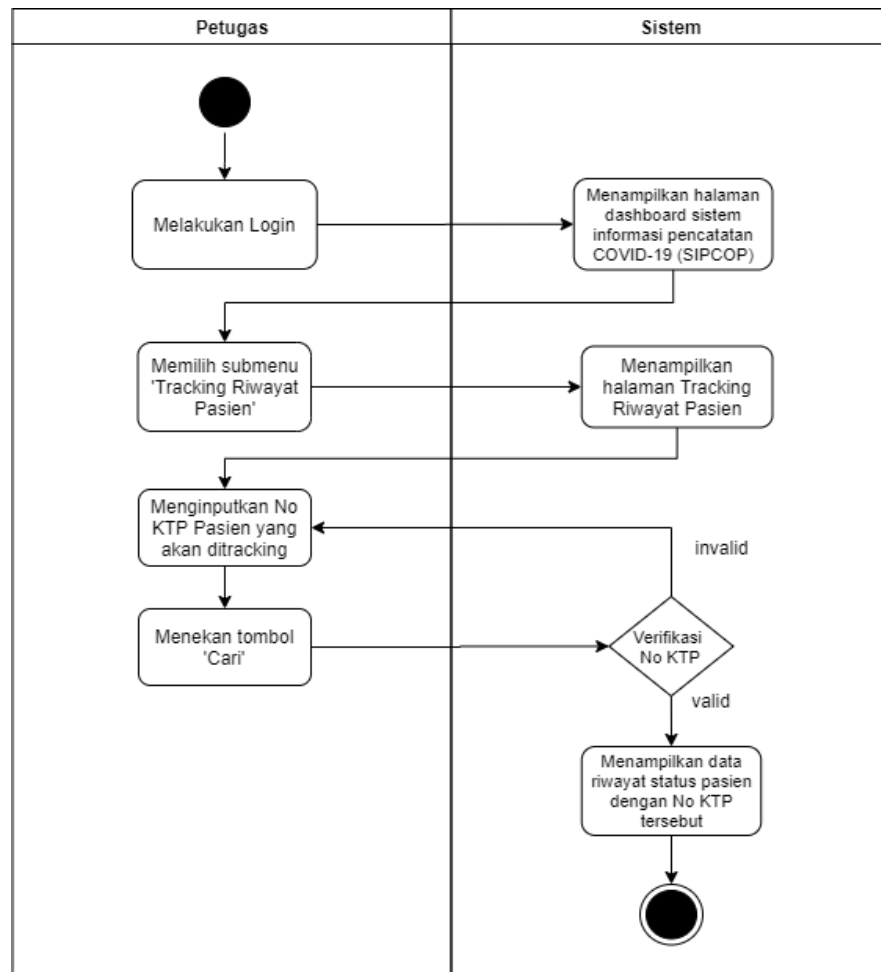
Berikut adalah Activity Diagram Melihat grafik harian dan bulanan yang akan dijelaskan pada gambar 5.12. Petugas pencatatan harus melakukan *login* terlebih dahulu, kemudian petugas dapat melihat halaman *Dashboard* yang terdiri dari 2 grafik laporan pasien yang terdaftar pada Rumah Sakit Darurat Gelora Joko Samudro. Yang pertama, grafik harian dalam bentuk diagram lingkaran pada hari tersebut, Jika petugas ingin melihat grafik pasien pada hari sebelumnya, petugas dapat memasukkan tanggal, bulan, dan tahun yang ingin dilihat. Yang kedua, petugas dapat melihat grafik pasien 1 tahun terakhir dalam bentuk diagram batang.



Gambar 5. 12 Activity Diagram melihat grafik harian dan bulanan

5.7.12 Activity Diagram Tracking Riwayat pasien

Berikut adalah activity diagram Tracking riwayat pasien yang akan dijelaskan pada gambar 5.13. Petugas pencatatan harus melakukan *login* terlebih dahulu, kemudian petugas dapat melakukan tracking atau penelusuran riwayat perubahan status dari pasien yang dicari. Dengan cara petugas dapat melakukan input No KTP, kemudian menekan tombol cari jika ditemukan maka sistem akan menampilkan data dari riwayat status pasien dengan No KTP tersebut, jika No KTP yang dimasukkan tidak ditemukan maka sistem tidak menampilkan apapun, dan kembali ke halaman tracking.

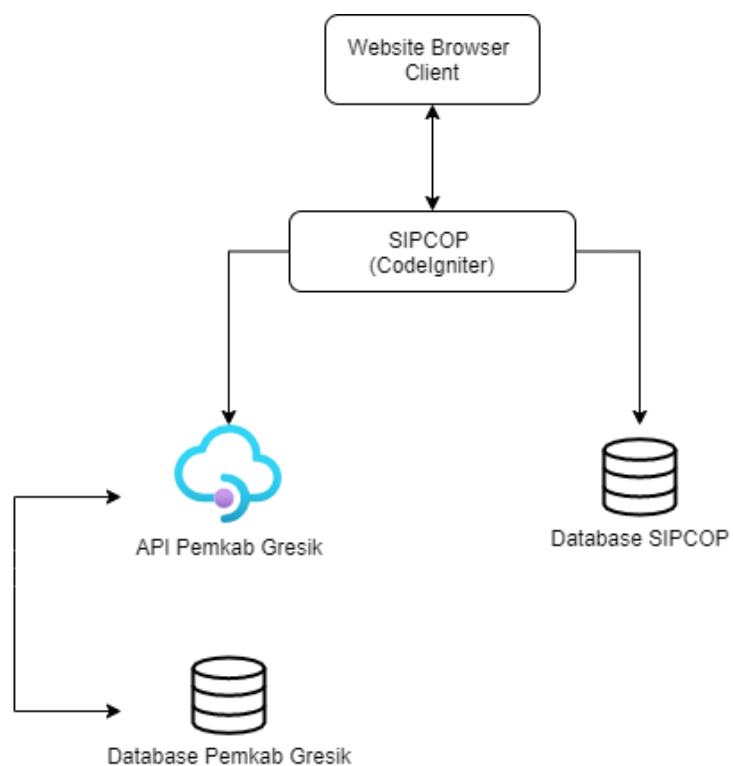


Gambar 5. 13 Activity Diagram Tracking riwayat pasien

BAB 6 PERANCANGAN SISTEM

6.1 Arsitektur Sistem

Pada arsitektur Sistem Informasi Pencatatan COVID-19 terdapat website *browser* sebagai *client*, CodeIgniter sebagai *server*, dan MySQL sebagai *database*. Proses otentikasi pada sistem dilakukan menggunakan *server* CodeIgniter untuk menentukan peran yang dimiliki akun pengguna. Setelah peran diketahui, *client* akan diarahkan ke services berdasarkan peran yang telah ditetapkan.



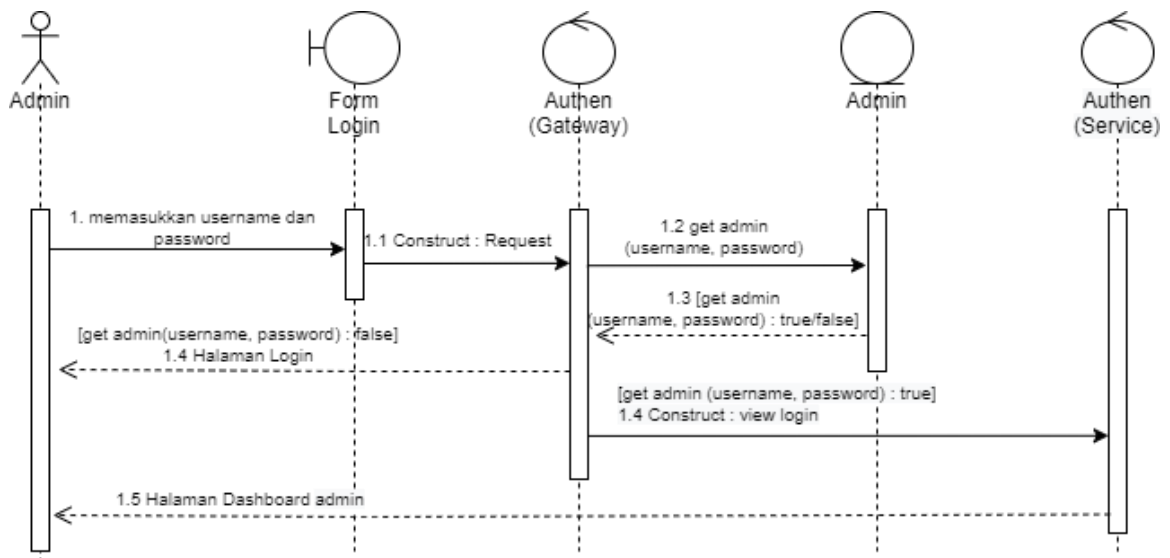
Gambar 6. 1 Arsitektur Sistem Informasi Pencatatan COVID-19

6.2 Pemodelan *Sequence Diagram*

Pemodelan *Sequence Diagram* ditujukan untuk memodelkan interaksi dan skenario yang dijalankan dalam sistem. Interaksi dalam Sistem Informasi Pencatatan COVID-19 (SIPCOP) dibagi menjadi login admin, login petugas, input petugas, input pasien, tambah ruang rawat, tambah status, dan tracking riwayat pasien.

6.2.1 *Sequence Diagram Login Admin*

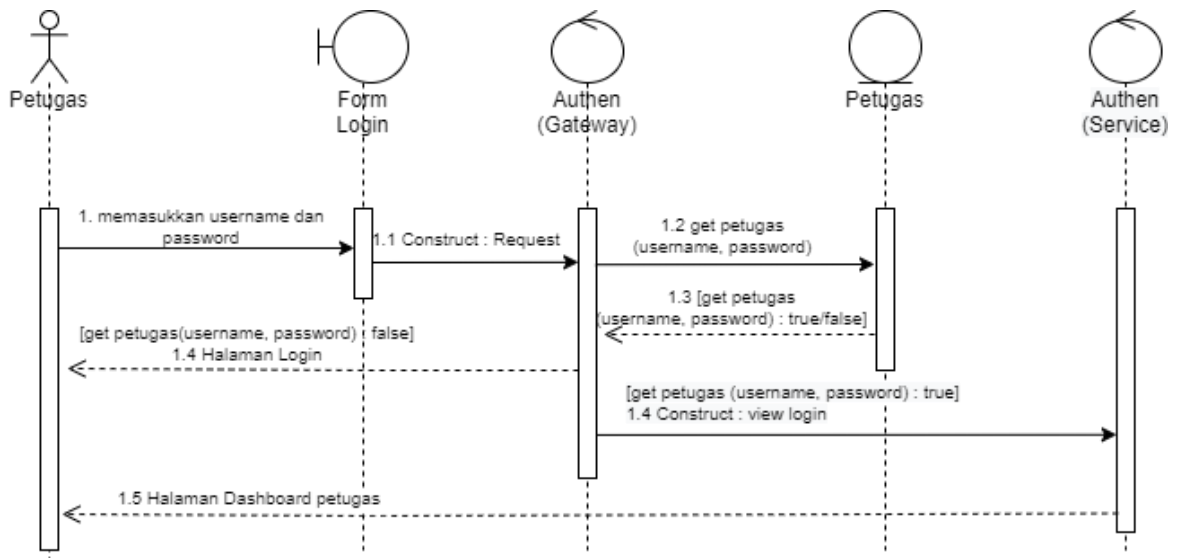
Berikut adalah *Sequence Diagram Login* oleh Admin yang dijelaskan pada gambar 6.2 Pertama-tama admin memasuki halaman *Login* admin, kemudian mengisi username dan password lalu menekan tombol “Masuk”. Sistem akan mengecek *data admin* ke *database* Admin dan mengembalikan halaman *admin*. Jika *credentials* tidak ditemukan di database, admin akan kembali menemui halaman *Login admin*. Jika *credentials* ditemukan di database, admin akan menemui halaman *dashboard*.



Gambar 6. 2 Sequence Diagram Login Admin

6.2.2 *Sequence Diagram Login petugas*

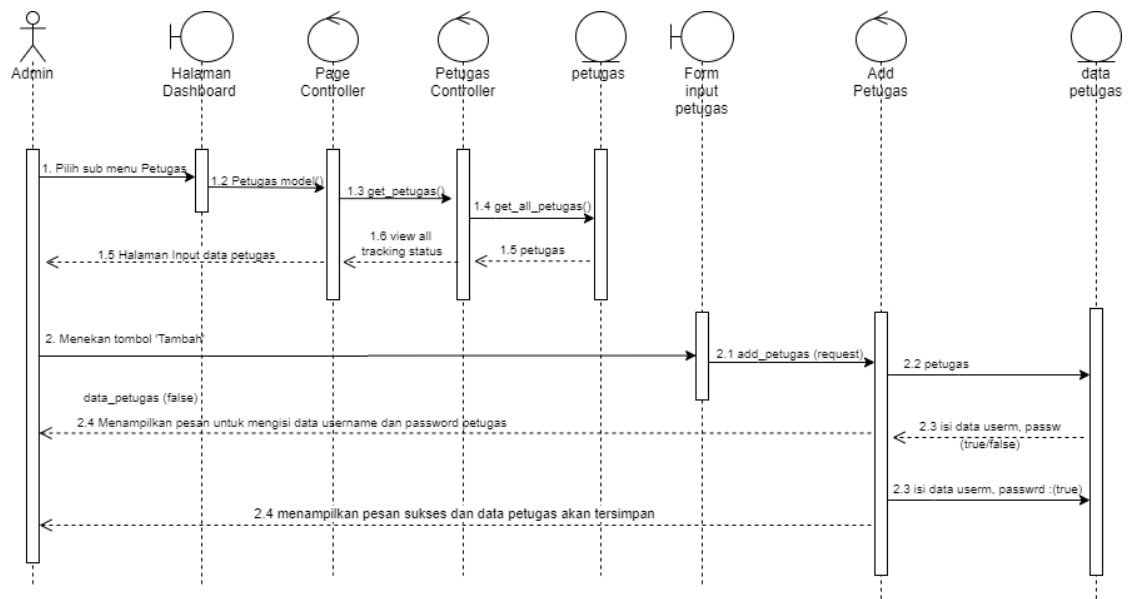
Berikut adalah *Sequence Diagram Login* oleh petugas yang dijelaskan pada gambar 6.3 Pertama-tama petugas memasuki halaman *Login* petugas, kemudian mengisi username dan password lalu menekan tombol “Masuk”. Sistem akan mengecek *data* petugas ke *database* petugas dan mengembalikan halaman tersebut. Jika *credentials* tidak ditemukan di database, petugas akan kembali menemui halaman *Login* petugas. Jika *credentials* ditemukan di database, petugas akan menemui halaman *dashboard*.



Gambar 6. 3 Sequence Diagram Login Petugas

6.2.3 Sequence Diagram Input data petugas

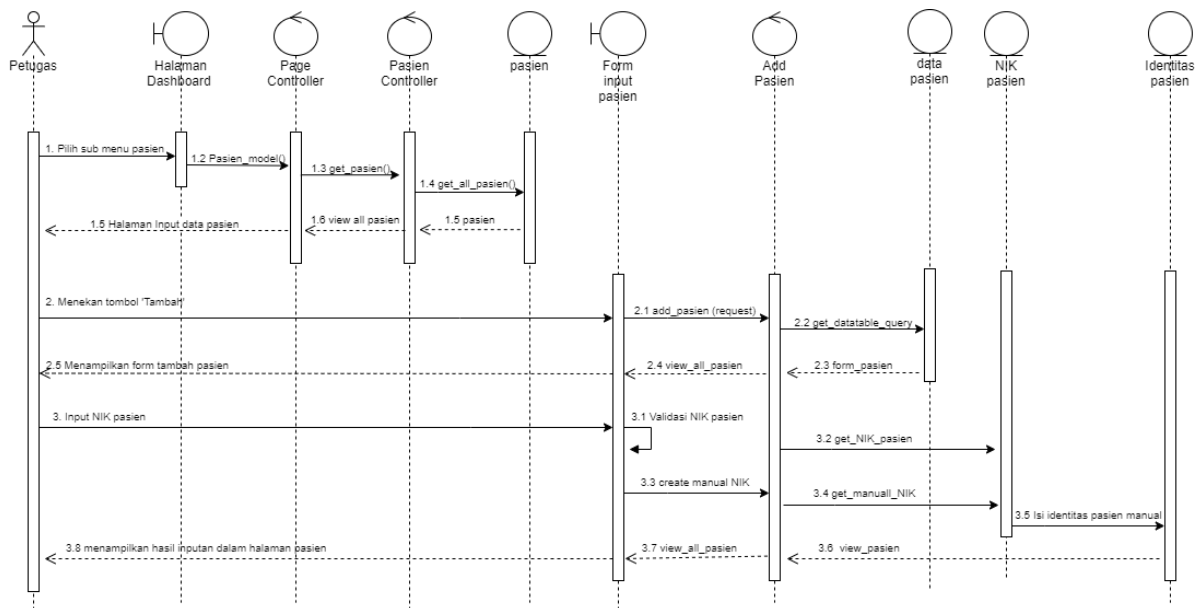
Berikut adalah *Sequence Diagram* input data petugas oleh admin yang dijelaskan pada gambar 6.4. Pertama-tama admin memasuki halaman *dashboard* yang merupakan daftar data petugas. Kemudian untuk menambah petugas baru maka admin menekan tombol “Tambah” dimana sistem akan menampilkan form input petugas dengan wajib mengisi username dan password sebagai kewenangan petugas. Selanjutnya inputan tersebut akan disimpan dalam database petugas, kemudian sistem akan mengembalikan dengan menemui halaman dashboard daftar petugas.



Gambar 6. 4 Sequence Diagram input data petugas

6.2.4 Sequence Diagram Input data pasien

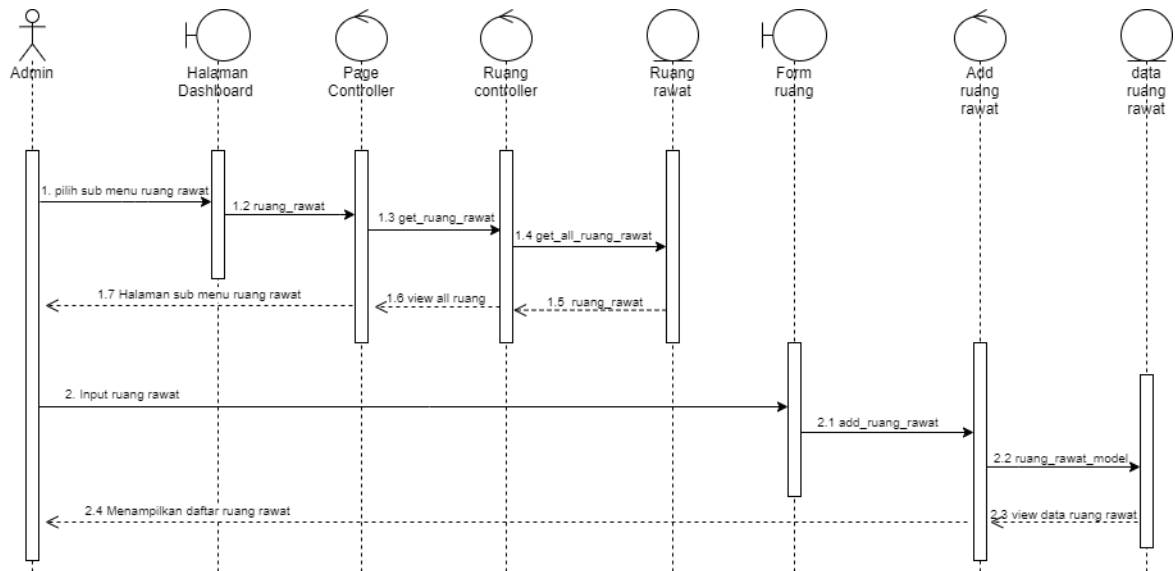
Berikut adalah *Sequence Diagram* Input data pasien oleh petugas yang dijelaskan pada gambar 6.5. Petugas memasuki halaman *dashboard* yang merupakan grafik pantauan untuk pasien. Kemudian untuk menambah pasien baru maka admin menekan tombol “Tambah”. Lalu sistem akan menampilkan form input pasien dengan wajib mengisi NIK pasien, dimana jika NIK yang diinput adalah warga Gresik maka sistem secara otomatis menampilkan identitas lengkap pasien. Sedangkan jika NIK yang diinput adalah warga luar Gresik maka petugas secara manual mengisi identitas pasien. Selanjutnya inputan tersebut akan disimpan dalam database pasien, kemudian sistem akan mengembalikan dengan menemui halaman daftar pasien.



Gambar 6. 5 Sequence Diagram input data pasien

6.2.5 Sequence Diagram Input ruang rawat

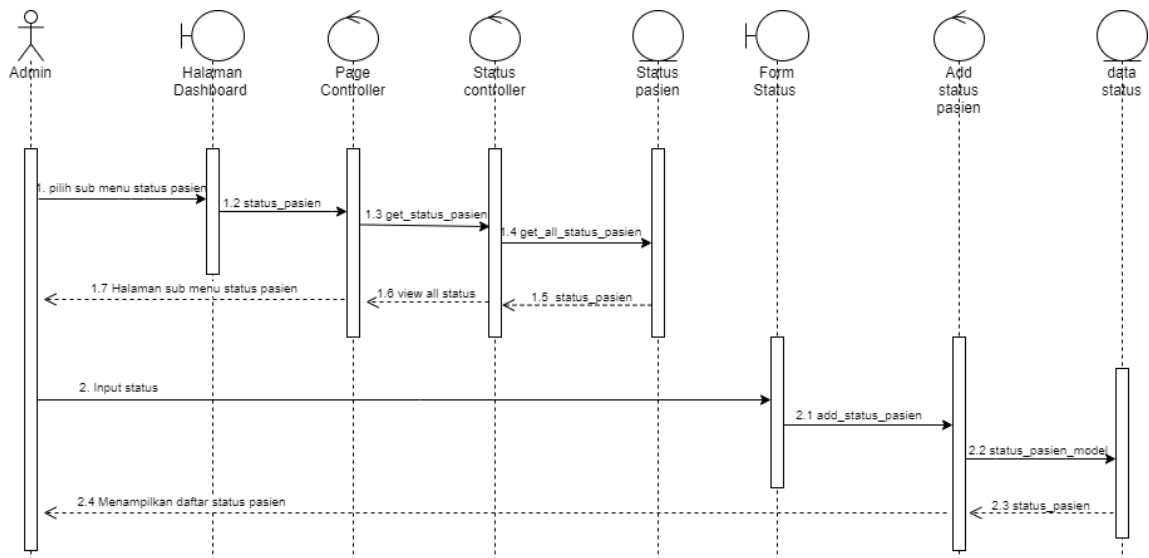
Berikut adalah *Sequence Diagram* input ruang rawat oleh admin yang dijelaskan pada gambar 6.6. Admin memasuki halaman *dashboard* admin. Kemudian memilih sub menu ruang rawat, lalu sistem akan menampilkan halaman daftar ruang rawat. Untuk menambah data ruang rawat maka admin menekan tombol ‘Tambah’ sehingga akan ditampilkan form input ruang rawat. Selanjutnya inputan tersebut akan disimpan dalam database ruang rawat.



Gambar 6. 6 Sequence Diagram input ruang rawat

6.2.6 Sequence Diagram Input status pasien

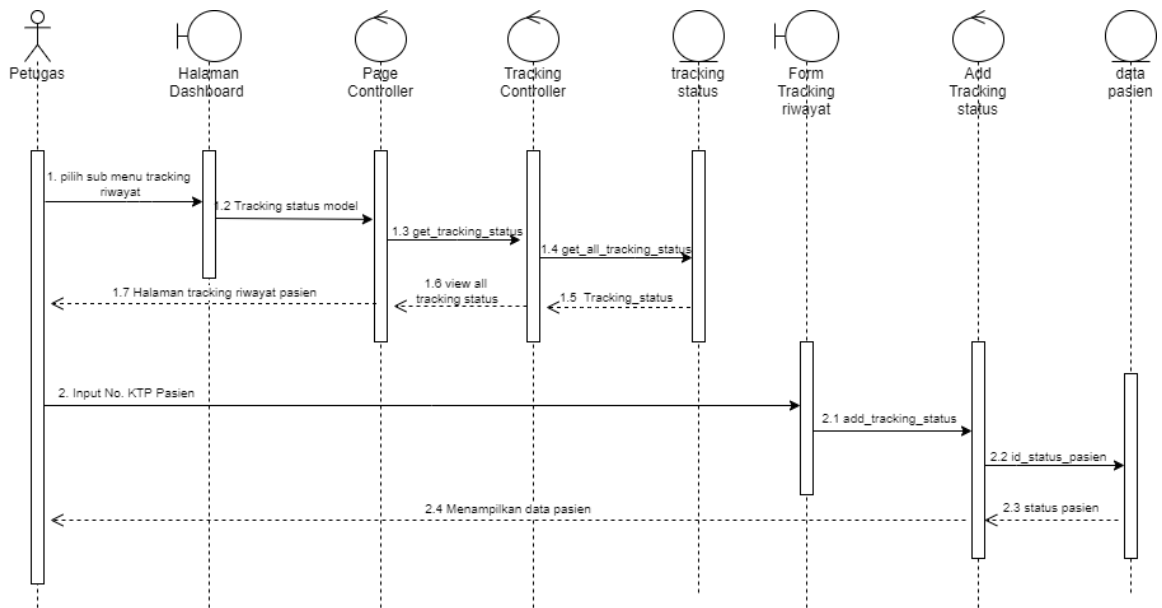
Berikut adalah *Sequence Diagram* input status pasien oleh admin yang dijelaskan pada gambar 6.7. Admin memasuki halaman *dashboard* admin. Kemudian memilih sub menu status pasien, lalu sistem akan menampilkan halaman daftar status pasien. Untuk menambah data status maka admin menekan tombol 'Tambah' sehingga akan ditampilkan form input status. Selanjutnya inputan tersebut akan disimpan dalam database status pasien.



Gambar 6. 7 Sequence Diagram input status pasien

6.2.7 Sequence Diagram Tracking riwayat pasien

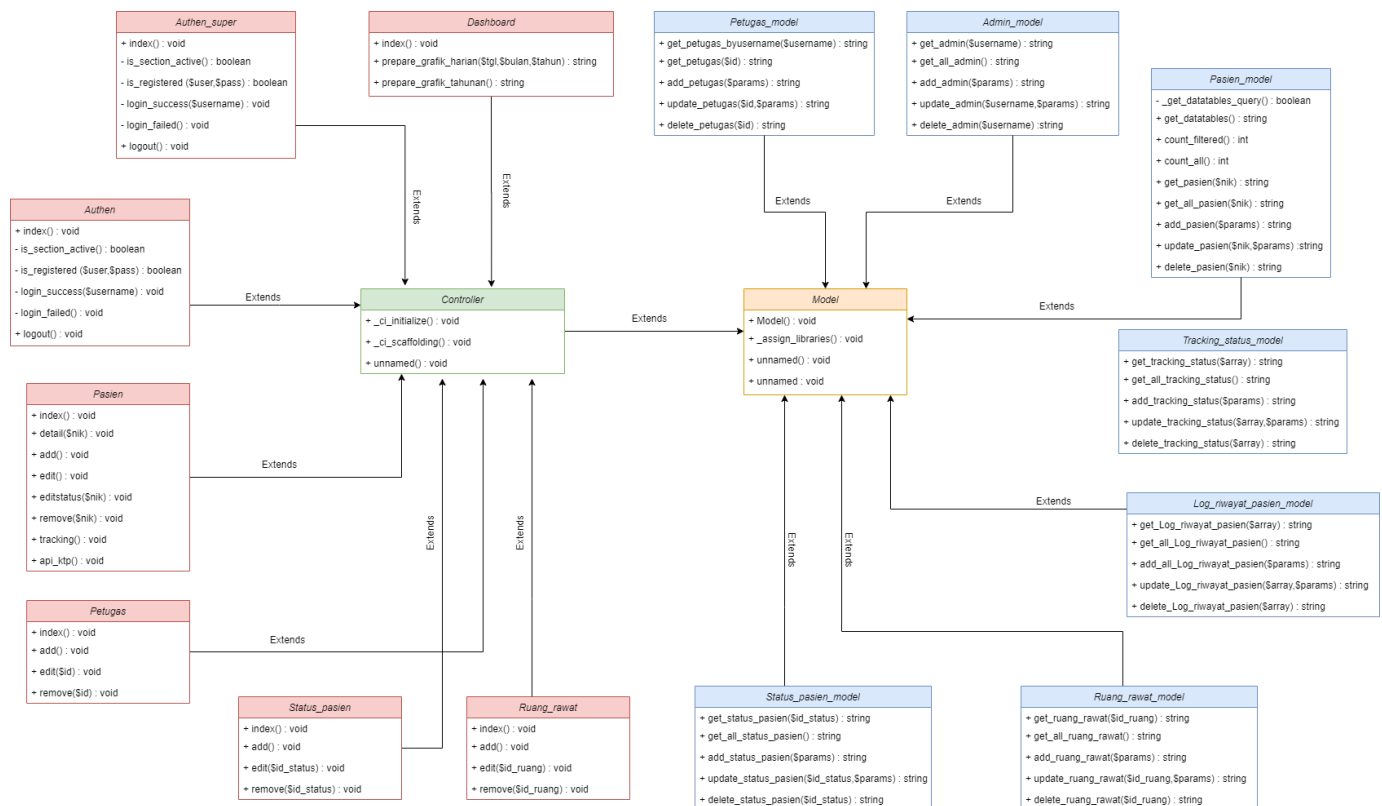
Berikut adalah *Sequence Diagram Tracking* riwayat pasien oleh Petugas yang dijelaskan pada gambar 6.8. Petugas memasuki halaman *dashboard* kemudian memilih sub menu tracking riwayat pasien. Lalu untuk melihat tracking riwayat pasien, petugas menginputkan NIK pasien yang telah terdaftar sebelumnya dan sistem akan menampilkan data pasien tersebut. Sedangkan jika menginput NIK pasien yang tidak terdaftar, pencarian riwayat pasien akan menampilkan pesan gagal.



Gambar 6. 8 Sequence Diagram Tracking riwayat pasien

6.3 Pemodelan *Class Diagram*

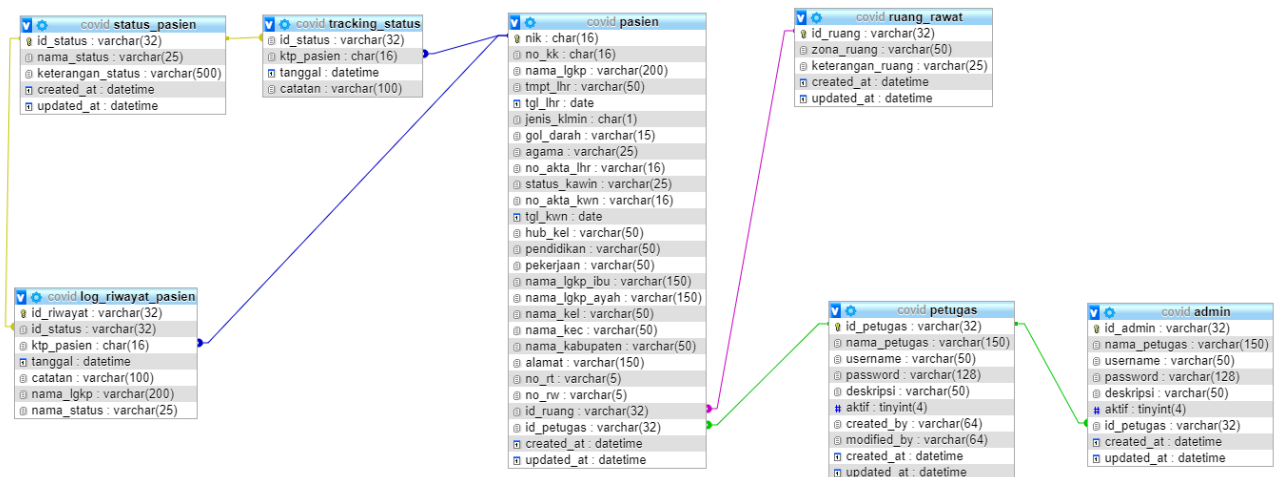
Class diagram merupakan struktur sistem yang digambarkan melalui kelas-kelas yang mempunyai atribut, *method*, dan relasi antar kelas. Kelas-kelas yang dirancang nantinya akan diimplementasikan ke *source code* pada *Framework CodeIgniter*. *Class Diagram* yang dibuat nantinya memperlihatkan relasi antara *Controller* dan *Model* pada *Framework CodeIgniter*. *Class Diagram* dirancang menjadi dua jenis berdasarkan kelas abstrak yang di-*extend* yaitu kelas *Controller* dan kelas *Model* yang kelas turunannya saling berelasi. Pada gambar 6.9, terdapat 7 kelas turunan dari kelas *Controller*, yaitu *Authen_super*, *Authen*, *Dashboard*, *Pasien*, *Petugas*, *Status_pasien*, *Ruang_rawat*. Ada 7 kelas yang merupakan turunan dari kelas *Model*, yaitu *Petugas_model*, *Admin_model*, *Pasien_model*, *Tracking_status_model*, *Log_rwayat_pasien_model*, *Ruang_rawat_model*, *Status_pasien_model*. Masing-masing memiliki fungsionalitas yang berbeda beda yang merupakan gambaran keseluruhan sistem.



Gambar 6. 9 Class Diagram Sistem Informasi Pencatatan COVID-19

6.4 Pemodelan PDM (Physical Data Model)

PDM atau *Physical Data Model* adalah *diagram* yang memodelkan hubungan atau relasi antar objek data sebagai rancangan dalam membangun sebuah *database*. Berikut adalah Physical Data Model yang dijelaskan pada Gambar 6.10 yang terdiri dari 7 tabel, yakni tabel pasien, tabel petugas, tabel admin, tabel tracking_status, tabel log_riwayat_pasien, tabel status_pasien, dan tabel ruang_rawat.



Gambar 6. 10 Physical Data Model Sistem Informasi Pencatatan COVID-19

6.4.1 Perancangan Tabel Admin

Tabel Admin digunakan untuk menyimpan data dari Admin. Dalam tabel admin terdapat beberapa atribut diantaranya :

- id_admin sebagai *Primary Key* dengan tipe data *varchar*,
- nama_petugas dengan tipe data *Varchar*,
- username dengan tipe data *Varchar*,
- password dengan tipe data *Varchar*,
- deskripsi dengan tipe data *Varchar*,
- aktif dengan tipe data *tinyint*,
- id_petugas dengan tipe data *Varchar*,
- created_at dengan tipe data *datetime*
- updated_at dengan tipe data *datetime*,

6.4.2 Perancangan Tabel Pasien

Tabel pasien digunakan untuk menyimpan data diri lengkap dari Pasien yang masuk di rumah sakit darurat Gelora Joko Samudro. Dalam tabel pasien terdapat beberapa atribut diantaranya :

- nik sebagai *Primary Key* dengan tipe data *char*,
- no_kk dengan tipe data *char*,
- nama_lgkp dengan tipe data *Varchar*,
- tmpt_lhr dengan tipe data *Varchar*,
- tgl_lhr dengan tipe data *date*,
- jenis_klmin dengan tipe data *char*,
- gol_darah dengan tipe data *Varchar*,
- agama dengan tipe data *Varchar*,
- no_akta_lhr dengan tipe data *Varchar*,
- status_kawin dengan tipe data *Varchar*,
- no_akta_kwn dengan tipe data *Varchar*,
- tgl_kwn dengan tipe data *date*,
- hub_kel dengan tipe data *Varchar*,
- pendidikan dengan tipe data *Varchar*,
- pekerjaan dengan tipe data *Varchar*,
- nama_lgkp_ibu dengan tipe data *Varchar*,
- nama_lgkp_ayah dengan tipe data *Varchar*,
- nama_kel Lead dengan tipe data *Varchar*,
- nama_kec dengan tipe data *Varchar*,
- nama_kabupaten dengan tipe data *Varchar*,
- alamat dengan tipe data *Varchar*,
- no_rt dengan tipe data *Varchar*,
- no_rw dengan tipe data *Varchar*,
- id_ruang dengan tipe data *Varchar*,
- id_petugas dengan tipe data *Varchar*,
- created_at dengan tipe data *datetime*,
- updated_at dengan tipe data *datetime*

6.4.3 Perancangan Tabel Petugas

Tabel Admin digunakan untuk menyimpan data dari Petugas pencatatan sistem. Dalam tabel petugas terdapat beberapa atribut diantaranya :

- id_petugas sebagai *Primary Key* dengan tipe data *Varchar*,
- nama_petugas dengan tipe data *Varchar*,
- username dengan tipe data *Varchar*,
- password dengan tipe data *Varchar*,
- deskripsi dengan tipe data *Varchar*,
- aktif dengan tipe data *tinyint*,
- created_by dengan tipe data *Varchar*,
- modified_by dengan tipe data *Varchar*,

- created_at dengan tipe data *datetime*,
- updated_at dengan tipe data *datetime*

6.4.4 Perancangan Tabel Ruang rawat

Tabel Ruang rawat digunakan untuk menyimpan data pengelolaan daftar ruang yang terdaftar pada rumah sakit darurat Gelora Joko Samudro. Dalam tabel ruang rawat terdapat beberapa atribut diantaranya :

- id_ruang sebagai *Primary Key* dengan tipe data *Varchar*,
- zona_ruang dengan tipe data *Varchar*,
- keterangan_ruang dengan tipe data *Varchar*
- created_at dengan tipe data *datetime*,
- updated_at dengan tipe data *datetime*

6.4.5 Perancangan Tabel Status pasien

Tabel status pasien digunakan untuk menyimpan data pengelolaan daftar sebutan status yang digunakan berdasarkan peraturan kemenkes. Dalam tabel status pasien terdapat beberapa atribut diantaranya :

- id_status sebagai *Primary Key* dengan tipe data *Varchar*,
- nama_status dengan tipe data *Varchar*,
- keterangan_status dengan tipe data *Varchar*
- created_at dengan tipe data *datetime*,
- updated_at dengan tipe data *datetime*

6.4.6 Perancangan Tabel Log riwayat pasien

Tabel Log riwayat pasien digunakan untuk menyimpan riwayat pergantian data tiap perubahan status dari tiap tiap pasien. Dalam tabel log riwayat pasien terdapat beberapa atribut diantaranya :

- id_riwayat sebagai *Primary Key* dengan tipe data *Varchar*,
- id_status dengan tipe data *Varchar*,
- ktp_pasien dengan tipe data *Varchar*,
- tanggal dengan tipe data *datetime*,
- catatan dengan tipe data *Varchar*,
- nama_lgkp dengan tipe data *Varchar*,
- nama_status dengan tipe data *Varchar*

6.4.7 Perancangan Tabel Tracking status

Tabel tracking status pasien digunakan untuk menyimpan data tiap perubahan status dari tiap tiap pasien. Dalam tabel tracking status terdapat beberapa atribut diantaranya :

- id_status sebagai *Foreign Key* dengan tipe data *Varchar*,

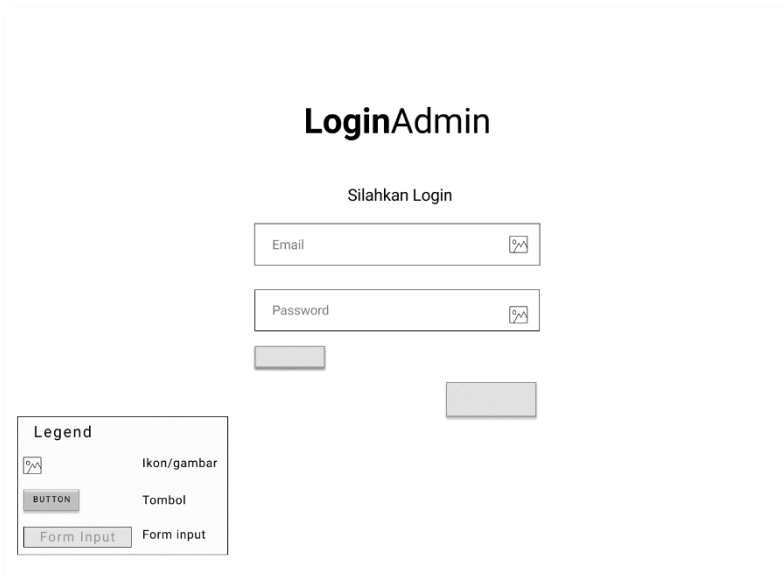
- ktp_pasien sebagai *Foreign Key* dengan tipe data *Varchar*,
- tanggal dengan tipe data *datetime*,
- catatan dengan tipe data *Varchar*

6.5 Perancangan *User Interface*

User Interface merupakan tampilan yang berhubungan langsung dengan pengguna untuk melakukan interaksi dengan sistem. Perancangan *User Interface* digunakan sebagai acuan untuk membuat tampilan dari Sistem Informasi Pencatatan *COVID-19* (SIPCOP).

6.5.1 User Interface Login Admin dan Login Petugas

Gambar 6.11 dan Gambar 6.12 menunjukkan rancangan dari *user interface* halaman *Login Admin* dan *Login Petugas*. Rancangan *user interface* ini akan menjadi acuan dalam mengimplementasikan *user interface* halaman *Login Admin* dan *Login Petugas*.



The mockup for the Admin Login page features a central title 'LoginAdmin' and a subtitle 'Silahkan Login'. Below these are two input fields labeled 'Email' and 'Password', each with a small icon on the right. Underneath the password field are two buttons: a smaller one on the left and a larger one on the right. A legend box in the bottom-left corner defines the symbols: a box with an icon for 'Ikon/gambar', a box labeled 'BUTTON' for 'Tombol', and a box labeled 'Form Input' for 'Form input'.

Gambar 6. 11 User Interface Login Admin

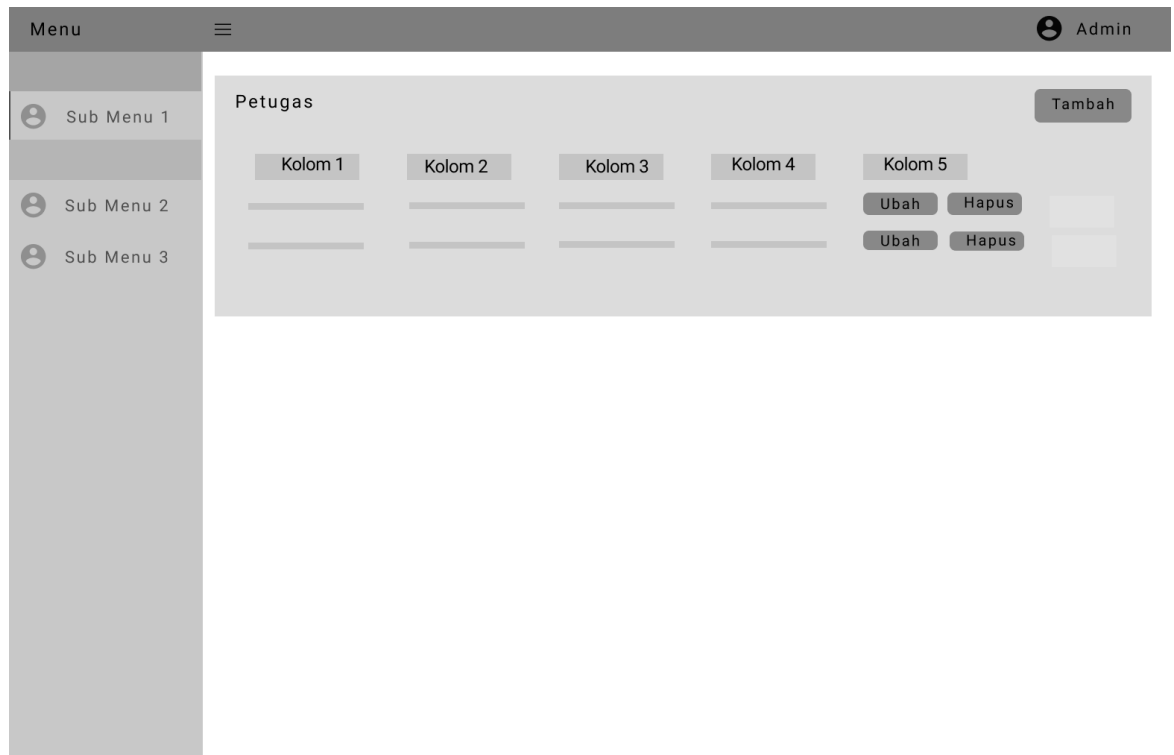


The mockup for the Petugas Login page is identical in layout to the Admin Login page. It has a central title 'LoginPetugas' and a subtitle 'Silahkan Login'. It includes 'Email' and 'Password' input fields with icons, followed by two buttons (one small, one large). A legend box in the bottom-left corner defines the symbols: a box with an icon for 'Ikon/gambar', a box labeled 'BUTTON' for 'Tombol', and a box labeled 'Form Input' for 'Form input'.

Gambar 6. 12 User Interface Login petugas

6.5.2 User Interface Halaman Dashboard Admin (Input Petugas)

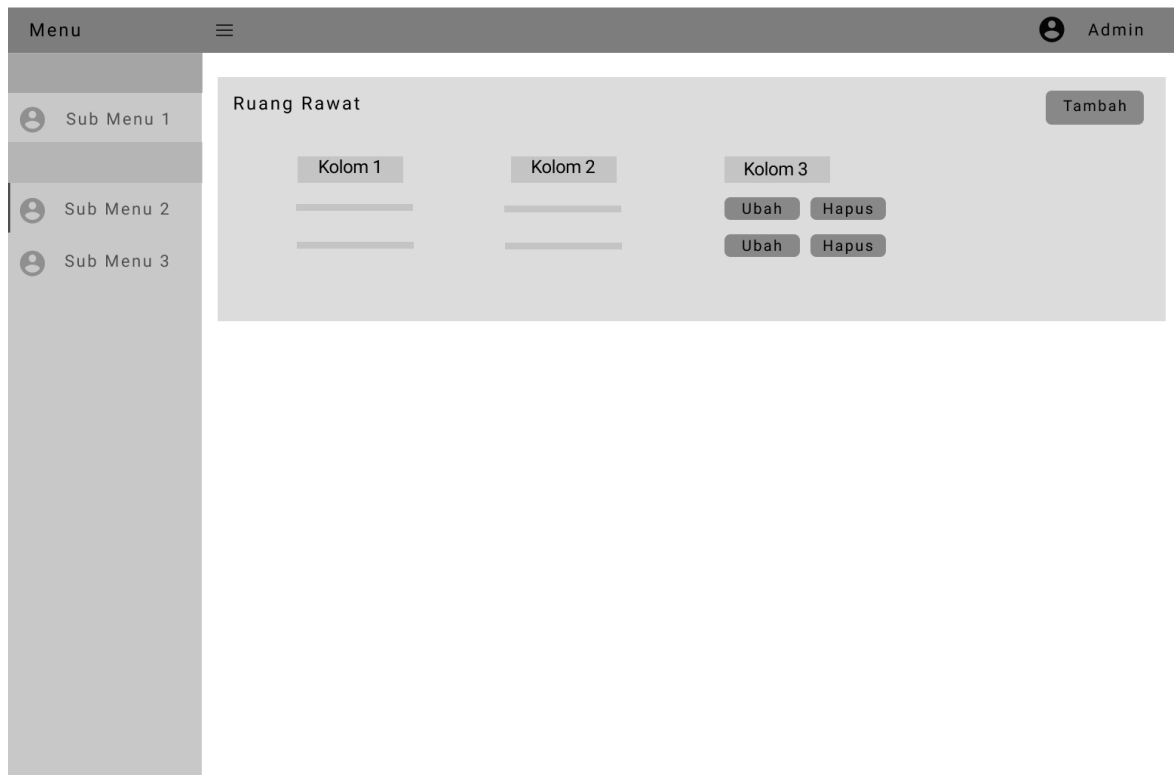
Gambar 6.13 menunjukkan rancangan dari *user interface* halaman Dashboard Admin (Input Petugas). Rancangan *user interface* ini akan menjadi acuan dalam mengimplementasikan *user interface* halaman Dashboard dengan 3 sub menu yaitu input petugas, ruang rawat, dan status.



Gambar 6. 13 User Interface Halaman Dashboard Admin

6.5.3 *User Interface* halaman Ruang Rawat

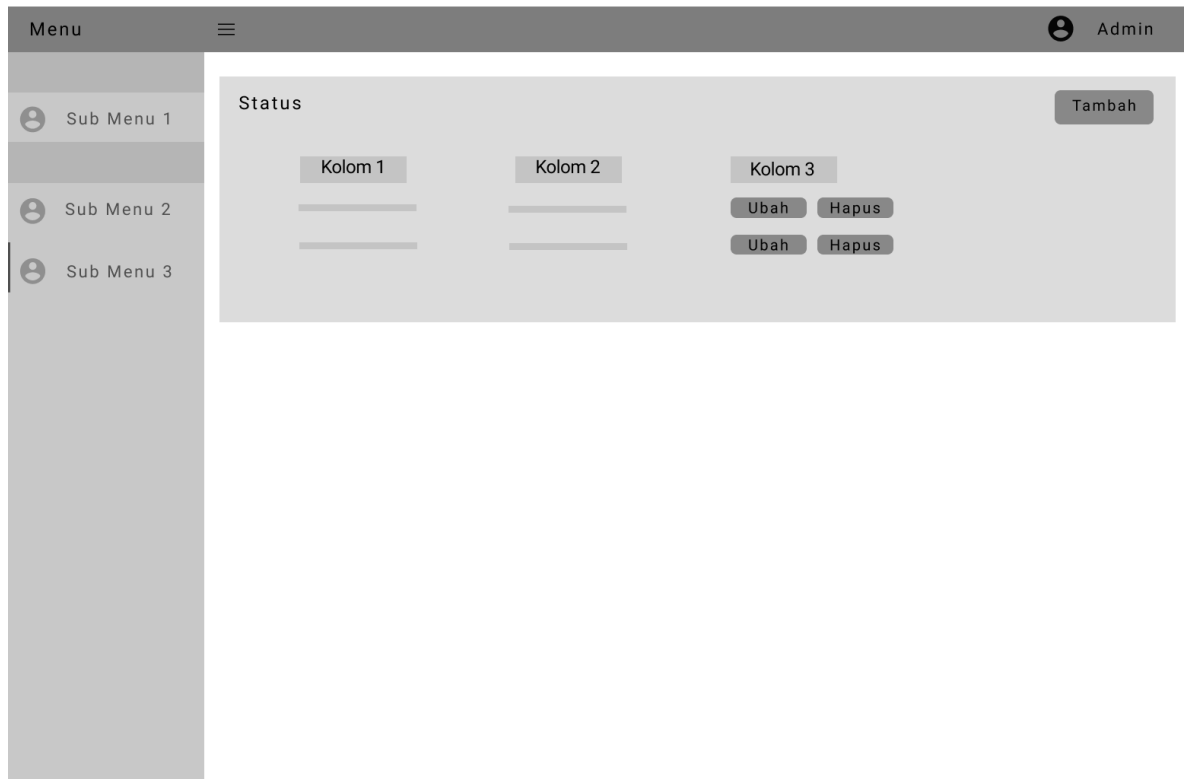
Gambar 6.14 menunjukkan rancangan dari *user interface* halaman ruang rawat. Rancangan *user interface* ini akan menjadi acuan dalam mengimplementasikan *user interface* halaman form ruang rawat.



Gambar 6. 14 User Interface halaman Ruang Rawat

6.5.4 *User Interface* halaman Status

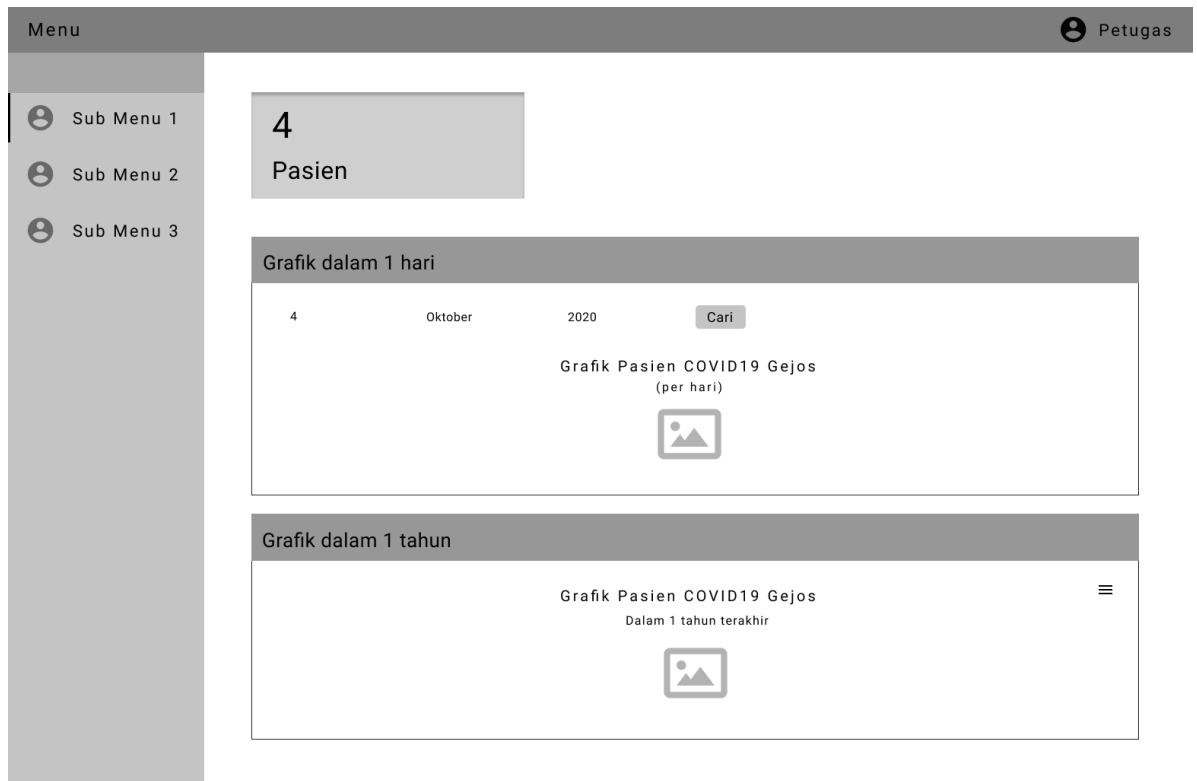
Gambar 6.15 menunjukkan rancangan dari *user interface* halaman status. Rancangan *user interface* ini akan menjadi acuan dalam mengimplementasikan *user interface* halaman ruang rawat.



Gambar 6. 15 User Interface halaman Status

6.5.5 *User Interface* halaman Dashboard petugas (Grafik)

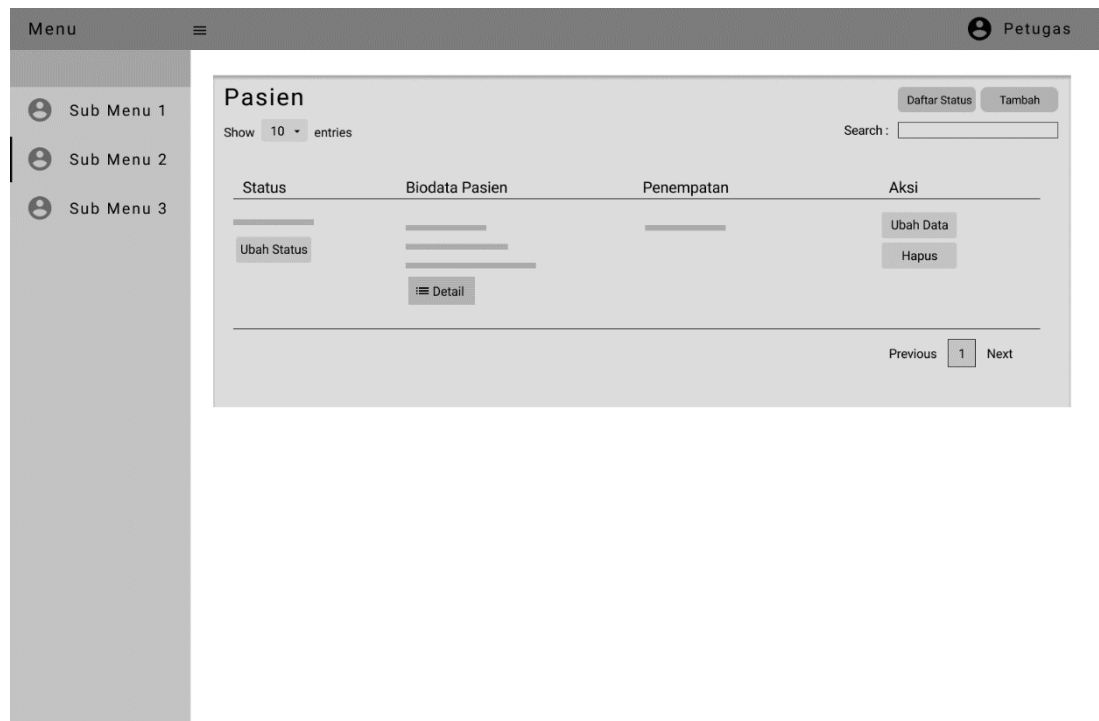
Gambar 6.16 Menunjukkan rancangan dari *user interface* halaman dashboard petugas yaitu berisi grafik harian dan bulanan. Rancangan *user interface* ini akan menjadi acuan dalam mengimplementasikan *user interface* halaman dashboard petugas.



Gambar 6. 16 User Interface halaman Dashboard petugas (Grafik)

6.5.6 *User Interface* halaman Input pasien

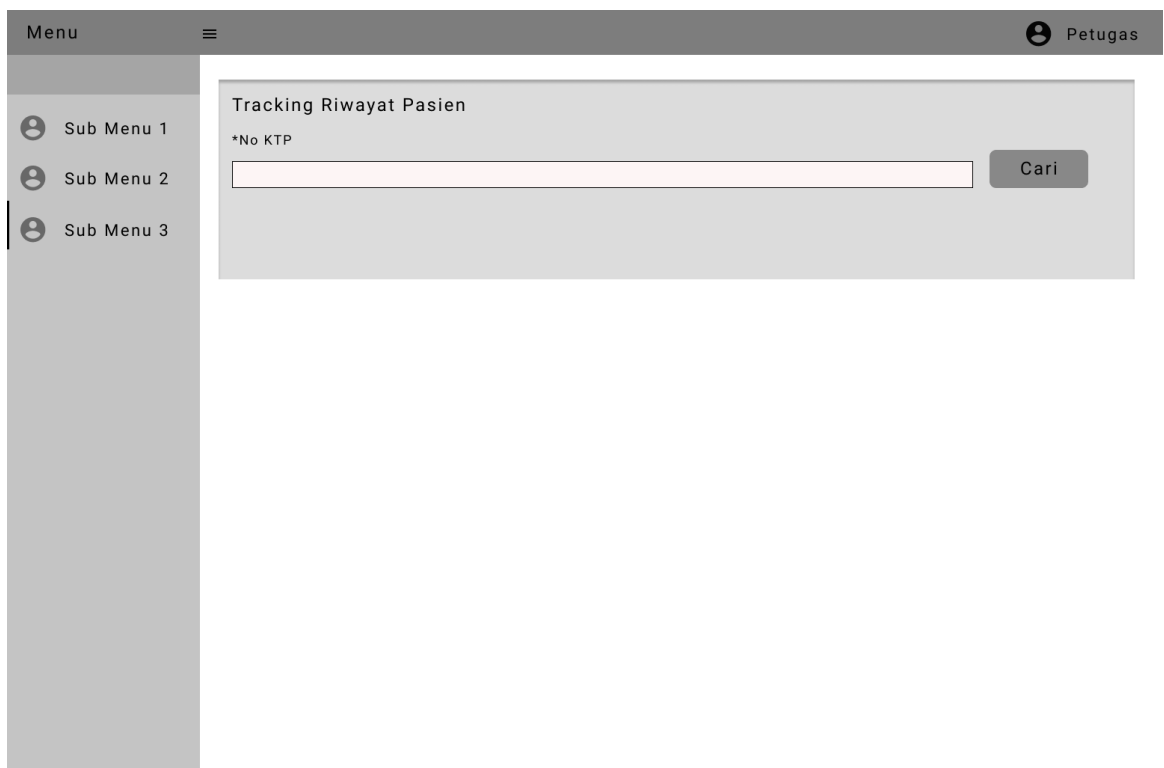
Gambar 6.17 Menunjukkan rancangan dari *user interface* halaman input pasien. Rancangan *user interface* ini akan menjadi acuan dalam mengimplementasikan *user interface* halaman input pasien.



Gambar 6. 17 User Interface halaman Input pasien

6.5.7 *User Interface* halaman Tracking riwayat pasien

Gambar 6.18 Menunjukkan rancangan dari *user interface* halaman tracking riwayat pasien. Rancangan *user interface* ini akan menjadi acuan dalam mengimplementasikan user interface halaman tracking riwayat pasien.



Gambar 6. 18 User Interface halaman Tracking riwayat pasien

BAB 7 IMPLEMENTASI

Pada bab ini akan dilakukan proses pengembangan sistem dengan menerapkan rancangan sistem. Implementasi pada Sistem Informasi Pencatatan COVID-19 (SIPCOP) meliputi :

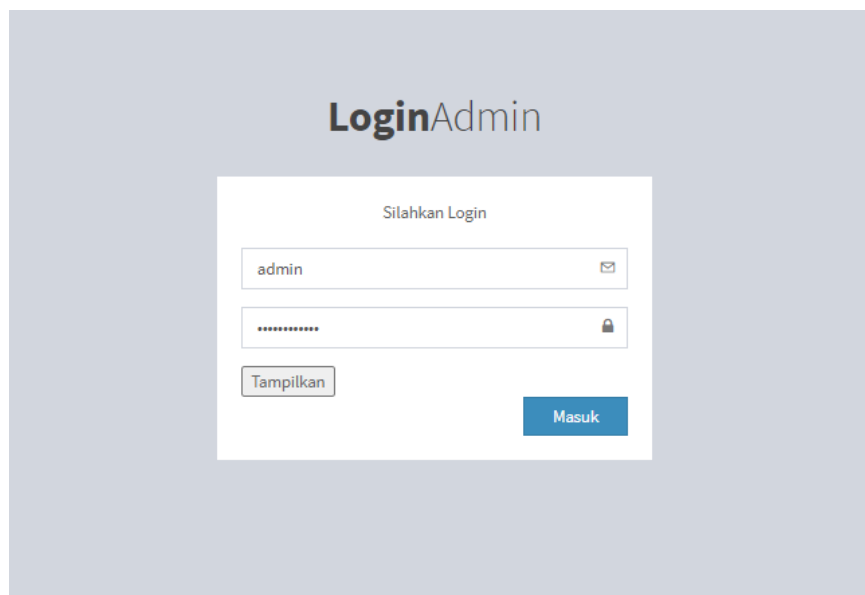
1. Implementasi *user interface*
2. Implementasi *database*
3. Implementasi kode program

7.1 Implementasi *User Interface*

Subbab ini memaparkan hasil implementasi dari *user interface* yang mengacu pada perancangan sebelumnya.

7.1.1 *User Interface Login Admin*

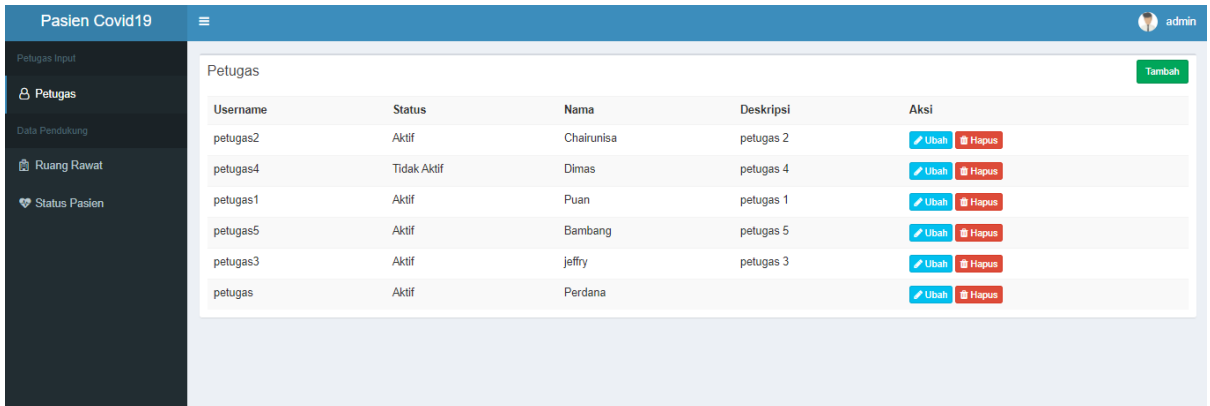
Gambar 7.1 menunjukkan implementasi dari *user interface* halaman Login Admin yang mengacu pada perancangan *user interface* halaman Login Admin.

The image shows a web-based login interface for an administrator. At the top, the text "LoginAdmin" is displayed in a large, bold, dark font. Below this, a white rectangular box contains the login form. Inside the box, the text "Silahkan Login" is centered. There are two input fields: the first is for the username, containing the text "admin", and the second is for the password, containing a series of dots. To the right of each input field is a small icon (an envelope for the username and a padlock for the password). Below the password field is a button labeled "Tampilkan" (Show). To the right of the "Tampilkan" button is a blue button labeled "Masuk" (Login). The entire form is set against a light gray background.

Gambar 7. 1 User Interface Login Admin

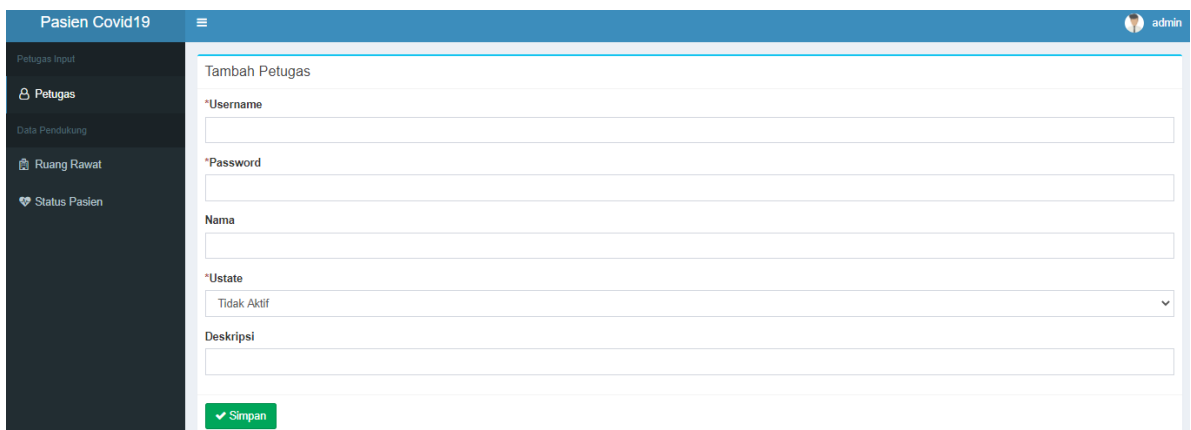
7.1.2 User Interface Halaman Input data petugas

Gambar 7.2 menunjukkan implementasi dari *user interface* halaman Dashboard Admin yang merupakan halaman input data petugas. Kemudian pada Gambar 7.3 menunjukkan halaman tambah form tambah petugas apabila user menekan tombol 'Tambah'. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman Input data petugas.



Username	Status	Nama	Deskripsi	Aksi
petugas2	Aktif	Chairunisa	petugas 2	Ubah Hapus
petugas4	Tidak Aktif	Dimas	petugas 4	Ubah Hapus
petugas1	Aktif	Puan	petugas 1	Ubah Hapus
petugas5	Aktif	Bambang	petugas 5	Ubah Hapus
petugas3	Aktif	jeffry	petugas 3	Ubah Hapus
petugas	Aktif	Perdana		Ubah Hapus

Gambar 7. 2 User Interface Halaman daftar petugas



Tambah Petugas

*Username

*Password

Nama

*Ustate

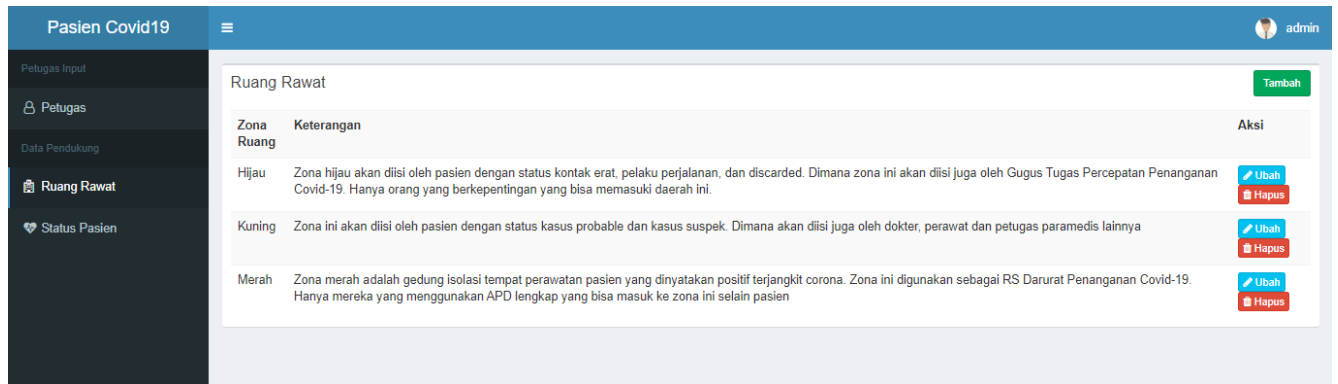
Deskripsi

Simpan

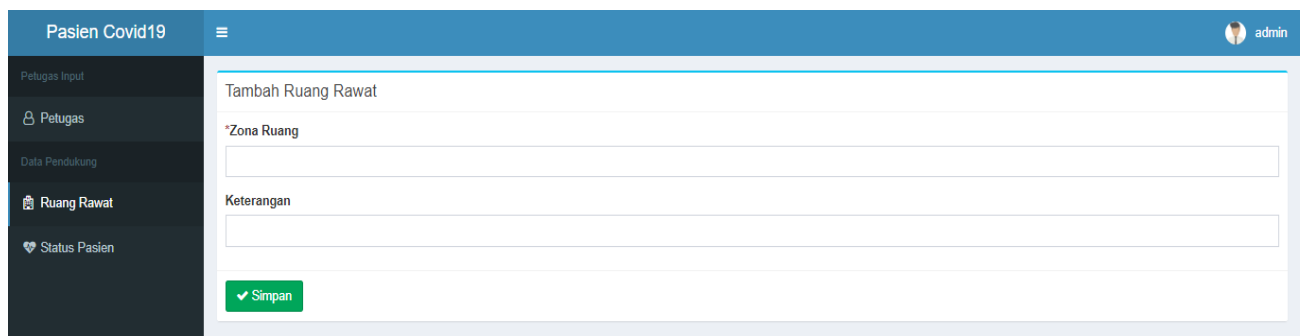
Gambar 7. 3 User Interface Halaman Input data petugas

7.1.3 User Interface Halaman Ruang rawat

Gambar 7.4 menunjukkan implementasi dari *user interface* halaman ruang rawat pada aktor Admin. Kemudian pada Gambar 7.5 menunjukkan halaman tambah form ruang rawat apabila user menekan tombol 'Tambah'. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman ruang rawat.



Gambar 7. 4 User Interface Halaman daftar Ruang rawat



Gambar 7. 5 User Interface Halaman input Ruang rawat

7.1.4 User Interface Halaman Status

Gambar 7.6 menunjukkan implementasi dari *user interface* halaman status Lead pada aktor Admin. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman status. Kemudian pada Gambar 7.7 menunjukkan halaman form tambah status yang akan ditampilkan apabila aktor menekan tombol 'Tambah'.

Nama Status	Keterangan	Aksi
Kontak Erat	Kontak erat adalah kondisi ketika seseorang melakukan kontak dengan orang yang termasuk ke dalam kategori konfirmasi dan probable, baik kontak fisik secara langsung, bertatap muka dengan jarak kurang dari 1 meter setidaknya selama 15 menit, atau merawat orang dengan status konfirmasi dan probable.	Edit Delete
Pelaku Perjalanan	orang yang pernah melakukan perjalanan dari dalam maupun luar negeri selama rentang waktu 14 hari terakhir.	Edit Delete
Discarded	orang berstatus kasus suspek dengan hasil pemeriksaan RT-PCR 2 kali negatif selama dua hari berturut-turut dengan selang waktu lebih dari 24 jam. Ini termasuk juga seseorang dengan status kontak erat yang telah menyelesaikan masa karantina selama 14 hari.	Edit Delete
Kasus Konfirmasi	Maksud istilah ini adalah seseorang yang dinyatakan positif terinfeksi virus Covid-19 setelah dibuktikan dengan pemeriksaan PCR/swab test di laboratorium secara real time. Kasus konfirmasi dibagi menjadi dua, yaitu konfirmasi dengan gejala (kasus simptomatik) dan konfirmasi tanpa gejala (kasus asimtomatik).	Edit Delete
Kasus Probable	Kasus probable adalah orang yang masih dalam kategori suspek dan memiliki gejala ISPA berat, gagal napas, atau meninggal dunia, namun belum ada hasil pemeriksaan yang memastikan bahwa dirinya positif COVID-19.	Edit Delete
Kasus Suspek	Kasus suspek adalah definisi baru yang menggantikan istilah PDP. Kasus suspek memiliki kriteria yang pertama yaitu Orang dengan infeksi saluran pernapasan akut (ISPA) dan pada 14 hari terakhir sebelum muncul gejala memiliki riwayat perjalanan atau tinggal di negara/wilayah Indonesia yang tercatat ada penyebaran lokal. Kedua, Orang dengan salah satu gejala/tanda ISPA dan pada 14 hari terakhir sebelum timbul gejala memiliki riwayat kontak dengan kasus konfirmasi/probable Covid-19. Ketiga, Orang dengan ISPA berat/pneumonia berat yang membutuhkan perawatan di rumah sakit dan tidak ada penyebab lain berdasarkan gambaran klinis yang meyakinkan.	Edit Delete
Kematian	Orang-orang dalam kategori kasus konfirmasi atau probable COVID-19 yang meninggal.	Edit Delete
Selesai Isolasi	Kasus konfirmasi asimtomatik atau simptomatik yang menyelesaikan karantina mandiri dan tidak lagi menunjukkan gejala demam dan gangguan pernapasan. Kasus konfirmasi asimtomatik yang tidak dilakukan pemeriksaan follow up RT-PCR dengan ditambah 10 hari isolasi mandiri sejak pengambilan spesimen diagnosis konfirmasi. Kasus probable atau kasus konfirmasi bergejala (simptomatik) yang tidak dilakukan pemeriksaan follow up RT-PCR dihitung 10 hari sejak tanggal onset dan ditambah minimal tiga hari isolasi setelah tidak lagi menunjukkan gejala demam dan gangguan pernapasan. Kasus probable atau kasus konfirmasi simptomatik	Edit Delete

Gambar 7. 6 User Interface Halaman status

Tambah Status Pasien

*Nama Status

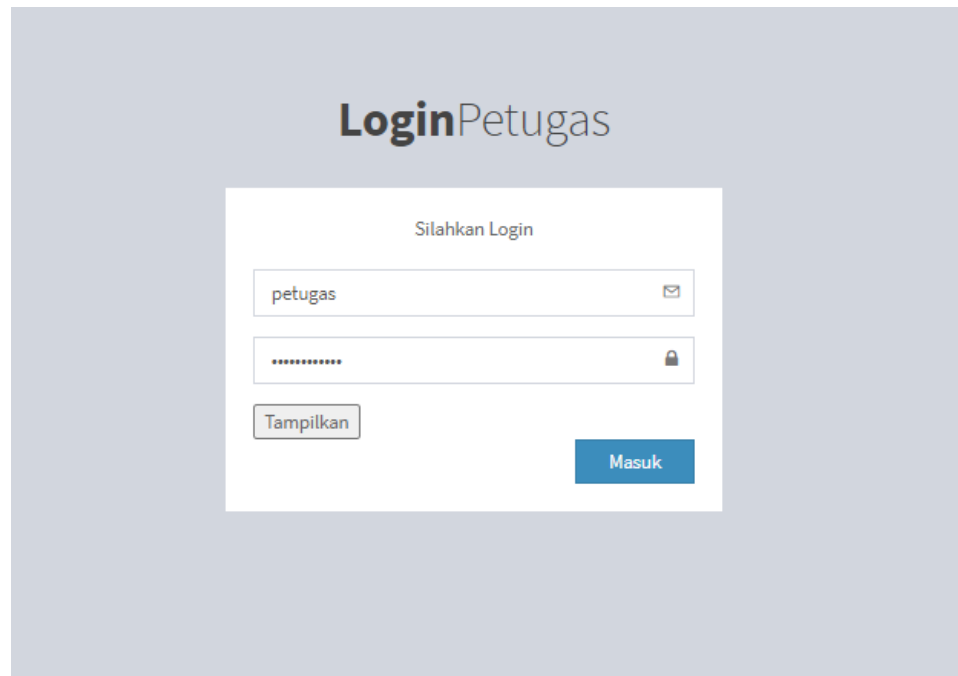
Keterangan

[Simpan](#)

Gambar 7. 7 User Interface tambah status

7.1.5 User Interface Login Petugas

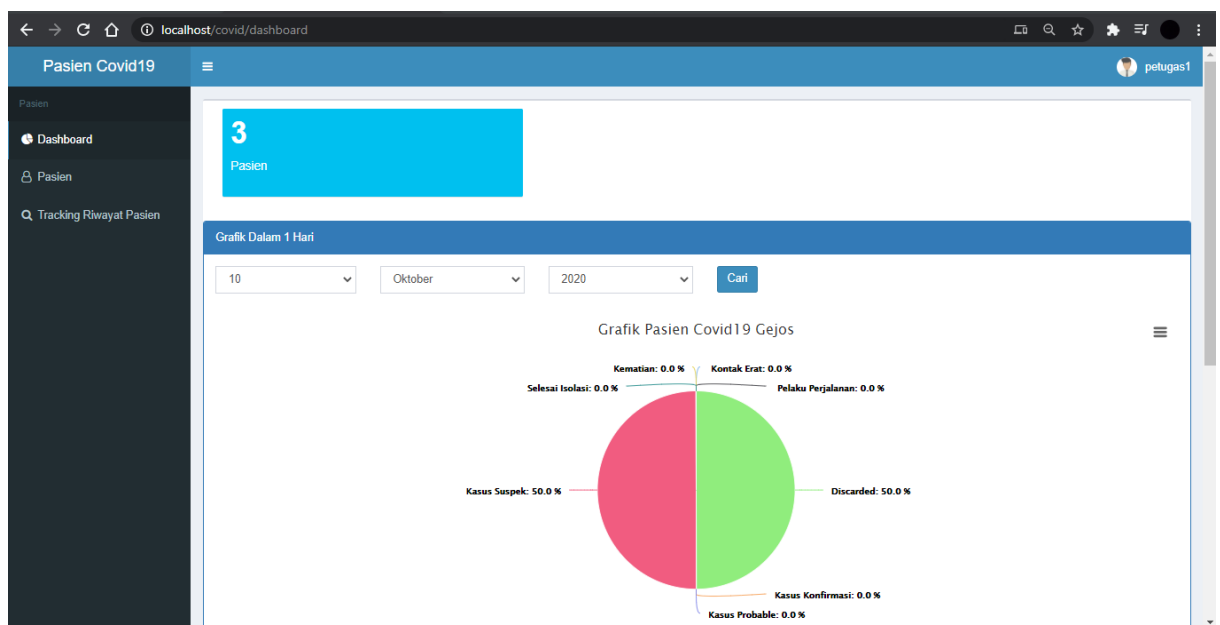
Gambar 7.8 menunjukkan implementasi dari *user interface* Login Petugas pada aktor Petugas. Implementasi *user interface* ini mengacu pada perancangan *user interface* Login Petugas.



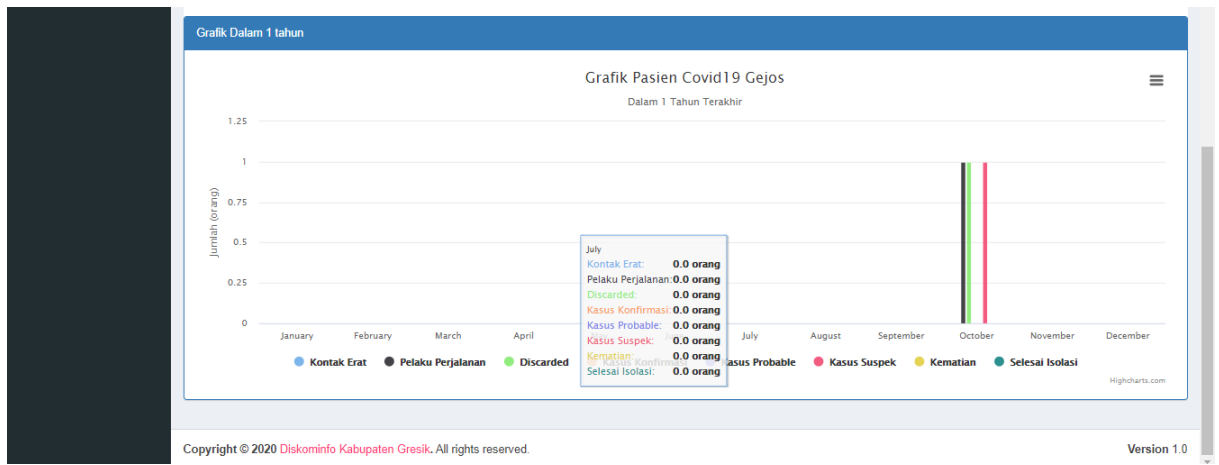
Gambar 7. 8 User Interface Login Petugas

7.1.6 User Interface Halaman Dashboard Petugas (Grafik)

Gambar 7.9 menunjukkan implementasi dari *user interface* halaman *dashboard* petugas dimana halaman ini adalah pemantauan pasien berupa grafik pada aktor petugas. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman dashboard petugas.



Gambar 7. 9 User Interface Halaman Dashboard Petugas (Grafik harian)



Gambar 7. 10 User Interface Halaman Dashboard Petugas (Grafik bulanan)

7.1.7 User Interface Halaman Daftar Pasien

Pada Gambar 7.11 menunjukkan halaman daftar pasien. Kemudian pada Gambar 7.12 menunjukkan halaman form tambah pasien yang akan ditampilkan apabila aktor menekan tombol 'Tambah'.

Pasien Covid19

Pasien

Dashboard

Pasien

Tracking Riwayat Pasien

Pasien

Daftar Status Tambah

Show 10 entries

Search:

Status	Biodata Pasien	Penempatan	Aksi
Kasus Suspek Ubah Status	No Ktp: 3573055310990003 Nama: VELIA WILDA IFANAH Detail	Hijau	Ubah Data Keluas
Discarded Ubah Status	No Ktp: 3573055310990002 Nama: CHAIRUNISA DWINANDA ASTI Detail	Hijau	Ubah Data Keluas
Pelaku Perjalanan Ubah Status	No Ktp: 3573055310990001 Nama: DEN AYU SAKINAH Detail	Hijau	Ubah Data Keluas

Showing 1 to 3 of 3 entries

Previous 1 Next

Copyright © 2020 Diskominfo Kabupaten Gresik. All rights reserved. Version 1.0

Gambar 7. 11 User Interface Halaman Daftar Pasien

Pasien Covid19

Pasien

- Dashboard
- Pasien
- Tracking Riwayat Pasien

Tambah Pasien

Biodata Pasien

*No KTP

Status

Kontak Erat

*Nama Lengkap

*Jenis Kelamin

Laki - Laki

Tempat Lahir

Tanggal Lahir

mm/dd/yyyy

Gol Darah

Agama

No Akta Lhr

Status Kawin

Cari

Daftar Status

Gambar 7. 12 User Interface Halaman tambah Pasien

7.1.8 User Interface Halaman Tracking riwayat pasien

Gambar 7.13 menunjukkan implementasi dari *user interface* halaman tracking riwayat pasien aktor petugas. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman tracking riwayat pasien.

Pasien Covid19

Pasien

- Dashboard
- Pasien
- Tracking Riwayat Pasien

Tracking Riwayat Pasien

*No KTP

3573055310990001

Cari

Tanggal	Status	No KTP	Nama Lengkap	Catatan
2020-10-07 22:29:12	Pelaku Perjalanan	3573055310990001	DEN AYU SAKINAH	

Copyright © 2020 Diskominfo Kabupaten Gresik. All rights reserved. Version 1.0

Gambar 7. 13 User Interface Halaman Tracking riwayat pasien

7.2 Implementasi Database

Database Management System yang digunakan pada Sistem Informasi Pencatatan COVID-19 (SIPCOP) adalah MySQL dengan menggunakan tools PHPMYADMIN agar memudahkan proses pengembangan *database*. Gambar implementasi *database* akan ditampilkan pada Gambar 7.14

Server: 127.0.0.1 > Basis data: covid

Struktur SQL Cari Kueri Ekspor Impor Operasi Hak Akses Routine Event Trigger

Filters

Mengandung kata:

Tabel	Tindakan	Baris	Jenis	Penyortiran	Ukuran	Beban
<input type="checkbox"/> admin	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	1	InnoDB	latin1_swedish_ci	32.0 KB	-
<input type="checkbox"/> log_riwayat_pasien	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	22	InnoDB	latin1_swedish_ci	16.0 KB	-
<input type="checkbox"/> pasien	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	3	InnoDB	latin1_swedish_ci	48.0 KB	-
<input type="checkbox"/> petugas	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	6	InnoDB	latin1_swedish_ci	16.0 KB	-
<input type="checkbox"/> ruang_rawat	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	3	InnoDB	latin1_swedish_ci	16.0 KB	-
<input type="checkbox"/> status_pasien	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	8	InnoDB	latin1_swedish_ci	16.0 KB	-
<input type="checkbox"/> tracking_status	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	3	InnoDB	latin1_swedish_ci	48.0 KB	-
7 tabel	Jumlah	46	InnoDB	utf8mb4_general_ci	192.0 KB	0 B

Gambar 7. 14 Database pada Sistem Informasi Pencatatan COVID-19

Database pada gambar ini adalah *database* yang dibuat dalam pengembangan Sistem Informasi Pencatatan COVID-19 (SIPCOP) dengan permintaan langsung dari pihak Diskominfo Kabupaten Gresik.

7.2.1 Implementasi Table Admin

Tabel Admin dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel admin pada *Database* Sistem Informasi Pencatatan COVID-19 (SIPCOP) pada Tabel 7.1.

Tabel 7. 1 Kode Program membuat tabel admin

admin.sql	
1	CREATE TABLE `admin` (
2	`id_admin` varchar(32) NOT NULL,
3	`nama_petugas` varchar(150) NOT NULL,
4	`username` varchar(50) NOT NULL,
5	`password` varchar(128) NOT NULL,
6	`deskripsi` varchar(50) NOT NULL,
7	`aktif` tinyint(4) NOT NULL,
8	`id_petugas` varchar (32) NOT NULL,
9	`created_at` datetime DEFAULT NULL,
10	`updated_at` datetime DEFAULT NULL
11)

7.2.2 Implementasi Tabel log riwayat pasien

Tabel log riwayat pasien dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel log riwayat pasien pada *Database* Sistem Informasi Pencatatan COVID-19 (SIPCOP) pada Tabel 7.2.

Tabel 7. 2 Kode Program membuat tabel log riwayat pasien

Log_riwayat_pasien.sql

```

1 CREATE TABLE `log_riwayat_pasien` (
2   `id_riwayat` varchar(32) NOT NULL,
3   `id_status` varchar(32) NOT NULL,
4   `ktp_pasien` char(16) NOT NULL,
5   `tanggal` datetime NOT NULL,
6   `catatan` varchar(100) NOT NULL,
7   `nama_lgkp` varchar(200) NOT NULL,
8   `nama_status` int(25) NOT NULL
9 )

```

7.2.3 Implementasi Tabel pasien

Tabel Pasien dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel pasien pada *Database* Sistem Informasi Pencatatan COVID-19 (SIPCOP) pada Tabel 7.3.

Tabel 7. 3 Kode Program membuat tabel pasien

pasien.sql

```

1 CREATE TABLE `pasien` (
2   `nik` char(16) NOT NULL,
3   `no_kk` char(16) NOT NULL,
4   `nama_lgkp` varchar(200) NOT NULL,
5   `tmpt_lhr` varchar(50) NOT NULL,
6   `tgl_lhr` date NOT NULL,
7   `jenis_klmin` char(1) NOT NULL,
8   `gol_darah` varchar(15) NOT NULL,
9   `agama` varchar(25) NOT NULL,
10  `no_akta_lhr` varchar(16) NOT NULL,
11  `status_kawin` varchar(25) NOT NULL,
12  `no_akta_kwn` varchar(16) NOT NULL,
13  `tgl_kwn` date NOT NULL,
14  `hub_kel` varchar(50) NOT NULL,
15  `pendidikan` varchar(50) NOT NULL,
16  `pekerjaan` varchar(50) NOT NULL,
17  `nama_lgkp_ibu` varchar(150) NOT NULL,
18  `nama_lgkp_ayah` varchar(150) NOT NULL,
19  `nama_kel` varchar(50) NOT NULL,
20  `nama_kec` varchar(50) NOT NULL,
21  `nama_kabupaten` varchar(50) NOT NULL,
22  `alamat` varchar(150) NOT NULL,
23  `no_rt` varchar(5) NOT NULL,
24  `no_rw` varchar(5) NOT NULL,
25  `id_ruang` varchar(32) NOT NULL,
26  `id_petugas` varchar(32) NOT NULL,
27  `created_at` datetime DEFAULT NULL,
28  `updated_at` datetime DEFAULT NULL
29 )

```

7.2.4 Implementasi Tabel Petugas

Tabel petugas dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel petugas pada *Database* Sistem Informasi Pencatatan COVID-19 (SIPCOP) pada Tabel 7.4.

Tabel 7. 4 Kode Program membuat tabel petugas

petugas.sql

1	CREATE TABLE `petugas` (
2	`id_petugas` varchar(32) NOT NULL,
3	`nama_petugas` varchar(150) NOT NULL,
4	`username` varchar(50) NOT NULL,
5	`password` varchar(128) NOT NULL,
6	`deskripsi` varchar(50) NOT NULL,
7	`aktif` tinyint(4) NOT NULL,
8	`created_by` varchar(64) NOT NULL,
9	`modified_by` varchar(64) NOT NULL,
10	`created_at` datetime DEFAULT NULL,
11	`updated_at` datetime DEFAULT NULL
)

7.2.5 Implementasi Tabel Ruang rawat

Tabel ruang rawat dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel ruang rawat pada *Database Sistem Informasi Pencatatan COVID-19 (SIPCOP)* pada Tabel 7.5.

Tabel 7. 5 Kode Program membuat tabel ruang rawat

Ruang_rawat.sql	
1	CREATE TABLE `ruang_rawat` (
2	`id_ruang` varchar(32) NOT NULL,
3	`zona_ruang` varchar(50) NOT NULL,
4	`keterangan_ruang` varchar(25) NOT NULL,
5	`created_at` datetime DEFAULT NULL,
6	`updated_at` datetime DEFAULT NULL
7)

7.2.6 Implementasi Tabel Status pasien

Tabel status pasien dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel status pasien pada *Database Sistem Informasi Pencatatan COVID-19 (SIPCOP)* pada Tabel 7.6.

Tabel 7. 6 Kode Program membuat tabel status pasien

Status_pasien.sql	
1	CREATE TABLE `status_pasien` (
2	`id_status` varchar(32) NOT NULL,
3	`nama_status` varchar(25) NOT NULL,
4	`keterangan_status` varchar(500) NOT NULL,
5	`created_at` datetime DEFAULT NULL,
6	`updated_at` datetime DEFAULT NULL
7)
8	

7.2.7 Implementasi Tabel tracking riwayat

Tabel tracking riwayat dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel tracking riwayat pada *Database Sistem Informasi Pencatatan COVID-19 (SIPCOP)* pada Tabel 7.7.

Tabel 7. 7 Kode Program membuat tabel tracking riwayat

Tracking_status.sql	
1	CREATE TABLE `tracking_status` (
2	`id_status` varchar(32) NOT NULL,
3	`ktp_pasien` char(16) NOT NULL,
4	`tanggal` datetime NOT NULL,
5	`catatan` varchar(100) NOT NULL
6)

7.3. Implementasi Kode Program

Implementasi kode program merupakan tahap penulisan kode program dengan bahasa pemrograman berdasarkan *class diagram* yang telah dirancang. Bahasa pemrograman yang digunakan adalah *PHP*. Pada bagian ini, pembahasan kode program meliputi fungsi-fungsi yang digunakan oleh actor admin dan petugas.

7.3.1. Implementasi *Login Admin*

Hasil implementasi kode program untuk proses *login* Admin ditunjukkan pada tabel 7.8. Pada Controller dengan nama *Authen_super* terdapat *Method* *index* yang akan melakukan seleksi kondisi saat user memasukkan *username* dan *password* untuk dicek pada *database*. Seleksi kondisi pertama yaitu *method* *is_registered*. Jika *username* dan *password* tersedia pada *database*, kemudian berhasil masuk dan dilanjutkan *method* *login_success* dengan ditampilkannya data admin. Namun Jika *username* dan *password* tidak tersedia akan mengarah pada *method* *login_failed* yang akan menampilkan pesan gagal. Selesi kondisi kedua yaitu *is_section_active* yang memastikan bahwa akun admin dalam kondisi aktif, sehingga dapat Sukses login pada halaman dashboard sistem.

Tabel 7. 8 Kode Program Controller Login Admin

Authen_super.php	
1	public function index(){
2	\$this->form_validation->set_rules('username', 'Username',
3	'required');
4	\$this->form_validation->set_rules('password', 'Password',
5	'required');
6	if(\$this->form_validation->run()){
7	
8	\$username= \$this->input->post('username');
9	\$password = \$this->input->post('password');
10	if(\$this->is_registered(\$username,\$password)){
11	\$this->login_success(\$username);
12	}else{
13	\$this->login_failed();
14	}
15	}else{
16	if(\$this->is_section_active()){
17	\$this->login_success(\$this->session->
18	>userdata(SESSIONDATA_LOGIN_ADMIN_USERNAME));

```

19         }
20         $this->load->view('login_super');
21     }
22 }
23 private function is_section_active() {
24     if($this->session-
25 >has_userdata(SESSIONDATA_LOGIN_ADMIN_ID)) {
26         return true;
27     }
28     return false;
29 }
30
31 private function is_registered($user,$pass){
32     $user = $this->Admin_model->get_admin($user);
33
34     //print_r($user);
35     if($user){
36         if(password_verify($pass,$user['password'])&&
37 $user['aktif']=='1') {
38             return true;
39         }
40     }
41     return false;
42 }
43
44 private function login_success($username){
45     $user = $this->Admin_model->get_admin($username);
46     $newdata = [
47         SESSIONDATA_LOGIN_ADMIN_ID => $user['id_petugas'],
48         SESSIONDATA_LOGIN_ADMIN_USERNAME => $username,
49         SESSIONDATA_LOGIN_ADMIN_NAMA => $user['nama_petugas'],
50         SESSIONDATA_LOGIN_ADMIN_DESKRIPSI =>
51 $user['deskripsi'],
52         SESSIONDATA_LOGIN_ADMIN_LOGGEDIN => TRUE
53     ];
54     $this->session->set_userdata($newdata);
55
56
57     redirect('petugas');
58     //redirect('slider/index');
59 }
60 private function login_failed(){
61     $data['_view']='login';
62     $this->session-
63 >set_flashdata(FLASHDATA_PATH_ALERTS,'alerts/login/error');
64     $this->load->view('login_super',$data);
65 }
66 public function logout(){
67     $this->session->sess_destroy();
68
69     redirect('authen_super/index');
70 }

```

Tabel 7. 9 Kode Program Model Login Admin

Admin_model.php	
1	/*

```

2      * Get admin by
3      */
4      function get_admin($username)
5      {
6          return $this->db->get_where('admin', array('username'=>$username))->row_array();
7      }
8
9
10     /*
11     * Get all admin
12     */
13     function get_all_admin()
14     {
15         $this->db->order_by('', 'desc');
16         return $this->db->get('admin')->result_array();
17     }
18     /*
19     * function to add new admin
20     */
21     function add_admin($params)
22     {
23         $this->db->insert('admin', $params);
24         return $this->db->insert_id();
25     }
26
27     /*
28     * function to update admin
29     */
30     function update_admin($username, $params)
31     {
32         $this->db->where('email', $username);
33         return $this->db->update('admin', $params);
34     }
35
36     /*
37     * function to delete admin
38     */
39     function delete_admin($username)
40     {
41         return $this->db->delete('admin', array('email'=>$username));
42     }
43

```

7.3.2. Implementasi *Login* Petugas

Hasil implementasi kode program untuk proses *login* Petugas ditunjukkan pada tabel 7.10. Pada Controller dengan nama Authen terdapat *Method* index yang akan melakukan seleksi kondisi saat user memasukkan *username* dan *password* untuk dicek pada *database*. Seleksi kondisi pertama yaitu method *is_registered* Jika *username* dan *password* tersedia pada *database*, kemudian berhasil masuk dan dilanjutkan *method* *login_success* dengan ditampilkannya data petugas. Namun Jika *username* dan *password* tidak tersedia akan mengarah pada *method* *login_failed* yang akan menampilkan pesan gagal. Selesi kondisi kedua yaitu *is_section_active* yang memastikan bahwa akun petugas dalam kondisi aktif, sehingga dapat Sukses login pada halaman dashboard sistem.

Tabel 7. 10 Kode Program Controller Login Petugas

Authen.php	
1	public function index(){
2	\$this->form_validation-
3	>set_rules('username','Username',required');
4	\$this->form_validation-
5	>set_rules('password','Password','required');
6	if(\$this->form_validation->run()){
7	
8	\$username= \$this->input->post('username');
9	\$password = \$this->input->post('password');
10	if(\$this->is_registered(\$username,\$password)){
11	\$this->login_success(\$username);
12	}else{
13	\$this->login_failed();
14	}
15	}else{
16	if(\$this->is_section_active()){
17	\$this->login_success(\$this->session-
18	>userdata(SESSIONDATA_LOGIN_PETUGAS_USERNAME));
19	}
20	\$this->load->view('login');
21	}
22	}
23	private function is_section_active(){
24	if(\$this->session-
25	>has_userdata(SESSIONDATA_LOGIN_PETUGAS_ID)){
26	return true;
27	}
28	return false;
29	}
30	
31	private function is_registered(\$user,\$pass){
32	\$user = \$this->Petugas_model-
33	>get_petugas_byusername(\$user);
34	
35	//print_r(\$user);
36	if(\$user){
37	if(password_verify(\$pass,\$user['password'])&&
38	\$user['aktif']=='1') {
39	return true;
40	}
41	}
42	return false;
43	}
44	
45	private function login_success(\$username){
46	\$user = \$this->Petugas_model-
47	>get_petugas_byusername(\$username);
48	\$newdata = [
49	SESSIONDATA_LOGIN_PETUGAS_ID => \$user['id_petugas'],
50	SESSIONDATA_LOGIN_PETUGAS_USERNAME => \$username,
51	SESSIONDATA_LOGIN_PETUGAS_PETUGAS =>
52	\$user['nama_petugas'],
53	SESSIONDATA_LOGIN_PETUGAS_DESKRIPSI =>
54	\$user['deskripsi'],
55	SESSIONDATA_LOGIN_PETUGAS_LOGGEDIN => TRUE

56];
57	\$this->session->set_userdata(\$newdata);
58	
59	
60	redirect('dashboard/index');
61	}
62	private function login_failed(){
63	\$this->session-
64	>set_flashdata(FLASHDATA_PATH_ALERTS,'alerts/login/error');
65	\$this->load->view('login');
	}
	public function logout(){
	\$this->session->sess_destroy();
	redirect('authen/index');
	}

Tabel 7. 11 Kode Program Model Login Petugas

Petugas_model.php	
1	/*
2	* Get petugas by
3	*/
4	function get_petugas_byusername(\$username)
5	{
6	return \$this->db-
7	>get_where('petugas',array('username'=>\$username))-
8	>row_array();
9	}
10	
11	/*
12	* Get petugas by
13	*/
14	function get_petugas(\$id)
15	{
16	return \$this->db-
17	>get_where('petugas',array('id_petugas'=>\$id))->row_array();
18	}
19	
20	/*
21	* Get all petugas
22	*/
23	function get_all_petugas()
24	{
25	\$this->db->order_by('','desc');
26	return \$this->db->get('petugas')->result_array();
27	}
28	
29	/*
30	* function to add new petugas
31	*/
32	function add_petugas(\$params)
33	{
34	return \$this->db->insert('petugas',\$params);
35	}
36	
37	/*

38	* function to update petugas
39	*/
40	function update_petugas(\$id,\$params)
41	{
42	\$this->db->where('id_petugas',\$id);
43	return \$this->db->update('petugas',\$params);
44	}
45	
46	/*
47	* function to delete petugas
48	*/
49	function delete_petugas(\$id)
50	{
51	return \$this->db->delete('petugas',array('id_petugas'=>\$id));
	}

7.3.3. Implementasi *Dashboard*

Hasil implementasi kode program untuk proses menampilkan Dashboard ditunjukkan pada tabel 7.12. Terdapat method index yang akan memanggil method prepare_grafik_harian untuk menampilkan diagram lingkaran yang memvisualisasi jumlah pasien pada hari tersebut dengan memasukkan parameter tanggal, bulan dan tahun. Dan memanggil method prepare_grafik_tahunan untuk menampilkan grafik 1 tahun terakhir dengan detail bulanan. Data yang ditampilkan merupakan data dari database Pasien_model, Tracking_status_model, dan Status_pasien_model.

Tabel 7. 12 Kode Program Controller dashboard

Dashboard.php	
1	function index()
2	{
3	//-----Grafik Tahunan
4	\$array_bulan_string=array();
5	\$array_bulan = [1,2,3,4,5,6,7,8,9,10,11,12];
6	foreach (\$array_bulan as \$bulan) {
7	array_push(\$array_bulan_string, date('F', mktime(0, 0, 0,
8	\$bulan, 10))) ;
9	}
10	\$data['sumbux_grafik_line'] = \$array_bulan_string;
11	\$data['data_grafik_tahunan'] = \$this->
12	prepare_grafik_tahunan();
13	
14	//-----Grafik Harian
15	if(isset(\$_POST) && count(\$_POST) > 0){
16	\$tahun = \$this->input->post('tahun');
17	\$bulan = \$this->input->post('bulan');
18	\$tgl = \$this->input->post('tgl');
19	\$data['data_grafik_harian'] = \$this->
20	prepare_grafik_harian(\$tgl,\$bulan,\$tahun);
21	}else{
22	\$data['data_grafik_harian'] = array();
23	}
24	\$data['total_pasien'] = \$this->Pasien_model->count_all();

```

25     $data['tanggal_sekarang'] = date('d', time());
26     $data['bulan_sekarang'] = date('m', time());
27     $data['tahun_sekarang'] = date('Y', time());
28     $data['_view'] = 'home/dashboard';
29     $this->load->view('home/layouts/main', $data);
30 }
31
32 function prepare_grafik_harian($tgl, $bulan, $tahun){
33     $tahun_sekarang = date('Y', time());
34     $bulan_sekarang = date('m', time());
35     $array_bulan = [1,2,3,4,5,6,7,8,9,10,11,12];
36     $array_bulan_string=array();
37     $array_status      =          $this->Status_pasien_model-
38 >get_all_status_pasien();
39     $arrayresult_harian = array();
40
41     foreach($array_status as $item_status){
42         $array_jumlahdata = [];
43
44         $result          =          $this->Tracking_status_model-
45 >get_tracking_statu(
46             array(
47                 'day(tanggal)' => $tgl,
48                 'month(tanggal)' => $bulan,
49                 'year(tanggal)' => $tahun,
50                 'tracking_status.id_status'
51 =>$item_status['id_status']
52             )
53         );
54         $jumlahdata = count($result);
55         $array_obj= array(
56             'name' => $item_status['nama_status'],
57             'y' => (int)$jumlahdata
58         );
59         array_push($arrayresult_harian, $array_obj);
60     }
61
62     return $arrayresult_harian;
63 }
64
65 function prepare_grafik_tahunan(){
66     $tahun_sekarang = date('Y', time());
67     $bulan_sekarang = date('m', time());
68     $array_bulan = [1,2,3,4,5,6,7,8,9,10,11,12];
69     $array_bulan_string=array();
70     $array_status      =          $this->Status_pasien_model-
71 >get_all_status_pasien();
72     $arrayresult = array();
73     //*****ALGORITMA          UNTUK          GRAFIK
74 LINE*****
75     foreach ($array_bulan as $bulan) {
76         array_push($array_bulan_string, date('F', mktime(0, 0, 0,
77 $bulan, 10))); ;
78     }
79
80     foreach($array_status as $item_status){
81         $array_jumlahdata = [];
82         for($j =0 ; $j< count($array_bulan); $j++){

```



```

83         $result          =          $this->Tracking_status_model-
84 >get_tracking_statu(
85         array(
86             'month(tanggal)' => $array_bulan[$j],
87             'year(tanggal)' => $tahun_sekarang,
88             'tracking_status.id_status'
89 =>$item_status['id_status']
90         )
91     );
92     $jumlahdata = count($result);
93     array_push($array_jumlahdata, (int)$jumlahdata);
94 }
95
96     $array_obj= array(
97         'name' => $item_status['nama_status'],
98         'data' => $array_jumlahdata
99     );
100     array_push($arrayresult, $array_obj);
101 }
102
103     return $arrayresult;
1-4 }
}

```

7.3.4. Implementasi Tracking riwayat Pasien

Pada tabel 7.13. ditunjukkan tabel Model dari tracking riwayat pasien yang akan digunakan pada Controller Pasien yang dipanggil untuk dapat dilakukan tracking pasien dari data log riwayat pasien tersebut.

Tabel 7. 13 Kode Program Model tracking riwayat pasien

Tracking_status_model.php	
1	<?php
2	
3	class Tracking_status_model extends CI_Model
4	{
5	function __construct()
6	{
7	parent::__construct();
8	}
9	
10	
11	/*
12	* Get tracking_status by
13	*/
14	function get_tracking_statu(\$array)
15	{
16	
17	\$this->db->select('tracking_status.*');
18	\$this->db->select('status_pasien.nama_status');
19	\$this->db->select('pasien.nama_lgkp');
20	\$this->db->
21	>join('status_pasien','tracking_status.id_status=status_pasien.id
22	_status');
23	\$this->db->
24	>join('pasien','tracking_status.ktp_pasien=pasien.nik');
25	\$this->db->order_by('tanggal','ASC');
26	return \$this->db->get_where('tracking_status',\$array)-
27	>result_array();

```

28     }
29
30     /*
31     * Get all tracking_status
32     */
33     function get_all_tracking_status()
34     {
35         $this->db->order_by('', 'desc');
36         return $this->db->get('tracking_status')->result_array();
37     }
38
39
40     /*
41     * function to add new tracking_statu
42     */
43     function add_tracking_statu($params)
44     {
45         return $this->db->insert('tracking_status', $params);
46     }
47
48
49     /*
50     * function to update tracking_statu
51     */
52     function update_tracking_statu($array, $params)
53     {
54         $this->db->where($array);
55         return $this->db->update('tracking_status', $params);
56     }
57
58
59     /*
60     * function to delete tracking_statu
61     */
62     function delete_tracking_statu($array)
63     {
64         return $this->db->delete('tracking_status', $array);
65     }
66
67
68     // function get_report_kunjungan_permonth_bynama($bulan,$tahun,
69     $nama){
70         // $query = "select COUNT(*) as jumlah, lengkap.nama FROM (SE
71     LECT tujuan.nama,kunjungan.tanggal FROM `kunjungan` JOIN tujuan O
72     N kunjungan.id_tujuan = tujuan.id) as lengkap WHERE year(tanggal)
73     =".$tahun." " AND month(tanggal)=".$bulan." AND lengkap.nama='".$ $
     nama."'";
74         //
75         // $data = $this->db->query($query);
76         // return $data->result_array();
77         // }
78     }

```

7.3.5. Implementasi Mengelola Pasien

Hasil implementasi kode program untuk proses mengelola pasien ditunjukkan pada tabel 7.14. Pada Controller dengan nama Pasien ini terdapat method-method yang berfungsi dalam mengelola data pasien yaitu Menampilkan detail data pasien, Menambahkan data pasien, melakukan edit data pasien, dan menghapus data pasien yang akan diteruskan pada database pasien.

Tabel 7. 14 Kode Program Controller Pasien

Pasien.php	
1	function index()
2	{
3	\$data['pasien'] = \$this->Pasien_model->get_all_pasien();
4	\$data['status'] = \$this->Status_pasien_model->
5	get_all_status_pasien();
6	\$data['_view'] = 'home/pasien/index';
7	\$this->load->view('home/layouts/main',\$data);
8	}
9	/*
10	* Detail a pasien
11	*/
12	function detail(\$nik)
13	{
14	// check if the pasien exists before trying to edit it
15	\$data['pasien'] = \$this->Pasien_model->get_pasien(\$nik);
16	
17	if(isset(\$data['pasien']['nik']))
18	{
19	\$daftarstatus=\$this->Tracking_status_model->
20	get_tracking_statu(array('ktp_pasien'=>\$nik));
21	\$data['statussekarang']=end(\$daftarstatus)['id_status'];
22	\$data['kecamatan']=\$this->array_kecamatan;
23	\$data['kabupaten']=\$this->array_kabupaten;
24	\$data['status']=\$this->Status_pasien_model->
25	get_all_status_pasien();
26	\$data['ruang_rawat']=\$this->Ruang_rawat_model->
27	get_all_ruang_rawat();
28	\$data['_view'] = 'home/pasien/detail';
29	\$this->load->view('home/layouts/main',\$data);
30	}
31	else
32	show_error('The pasien you are trying to edit does not
33	exist.');
34	}
35	
36	/*
37	* Adding a new pasien
38	*/
39	function add()
40	{
41	\$this->load->library('form_validation');
42	
43	\$this->form_validation->set_rules('nik','No
44	KTP','required numeric');
45	\$this->form_validation->set_rules('no_kk','No
46	Kk','required numeric');
47	\$this->form_validation->set_rules('nama_lgkp','Nama
48	Lgkp','required');
49	\$this->form_validation->set_rules('jenis_klmin','Jenis
50	Klmin','required');
51	\$this->form_validation->set_rules('hub_kel','Hub
52	Kel','required');
53	\$this->form_validation->set_rules('nama_lgkp_ibu','Nama
54	Lgkp Ibu','required');
55	

```

56     $this->form_validation->set_rules('nama_lgkp_ayah','Nama
57     Lgkp Ayah','required');
58     $this->form_validation->set_rules('nama_kel','Nama
59     Kel','required');
60     $this->form_validation->set_rules('nama_kec','Nama
61     Kec','required');
62     $this->form_validation-
63     >set_rules('alamat','Alamat','required');
64     $this->form_validation->set_rules('no_rt','No
65     Rt','required');
66     $this->form_validation->set_rules('no_rw','No
67     Rw','required');
68
69     if($this->form_validation->run())
70     {
71         $params = array(
72             'nik' => $this->input->post('nik'),
73             'jenis_klmin' => $this->input->post('jenis_klmin'),
74             'no_kk' => $this->input->post('no_kk'),
75             'nama_lgkp' => $this->input->post('nama_lgkp'),
76             'tmpt_lhr' => $this->input->post('tmpt_lhr'),
77             'tgl_lhr' => $this->input->post('tgl_lhr'),
78             'gol_darah' => $this->input->post('gol_darah'),
79             'agama' => $this->input->post('agama'),
80             'no_akta_lhr' => $this->input->post('no_akta_lhr'),
81             'status_kawin' => $this->input->post('status_kawin'),
82             'no_akta_kwn' => $this->input->post('no_akta_kwn'),
83             'tgl_kwn' => $this->input->post('tgl_kwn'),
84             'hub_kel' => $this->input->post('hub_kel'),
85             'pendidikan' => $this->input->post('pendidikan'),
86             'pekerjaan' => $this->input->post('pekerjaan'),
87             'nama_lgkp_ibu' => $this->input->post('nama_lgkp_ibu'),
88             'nama_lgkp_ayah' => $this->input-
89 >post('nama_lgkp_ayah'),
90             'nama_kel' => $this->input->post('nama_kel'),
91             'nama_kec' => $this->input->post('nama_kec'),
92             'alamat' => $this->input->post('alamat'),
93             'no_rt' => $this->input->post('no_rt'),
94             'no_rw' => $this->input->post('no_rw'),
95             'id_ruang' => $this->input->post('id_ruang'),
96             'id_petugas' => $this->session-
97 >userdata(SESSIONDATA_LOGIN_PETUGAS_ID),
98             'created_at' => date("Y-m-d H:i:s"),
99             'updated_at' => date("Y-m-d H:i:s"),
100         );
101         $params_tracking = array(
102             'ktp_pasien' => $this->input->post('nik'),
103             'id_status' => $this->input->post('id_status'),
104             'tanggal' => $params['created_at'],
105         );
106         // print_r($params);
107         // print_r($params_tracking);
108         $pasien_return = $this->Pasien_model-
109 >add_pasien($params);
110         $tracking_return = $this->Tracking_status_model-
111 >add_tracking_statu($params_tracking);
112         $this->session-
113 >set_flashdata(FLASHDATA_PATH_ALERTS,'alerts/add/success');

```

```

114         redirect('pasien/index');
115     }
116     else
117     {
118         $data['kecamatan']=$this->array_kecamatan;
119         $data['kabupaten']=$this->array_kabupaten;
120         $data['status']=$this->Status_pasien_model-
121 >get_all_status_pasien();
122         $data['ruang_rawat']=$this->Ruang_rawat_model-
123 >get_all_ruang_rawat();
124         $data['_view'] = 'home/pasien/add';
125         $this->load->view('home/layouts/main',$data);
126     }
127 }
128
129 /*
130 * Editing a pasien
131 */
132 function edit($nik)
133 {
134     // check if the pasien exists before trying to edit it
135     $data['pasien'] = $this->Pasien_model->get_pasien($nik);
136
137     if(isset($data['pasien']['nik']))
138     {
139         $this->load->library('form_validation');
140
141         $this->form_validation->set_rules('nik','No
142 KTP','required|numeric');
143         $this->form_validation->set_rules('no_kk','No
144 Kk','required|numeric');
145         $this->form_validation->set_rules('nama_lgkp','Nama
146 Lgkp','required');
147         $this->form_validation->set_rules('jenis_klmin','Jenis
148 Klmin','required');
149         $this->form_validation->set_rules('hub_kel','Hub
150 Kel','required');
151         $this->form_validation->set_rules('nama_lgkp_ibu','Nama
152 Lgkp Ibu','required');
153         $this->form_validation->set_rules('nama_lgkp_ayah','Nama
154 Lgkp Ayah','required');
155         $this->form_validation->set_rules('nama_kel','Nama
156 Kel','required');
157         $this->form_validation->set_rules('nama_kec','Nama
158 Kec','required');
159         $this->form_validation-
160 >set_rules('alamat','Alamat','required');
161         $this->form_validation->set_rules('no_rt','No
162 Rt','required');
163         $this->form_validation->set_rules('no_rw','No
164 Rw','required');
165
166         if($this->form_validation->run())
167         {
168             $params = array(
169                 'jenis_klmin' => $this->input->post('jenis_klmin'),
170                 'nik' => $this->input->post('nik'),
171                 'no_kk' => $this->input->post('no_kk'),

```

```

172         'nama_lgkp' => $this->input->post('nama_lgkp'),
173         'tmpt_lhr' => $this->input->post('tmpt_lhr'),
174         'tgl_lhr' => $this->input->post('tgl_lhr'),
175         'gol_darah' => $this->input->post('gol_darah'),
176         'agama' => $this->input->post('agama'),
177         'no_akta_lhr' => $this->input->post('no_akta_lhr'),
178         'status_kawin' => $this->input->post('status_kawin'),
179         'no_akta_kwn' => $this->input->post('no_akta_kwn'),
180         'tgl_kwn' => $this->input->post('tgl_kwn'),
181         'hub_kel' => $this->input->post('hub_kel'),
182         'pendidikan' => $this->input->post('pendidikan'),
183         'pekerjaan' => $this->input->post('pekerjaan'),
184         'nama_lgkp_ibu' => $this->input-
185 >post('nama_lgkp_ibu'),
186         'nama_lgkp_ayah' => $this->input-
187 >post('nama_lgkp_ayah'),
188         'nama_kel' => $this->input->post('nama_kel'),
189         'nama_kec' => $this->input->post('nama_kec'),
190         'nama_kabupaten' => $this->input-
191 >post('nama_kabupaten'),
192         'alamat' => $this->input->post('alamat'),
193         'no_rt' => $this->input->post('no_rt'),
194         'no_rw' => $this->input->post('no_rw'),
195         'id_ruang' => $this->input->post('id_ruang'),
196         'id_petugas' => $this->session-
197 >userdata(SESSIONDATA_LOGIN_PETUGAS_ID),
198         'updated_at' => date('Y-m-d H:i:s'),
199     );
200
201     $this->Pasien_model->update_pasien($nik,$params);
202     $this->session-
203 >set_flashdata(FLASHDATA_PATH_ALERTS,'alerts/edit/success');
204     redirect('pasien/index');
205 }
206 else
207 {
208     $data['kecamatan']=$this->array_kecamatan;
209     $data['kabupaten']=$this->array_kabupaten;
210     $data['status']=$this->Status_pasien_model-
211 >get_all_status_pasien();
211     $data['ruang_rawat']=$this->Ruang_rawat_model-
212 >get_all_ruang_rawat();
213     $data['_view'] = 'home/pasien/edit';
214     $this->load->view('home/layouts/main',$data);
215 }
216 }
217 else
218     show_error('The pasien you are trying to edit does not
219 exist.');
```

```

220 }
221
222 /*
223  * Editing a pasien
224  */
225 function editstatus($nik)
226 {
227     // check if the pasien exists before trying to edit it
228     $data['pasien'] = $this->Pasien_model->get_pasien($nik);

```

```

229
230     if(isset($data['pasien']['nik']))
231     {
232         $this->load->library('form_validation');
233         $this->form_validation->
234 >set_rules('id_status','Status','required');
235         if($this->form_validation->run())
236         {
237             $params = array(
238                 'id_petugas' => $this->session->
239 >userdata(SESSIONDATA_LOGIN_PETUGAS_ID),
240                 'updated_at' => date('Y-m-d H:i:s'),
241             );
242             $params_tracking = array(
243                 'ktp_pasien' => $nik,
244                 'id_status' => $this->input->post('id_status'),
245                 'catatan' => $this->input->post('catatan'),
246                 'tanggal' => $params['updated_at'],
247             );
248
249             if($this->input->post('id_status') != $this->input->
250 >post('old_status')) {
251                 $tracking_return = $this->Tracking_status_model->
252 >add_tracking_statu($params_tracking);
253                 $this->Pasien_model->update_pasien($nik,$params);
254                 $this->session->
255 >set_flashdata(FLASHDATA_PATH_ALERTS,'alerts/edit/success');
256             }
257             redirect('pasien/index');
258         }
259         else
260         {
261             $daftarstatus=$this->Tracking_status_model->
262 >get_tracking_statu(array('ktp_pasien'=>$nik));
263             $data['statussekarang']=end($daftarstatus)['id_status'];
264             $data['status']=$this->Status_pasien_model->
265 >get_all_status_pasien();
266             $data['_view'] = 'home/pasien/editstatus';
267             $this->load->view('home/layouts/main',$data);
268         }
269     }
270 }
271 else
272     show_error('The pasien you are trying to edit does not
273 exist.');
```

```

274 }
275 /*
276 * Deleting pasien
277 */
278 function remove($nik)
279 {
280     $pasien = $this->Pasien_model->get_pasien($nik);
281
282     // check if the pasien exists before trying to delete it
283     if(isset($pasien['nik']))
284     {
285         $array_tracking=array(
286             'ktp_pasien'=>$nik

```

```

287     );
288     $riwayat_pasien_old=$this->Tracking_status_model-
289 >get_tracking_statu($array_tracking);
290     foreach ($riwayat_pasien_old as $item) {
291         $params_log=array(
292             'id_riwayat' => str_replace("-", "", $this->uuid-
293 >v4()),
294             'id_status' => $item['id_status'],
295             'ktp_pasien'=> $item['ktp_pasien'],
296             'nama_status'=> $item['nama_status'],
297             'tanggal'=> $item['tanggal'],
298             'catatan'=> $item['catatan'],
299             'nama_lgkp'=> $item['nama_lgkp'],
300             'nama_status'=> $item['nama_status']
301         );
302
303         $return = $this->Log_riwayat_pasien_model-
304 >add_Log_riwayat_pasien($params_log);
305     }
306
307     $this->Tracking_status_model-
308 >delete_tracking_statu($array_tracking);
309     $this->Pasien_model->delete_pasien($nik);
310     $this->session-
311 >set_flashdata(FLASHDATA_PATH_ALERTS, 'alerts/delete/success'
312 );
313     redirect('pasien/index');
314 }
315 else
316     show_error('The pasien you are trying to delete does not
317 exist.');
```

```

318 }
319
320 function tracking(){
321     // check if the pasien exists before trying to edit it
322
323
324     $this->load->library('form_validation');
325     $this->form_validation->set_rules('nik', 'No
326 KTP', 'integer|required');
327     if($this->form_validation->run())
328     {
329         $params=array(
330             'ktp_pasien' => $this->input->post('nik')
331         );
332         $data['riwayat'] = $this->Tracking_status_model-
333 >get_tracking_statu($params);
334         $data['_view'] = 'home/pasien/trackingriwayat_pasien';
335         $this->load->view('home/layouts/main', $data);
336     }
337     else
338     {
339         $data['_view'] = 'home/pasien/trackingriwayat_pasien';
340         $this->load->view('home/layouts/main', $data);
341     }
342
343 }
344

```



```

345 // FUNCTION PROCESS
346 function api_ktp() {
347
348 $url="http://simantra.gresikkab.go.id/mantra/json/kominfo/dukcapil/dukcapil_penduduk_by_nik";
349 $postdata = $this->input->post();
350 $accesskey="shiqds3zgq"; //Kunci akses diperoleh dari
351 permohonan akses requester
352 $pardata=array("NIK"=>urlencode($postdata['nik']));
353 $par="/".http_build_query($pardata);
354 $options=array('http'=>
355 array(
356 'method'=>'GET',
357 'header'=>"User-Agent: MANTRA\r\n".
358 "AccessKey: $accesskey"
359 )
360 );
361 $context=stream_context_create($options);
362 $content=file_get_contents($url.$par,false,$context);
363 echo json_encode($content);
364 }
365
366 function get_data_pasien_json()
367 {
368
369 $list = $this->Pasien_model->get_datatables();
370 $data = array();
371 $no = $_POST['start'];
372 foreach ($list as $field) {
373 $no++;
374 $riwayat_pasien=$this->Tracking_status_model-
375 >get_tracking_statu(array('ktp_pasien'=>$field->nik));
376 $status_pasien = end($riwayat_pasien)['nama_status'];
377
378 if($status_pasien=='Selesai' ||
379 $status_pasien=='Kematian'){
380 $buttonstatus = "<a disabled href='\" . '#' . '\" .
381 \"class='\" btn btn-warning btn-xs \"><span class='\" fa fa-
382 heartbeat \"></span> Ubah Status</a>\".
383 \"<br>\" .
384 \"<small>Silahkan Klik Tombol Keluar Untuk Hapus Data
385 Pasien</small>\";
386 }else{
387 $buttonstatus = "<a href='\" . 'editstatus/'. $field-
388 >nik . '\" . \"class='\" btn btn-warning btn-xs \"><span class='\" fa fa-
389 heartbeat \"></span> Ubah Status</a>\";
390 }
391
392 $row = array();
393 $row[] = $status_pasien .
394 \"<br>\" .
395 $buttonstatus;
396 $row[] = \"<small>No Ktp</small><br>\".
397 $field->nik.\"<br>\".
398 \"<small>Nama</small><br>\".
399 $field->nama_lgkp.\"<br>\".
400
401
402

```

403	"nik .'" . "class='btn
404	btn-primary btn-xs'>
405	Detail";
406	\$row[] = \$field->zona_ruang;
407	\$row[] = "nik .'" .
408	"class='btn btn-info btn-xs'>
409	Ubah Data" .
410	" " .
411	"nik .'" . "class='btn
412	btn-danger btn-xs' onclick='return confirm(\" Apakah Anda Yakin
413	Akan Menghapus Data Ini ? \")'>
414	Keluar";
415	\$data[] = \$row;
416	}
417	
418	\$output = array(
419	"draw" => \$_POST['draw'],
420	"recordsTotal" => \$this->Pasien_model->count_all(),
421	"recordsFiltered" => \$this->Pasien_model-
422	>count_filtered(),
423	"data" => \$data,
424);
425	//output dalam format JSON
	echo json_encode(\$output);
	}
	}

Tabel 7. 15 Kode Program Model Pasien

Pasien_model.php	
1	//UNTUK DATATABEL
2	private function _get_datatables_query()
3	{
4	if(\$this->input->post('id'))
5	{
6	\$this->db->where('id', \$this->input->post('id'));
7	}
8	\$this->db->select('pasien.*');
9	\$this->db->select('petugas.nama_petugas');
10	\$this->db->select('ruang_rawat.zona_ruang');
11	\$this->db->from(\$this->table);
12	\$this->db-
13	>join('petugas', 'pasien.id_petugas=petugas.id_petugas');
14	\$this->db-
15	>join('ruang_rawat', 'pasien.id_ruang=ruang_rawat.id_ruang');
16	\$i = 0;
17	
18	foreach (\$this->column_search as \$item) // looping awal
19	{
20	if(\$_POST['search']['value']) // jika datatable
21	mengirimkan pencarian dengan metode POST
22	{
23	
24	if(\$i===0) // looping awal
25	{
26	\$this->db->group_start();
27	

```

28         $this->db->like($item,
29     $_POST['search']['value']);
30     }
31     else
32     {
33         $this->db->or_like($item,
34     $_POST['search']['value']);
35     }
36
37     if(count($this->column_search) - 1 == $i)
38         $this->db->group_end();
39     }
40     $i++;
41 }
42
43 if(isset($_POST['order']))
44 {
45     $this->db->order_by($this-
46 >column_order[$_POST['order']['0']['column']],
47 $_POST['order']['0']['dir']);
48 }
49 else if(isset($this->order))
50 {
51     $order = $this->order;
52     $this->db->order_by(key($order),
53 $order[key($order)]);
54 }
55 }
56
57 function get_datatables()
58 {
59     $this->_get_datatables_query();
60     if($_POST['length'] != -1)
61         $this->db->limit($_POST['length'], $_POST['start']);
62     $query = $this->db->get();
63     return $query->result();
64 }
65
66 function count_filtered()
67 {
68     $this->_get_datatables_query();
69     $query = $this->db->get();
70     return $query->num_rows();
71 }
72 public function count_all()
73 {
74     $this->db->from($this->table);
75     return $this->db->count_all_results();
76 }
77
78 /*
79  * Get pasien by nik
80  */
81 function get_pasien($nik)
82 {
83     return $this->db->get_where('pasien',array('nik'=>$nik))-
84 >row_array();
85 }

```

86	
87	/*
88	* Get all pasien
89	*/
90	function get_all_pasien()
91	{
92	\$this->db->select('pasien.*');
93	\$this->db->select('petugas.nama_petugas');
94	\$this->db->select('ruang_rawat.zona_ruang');
95	\$this->db-
96	>join('petugas','pasien.id_petugas=petugas.id_petugas');
97	\$this->db-
98	>join('ruang_rawat','pasien.id_ruang=ruang_rawat.id_ruang');
99	\$this->db->order_by('nik','desc');
100	return \$this->db->get('pasien')->result_array();
101	}
102	
103	/*
104	* function to add new pasien
105	*/
106	function add_pasien(\$params)
107	{
108	return \$this->db->insert('pasien',\$params);
109	}
110	
111	/*
112	* function to update pasien
113	*/
114	function update_pasien(\$nik,\$params)
115	{
116	\$this->db->where('nik',\$nik);
117	return \$this->db->update('pasien',\$params);
118	}
119	
120	/*
121	* function to delete pasien
	*/
	function delete_pasien(\$nik)
	{
	return \$this->db->delete('pasien',array('nik'=>\$nik));
	}

7.3.6. Implementasi Mengelola Petugas

Hasil implementasi kode program untuk proses mengelola petugas ditunjukkan pada tabel 7.16. Pada Controller dengan nama Petugas ini terdapat method-method yang berfungsi dalam mengelola data petugas yaitu Menampilkan detail data petugas, Menambahkan data petugas, melakukan edit data petugas, dan menghapus data petugas yang akan diteruskan pada database petugas.

Tabel 7. 16 Kode Program Controller Petugas

Petugas.php

```

1 function index()
2 {
3     $data['petugas'] = $this->Petugas_model->get_all_petugas();
4
5     $data['_view'] = 'admin/petugas/index';
6     $this->load->view('admin/layouts/main',$data);
7
8 }
9
10 /*
11  * Adding a new petugas
12  */
13 function add()
14 {
15     $this->form_validation-
16 >set_rules('username','Username','required');
17     $this->form_validation-
18 >set_rules('password','Password','required');
19     if($this->form_validation->run())
20     {
21         $params = array(
22             'id_petugas' =>str_replace("-", "", $this->uuid->v4()),
23             'aktif' => $this->input->post('aktif'),
24             'username' => $this->input->post('username'),
25             'password' => hash_password($this->input-
26 >post('password')),
27             'nama_petugas' => $this->input->post('nama_petugas'),
28             'deskripsi' => $this->input->post('deskripsi'),
29             'created_at' => date('Y-m-d H:i:s'),
30             'updated_at' => date('Y-m-d H:i:s'),
31             'created_by' => $this->session-
32 >userdata(SESSIONDATA_LOGIN_ADMIN_USERNAME),
33             'modified_by' => $this->session-
34 >userdata(SESSIONDATA_LOGIN_ADMIN_USERNAME),
35         );
36         $petugas_id = $this->Petugas_model->add_petugas($params);
37         $this->session-
38 >set_flashdata(FLASHDATA_PATH_ALERTS,'alerts/add/success');
39         redirect('petugas/index');
40     }
41     else
42     {
43         $data['_view'] = 'admin/petugas/add';
44         $this->load->view('admin/layouts/main',$data);
45     }
46 }
47
48 /*
49  * Editing a petugas
50  */
51 function edit($id)
52 {
53     // check if the petugas exists before trying to edit it
54     $data['petugas'] = $this->Petugas_model->get_petugas($id);
55
56     if(isset($data['petugas']['id_petugas']))
57     {
58

```

```

59     $this->form_validation-
60 >set_rules('username','Username','required');
61     $this->form_validation-
62 >set_rules('password','Password','required');
63     if($this->form_validation->run())
64     {
65         $params = array(
66             'aktif' => $this->input->post('aktif'),
67             'username' => $this->input->post('username'),
68             'password' => hash_password($this->input-
69 >post('password')),
70             'nama_petugas' => $this->input->post('nama_petugas'),
71             'deskripsi' => $this->input->post('deskripsi'),
72             'updated_at' => date('Y-m-d H:i:s'),
73             'modified_by' => $this->session-
74 >userdata(SESSIONDATA_LOGIN_ADMIN_USERNAME),
75         );
76         if($this->input->post('password')){
77             $param['password'] = hash_password($this->input-
78 >post('password'));
79         }
80         $this->Petugas_model->update_petugas($id,$params);
81         $this->session-
82 >set_flashdata(FLASHDATA_PATH_ALLERTS,'alerts/edit/success');
83         redirect('petugas/index');
84     }
85     else
86     {
87         $data['_view'] = 'admin/petugas/edit';
88         $this->load->view('admin/layouts/main',$data);
89     }
90 }
91 else
92     show_error('The petugas you are trying to edit does not
93 exist.');
```

```

94 }
95
96 /*
97  * Deleting petugas
98  */
99 function remove($id)
100 {
101     $petugas = $this->Petugas_model->get_petugas($id);
102
103     // check if the petugas exists before trying to delete it
104     if(isset($petugas['id_petugas']))
105     {
106         $this->Petugas_model->delete_petugas($id);
107         $this->session-
108 >set_flashdata(FLASHDATA_PATH_ALLERTS,'alerts/delete/success');
109         redirect('petugas/index');
110     }
111     else
112         show_error('The petugas you are trying to delete does not
exist.');
```

```

    }

```

Tabel 7. 17 Kode Program Model Petugas

Petugas_model.php	
1	/*
2	* Get petugas by
3	*/
4	function get_petugas_byusername(\$username)
5	{
6	return \$this->db->
7	get_where('petugas',array('username'=>\$username))->row_array();
8	}
9	
10	/*
11	* Get petugas by
12	*/
13	function get_petugas(\$id)
14	{
15	return \$this->db->
16	get_where('petugas',array('id_petugas'=>\$id))->row_array();
17	}
18	
19	/*
20	* Get all petugas
21	*/
22	function get_all_petugas()
23	{
24	\$this->db->order_by('', 'desc');
25	return \$this->db->get('petugas')->result_array();
26	}
27	
28	/*
29	* function to add new petugas
30	*/
31	function add_petugas(\$params)
32	{
33	return \$this->db->insert('petugas',\$params);
34	}
35	
36	/*
37	* function to update petugas
38	*/
39	function update_petugas(\$id,\$params)
40	{
41	\$this->db->where('id_petugas',\$id);
42	return \$this->db->update('petugas',\$params);
43	}
44	
45	/*
46	* function to delete petugas
47	*/
48	function delete_petugas(\$id)
49	{
50	return \$this->db->
51	delete('petugas',array('id_petugas'=>\$id));
	}

7.3.7. Implementasi Mengelola Ruang Rawat

Hasil implementasi kode program untuk proses mengelola daftar ruang rawat ditunjukkan pada tabel 7.18. Pada Controller dengan nama Ruang_rawat ini terdapat method-method yang berfungsi dalam mengelola data daftar ruang rawat yang tersedia pada rumah sakit darurat yaitu Menampilkan nama ruang, menambahkan daftar ruang baru dan keterangannya, melakukan edit keterangan ruang, dan menghapus nama ruang yang mungkin sudah tidak dibutuhkan yang akan diteruskan pada database Ruang_rawat_model.

Tabel 7. 18 Kode Program Controller Ruang rawat

Ruang_rawat.php	
1	function index()
2	{
3	\$data['ruang_rawat'] = \$this->Ruang_rawat_model->
4	get_all_ruang_rawat();
5	
6	\$data['_view'] = 'admin/ruang_rawat/index';
7	\$this->load->view('admin/layouts/main',\$data);
8	}
9	
10	/*
11	* Adding a new ruang_rawat
12	*/
13	function add()
14	{
15	\$this->load->library('form_validation');
16	
17	\$this->form_validation->set_rules('zona_ruang','Zona
18	Ruang','required');
19	
20	if(\$this->form_validation->run())
21	{
22	\$params = array(
23	'id_ruang' => str_replace("-", "", \$this->uuid->v4()),
24	'zona_ruang' => \$this->input->post('zona_ruang'),
25	'keterangan_ruang' => \$this->input->
26	post('keterangan_ruang'),
27	'created_at' => date('Y-m-d H:i:s'),
28	'updated_at' => date('Y-m-d H:i:s'),
29);
30	
31	\$ruang_rawat_id = \$this->Ruang_rawat_model->
32	add_ruang_rawat(\$params);
33	\$this->session->
34	set_flashdata(FLASHDATA_PATH_ALERTS, 'alerts/add/success');
35	redirect('ruang_rawat/index');
36	}
37	else
38	{
39	\$data['_view'] = 'admin/ruang_rawat/add';
40	\$this->load->view('admin/layouts/main',\$data);
41	}
42	}
43	


```

44      /*
45      * Editing a ruang_rawat
46      */
47      function edit($id_ruang)
48      {
49          // check if the ruang_rawat exists before trying to edit it
50          $data['ruang_rawat'] = $this->Ruang_rawat_model-
51      >get_ruang_rawat($id_ruang);
52
53          if(isset($data['ruang_rawat']['id_ruang']))
54          {
55              $this->load->library('form_validation');
56
57              $this->form_validation->set_rules('zona_ruang','Zona
58      Ruang','required');
59
60              if($this->form_validation->run())
61              {
62                  $params = array(
63                      'zona_ruang' => $this->input->post('zona_ruang'),
64                      'keterangan_ruang' => $this->input-
65      >post('keterangan_ruang'),
66                      'updated_at' => date('Y-m-d H:i:s'),
67                  );
68
69                  $this->Ruang_rawat_model-
70      >update_ruang_rawat($id_ruang,$params);
71                  $this->session-
72      >set_flashdata(FLASHDATA_PATH_ALLERTS,'alerts/edit/success');
73                  redirect('ruang_rawat/index');
74              }
75              else
76              {
77                  $data['_view'] = 'admin/ruang_rawat/edit';
78                  $this->load->view('admin/layouts/main',$data);
79              }
80          }
81          else
82              show_error('The ruang_rawat you are trying to edit does not
83      exist.');
```

```

84      }
85
86      /*
87      * Deleting ruang_rawat
88      */
89      function remove($id_ruang)
90      {
91          $ruang_rawat = $this->Ruang_rawat_model-
92      >get_ruang_rawat($id_ruang);
93
94          // check if the ruang_rawat exists before trying to delete it
95          if(isset($ruang_rawat['id_ruang']))
96          {
97              $this->Ruang_rawat_model->delete_ruang_rawat($id_ruang);
98              $this->session-
99      >set_flashdata(FLASHDATA_PATH_ALLERTS,'alerts/delete/success');
100              redirect('ruang_rawat/index');
101          }

```

102	else
103	show_error('The ruang_rawat you are trying to delete does not
104	exist.');
	}
	}

Tabel 7. 19 Kode Program Model Ruang rawat

Ruang_rawat_model.php	
1	/*
2	* Get status_pasien by id_status
3	*/
4	function get_status_pasien(\$id_status)
5	{
6	return
7	\$this->db->get_where('status_pasien',array('id_status'=>\$id_status))-
8	->row_array();
9	}
10	
11	/*
12	* Get all status_pasien
13	*/
14	function get_all_status_pasien()
15	{
16	\$this->db->order_by('nama_status', 'asc');
17	return \$this->db->get('status_pasien')->result_array();
18	}
19	
20	/*
21	* function to add new status_pasien
22	*/
23	function add_status_pasien(\$params)
24	{
25	\$this->db->insert('status_pasien',\$params);
26	return \$this->db->insert_id();
27	}
28	
29	/*
30	* function to update status_pasien
31	*/
32	function update_status_pasien(\$id_status,\$params)
33	{
34	\$this->db->where('id_status',\$id_status);
35	return \$this->db->update('status_pasien',\$params);
36	}
37	
38	/*
39	* function to delete status_pasien
40	*/
41	function delete_status_pasien(\$id_status)
42	{
43	return
44	\$this->db->delete('status_pasien',array('id_status'=>\$id_status));
45	}
46	}
47	

7.3.8. Implementasi Mengelola Status Pasien

Hasil implementasi kode program untuk proses mengelola daftar status pasien ditunjukkan pada tabel 7.20. Pada Controller dengan nama Status_pasien ini terdapat method-method yang berfungsi dalam mengelola data daftar status pasien berdasarkan ketentuan dari kemenkes. yaitu Menampilkan daftar status pasien, menambahkan status baru dan keterangannya, melakukan edit keterangan status, dan menghapus nama status yang mungkin sudah tidak dibutuhkan yang akan diteruskan pada database Status_pasien_model.

Tabel 7. 20 Kode Program Controller Status pasien

Status_pasien.php	
1	function index()
2	{
3	\$data['status_pasien'] = \$this->Status_pasien_model->
4	get_all_status_pasien();
5	
6	\$data['_view'] = 'admin/status_pasien/index';
7	\$this->load->view('admin/layouts/main',\$data);
8	}
9	
10	/*
11	* Adding a new status_pasien
12	*/
13	function add()
14	{
15	\$this->load->library('form_validation');
16	
17	\$this->form_validation->set_rules('nama_status','Nama
18	Status','required');
19	
20	if(\$this->form_validation->run())
21	{
22	\$params = array(
23	'id_status' => str_replace("-", "", \$this->uuid->v4()),
24	'nama_status' => \$this->input->post('nama_status'),
25	'keterangan_status' => \$this->input->
26	post('keterangan_status'),
27	'created_at' => date('Y-m-d H:i:s'),
28	'updated_at' => date('Y-m-d H:i:s'),
29);
30	
31	\$status_pasien_id = \$this->Status_pasien_model->
32	add_status_pasien(\$params);
33	\$this->session->
34	set_flashdata(FLASHDATA_PATH_ALERTS, 'alerts/add/success');
35	redirect('status_pasien/index');
36	}
37	else
38	{
39	\$data['_view'] = 'admin/status_pasien/add';
40	\$this->load->view('admin/layouts/main',\$data);
41	}
42	}

```

43
44  /*
45  * Editing a status_pasien
46  */
47  function edit($id_status)
48  {
49      // check if the status_pasien exists before trying to edit it
50      $data['status_pasien'] = $this->Status_pasien_model-
51  >get_status_pasien($id_status);
52
53      if(isset($data['status_pasien']['id_status']))
54      {
55          $this->load->library('form_validation');
56
57          $this->form_validation->set_rules('nama_status','Nama
58  Status','required');
59
60          if($this->form_validation->run())
61          {
62              $params = array(
63                  'nama_status' => $this->input->post('nama_status'),
64                  'keterangan_status' => $this->input-
65  >post('keterangan_status'),
66                  'updated_at' => date('Y-m-d H:i:s'),
67              );
68
69              $this->Status_pasien_model-
70  >update_status_pasien($id_status,$params);
71              $this->session-
72  >set_flashdata(FLASHDATA_PATH_ALERTS,'alerts/edit/success');
73              redirect('status_pasien/index');
74          }
75          else
76          {
77              $data['_view'] = 'admin/status_pasien/edit';
78              $this->load->view('admin/layouts/main',$data);
79          }
80      }
81      else
82          show_error('The status_pasien you are trying to edit does not
83  exist.');
```

```

84  }
85
86  /*
87  * Deleting status_pasien
88  */
89  function remove($id_status)
90  {
91      $status_pasien = $this->Status_pasien_model-
92  >get_status_pasien($id_status);
93
94      // check if the status_pasien exists before trying to delete
95  it
96      if(isset($status_pasien['id_status']))
97      {
98          $this->Status_pasien_model-
99  >delete_status_pasien($id_status);
100

```

101	\$this->session-
102	>set_flashdata(FLASHDATA_PATH_ALERTS, 'alerts/delete/success');
103	redirect('status_pasien/index');
104	}
105	else
106	show_error('The status_pasien you are trying to delete does
107	not exist.');
	}
	}

Tabel 7. 21 Kode Program Model Status pasien

Status_pasien_model.php	
1	/*
2	* Get ruang_rawat by id_ruang
3	*/
4	function get_ruang_rawat(\$id_ruang)
5	{
6	return
7	>get_where('ruang_rawat', array('id_ruang'=>\$id_ruang))-
8	>row_array();
9	}
10	
11	/*
12	* Get all ruang_rawat
13	*/
14	function get_all_ruang_rawat()
15	{
16	\$this->db->order_by('zona_ruang', 'asc');
17	return \$this->db->get('ruang_rawat')->result_array();
18	}
19	
20	/*
21	* function to add new ruang_rawat
22	*/
23	function add_ruang_rawat(\$params)
24	{
25	\$this->db->insert('ruang_rawat', \$params);
26	return \$this->db->insert_id();
27	}
28	
29	/*
30	* function to update ruang_rawat
31	*/
32	function update_ruang_rawat(\$id_ruang, \$params)
33	{
34	\$this->db->where('id_ruang', \$id_ruang);
35	return \$this->db->update('ruang_rawat', \$params);
36	}
37	
38	/*
39	* function to delete ruang_rawat
40	*/
41	function delete_ruang_rawat(\$id_ruang)
42	{
43	

44	return	\$this->db-
45	>delete('ruang_rawat', array('id_ruang'=>\$id_ruang));	
	}	
	}	

7.3.9. Implementasi log riwayat Pasien

Pada tabel 7.22. ditunjukkan tabel Model dari log riwayat pasien yang akan digunakan pada Controller Pasien yang dipanggil untuk dapat dilakukan tracking pasien dari data log riwayat pasien tersebut.

Tabel 7. 22 Kode Program Model Log riwayat pasien

Log_riwayat_pasien.php		
1	<?php	
2	class Log_riwayat_pasien_model extends CI_Model	
3	{	
4	function __construct()	
5	{	
6	parent::__construct();	
7	}	
8		
9		
10	/*	
11	* Get tracking_statu by	
12	*/	
13	function get_Log_riwayat_pasien(\$array)	
14	{	
15	\$this->db->select('Log_riwayat_pasien.*');	
16	// \$this->db->select('status_pasien.nama_status');	
17	//\$this->db-	
18	>join('status_pasien', 'Log_riwayat_pasien.id_status=status_pasien	
19	.id_status');	
20	\$this->db->order_by('tanggal', 'ASC');	
21	return \$this->db->get_where('log_riwayat_pasien', \$array)-	
22	>result_array();	
23	}	
24		
25		
26	/*	
27	* Get all Log_riwayat_pasien	
28	*/	
29	function get_all_Log_riwayat_pasien()	
30	{	
31	\$this->db->order_by('', 'desc');	
32	return \$this->db->get('log_riwayat_pasien')->result_array();	
33	}	
34		
35		
36	/*	
37	* function to add new tracking_statu	
38	*/	
39	function add_Log_riwayat_pasien(\$params)	
40	{	
41	return \$this->db->insert('log_riwayat_pasien', \$params);	
42	}	
43		
44		
45	/*	
46	* function to update tracking_statu	
47	*/	
48	function update_Log_riwayat_pasien(\$array, \$params)	
49		

```

50 {
51     $this->db->where($array);
52     return $this->db->update('log_riwayat_pasien',$params);
53 }
54
55 /*
56 * function to delete tracking_statu
57 */
58
59 function delete_Log_riwayat_pasien($array)
60 {
61     return $this->db->delete('log_riwayat_pasien',$array);
62 }
63
64 // function get_report_kunjungan_permonth_bynama($bulan,$tahun,
65 $nama){
66     // $query = "select COUNT(*) as jumlah, lengkap.nama FROM (SE
67     LECT tujuan.nama,kunjungan.tanggal FROM `kunjungan` JOIN tujuan O
68     N kunjungan.id_tujuan = tujuan.id) as lengkap WHERE year(tanggal)
69     =".$tahun." AND month(tanggal)=".$bulan." AND lengkap.nama='".$ $
70     nama."'";
71     //
72     // $data = $this->db->query($query);
73     // return $data->result_array();
74     // }
75 }

```

BAB 8 PENGUJIAN

Pada sub bab ini akan dipaparkan hasil pengujian pada Sistem Informasi Pencatatan COVID-19 (SIPCOP) yang penulis kembangkan. Pengujian dilakukan dengan metode *Black Box Testing* untuk pengujian kebutuhan fungsional dan metode *Compatibility Testing* untuk pengujian kebutuhan *Compatibility Browser*.

8.1 Hasil Pengujian Menggunakan *Black Box Testing*

Dibawah ini terdapat beberapa tabel yang berfungsi untuk memaparkan hasil pengujian pada kebutuhan fungsional Sistem Informasi Pencatatan COVID-19 (SIPCOP) menggunakan metode *Black Box Testing*.

Tabel 8. 1 Pengujian pada proses Login Admin

Nama	Login Admin
Objek Uji	F_ADM_LOGIN
Skenario Pengujian	<p>Skenario 1: Aktor mengosongkan <i>Username</i> dan <i>Password</i>, lalu menekan tombol "Login".</p> <p>Skenario 2: Aktor hanya mengisi <i>Username</i>, lalu menekan tombol "Login".</p> <p>Skenario 3: Aktor hanya mengisi <i>Password</i>, lalu menekan tombol "Login".</p> <p>Skenario 4: Aktor mengisi salah satu <i>field</i> dengan data yang benar dan <i>field</i> yang lain dengan data yang salah.</p> <p>Skenario 5: Aktor mengisi kedua <i>field</i> dengan data yang benar.</p>
Hasil yang Diharapkan	<p>Skenario 1-4: Sistem akan menolak akses <i>Login</i> dan mengembalikan <i>User</i> menuju halaman <i>Login</i>.</p> <p>Skenario 5: Sistem akan menerima akses <i>login</i> dan <i>aktor</i> memasuki halaman <i>Dashboard</i>.</p>

Hasil yang Didapatkan	<p>Skenario 1-4: Sistem berhasil menolak akses <i>Login</i> dan mengembalikan aktor menuju halaman <i>Login</i>.</p> <p>Skenario 5: Sistem berhasil menerima akses <i>login</i> dan aktor memasuki halaman <i>Dashboard</i>.</p>
Kesimpulan	Diterima.

Tabel 8. 2 Pengujian pada proses Login Petugas

Nama	Login Petugas
Objek Uji	F_PTGS_LOGIN
Skenario Pengujian	<p>Skenario 1: Aktor mengosongkan <i>Username</i> dan <i>Password</i>, lalu menekan tombol "Login".</p> <p>Skenario 2: Aktor hanya mengisi <i>Username</i>, lalu menekan tombol "Login".</p> <p>Skenario 3: Aktor hanya mengisi <i>Password</i>, lalu menekan tombol "Login".</p> <p>Skenario 4: Aktor mengisi salah satu <i>field</i> dengan data yang benar dan <i>field</i> yang lain dengan data yang salah.</p> <p>Skenario 5: Aktor mengisi kedua field dengan data yang benar.</p>
Hasil yang Diharapkan	<p>Skenario 1-4: Sistem akan menolak akses <i>Login</i> dan mengembalikan <i>User</i> menuju halaman <i>Login</i>.</p> <p>Skenario 5: Sistem akan menerima akses <i>login</i> dan <i>aktor</i> memasuki halaman <i>Dashboard</i>.</p>

Hasil yang Didapatkan	Skenario 1-4: Sistem berhasil menolak akses <i>Login</i> dan mengembalikan aktor menuju halaman <i>Login</i> . Skenario 5: Sistem berhasil menerima akses <i>login</i> dan aktor memasuki halaman <i>Dashboard</i> .
Kesimpulan	Diterima.

Tabel 8. 3 Pengujian pada proses Input data petugas

Nama	Input Data Petugas
Objek Uji	F_ADM_INPUT
Skenario Pengujian	Skenario 1: Aktor mengosongkan <i>field</i> username petugas Skenario 2: Aktor mengosongkan <i>field</i> password petugas Skenario 3: Aktor mengosongkan <i>field</i> username dan password petugas Skenario 4: Aktor mengosongkan <i>field</i> selain username dan password Skenario 5: Aktor mengisi seluruh <i>field</i> yang wajib diisi yaitu <i>username</i> dan <i>password</i>
Hasil yang Diharapkan	Skenario 1-3: Sistem menolak masukan data dari aktor dan memberi perintah wajib mengisi <i>field username</i> dan <i>password</i> Skenario 4-5: Sistem akan menerima dan menyimpan masukan data dari aktor.
Hasil yang Didapatkan	Skenario 1-3: Sistem berhasil menolak masukan data dari aktor. Skenario 4-5: Sistem berhasil menerima dan menyimpan masukan data dari aktor.

Kesimpulan	Diterima.
-------------------	-----------

Tabel 8. 4 Pengujian pada proses Melihat data petugas

Nama	Melihat Data Petugas
Objek Uji	F_ADM_VIEW
Skenario Pengujian	Aktor memilih sub menu petugas
Hasil yang Diharapkan	Sistem menampilkan daftar petugas pada sub menu petugas
Hasil yang Didapatkan	Sistem berhasil menampilkan daftar petugas pada sub menu petugas
Kesimpulan	Diterima.

Tabel 8. 5 Pengujian pada proses edit data petugas

Nama	Edit data petugas
Objek Uji	F_ADM_EDIT
Skenario Pengujian	<p>Skenario 1: Aktor mengosongkan <i>field</i> username petugas</p> <p>Skenario 2: Aktor mengosongkan <i>field</i> password petugas</p> <p>Skenario 3: Aktor mengosongkan <i>field</i> username dan password petugas</p> <p>Skenario 4: Aktor mengosongkan <i>field</i> selain username dan password</p> <p>Skenario 5: Aktor mengisi seluruh <i>field</i> yang wajib diisi yaitu <i>username</i> dan <i>password</i></p>
Hasil yang Diharapkan	Skenario 1-3:

	<p>Sistem menolak masukan data dari aktor dan memberi perintah wajib mengisi field <i>username</i> dan <i>password</i></p> <p>Skenario 4-5:</p> <p>Sistem akan menerima dan menyimpan masukan data yang diubah dari aktor.</p>
Hasil yang Didapatkan	<p>Skenario 1-3:</p> <p>Sistem berhasil menolak masukan data dari aktor.</p> <p>Skenario 4-5:</p> <p>Sistem berhasil menerima dan menyimpan masukan data yang diubah dari aktor.</p>
Kesimpulan	Diterima.

Tabel 8. 6 Pengujian pada proses delete data petugas

Nama	Delete Petugas
Objek Uji	F_ADM_DELETE
Skenario Pengujian	Aktor menghapus daftar petugas
Hasil yang Diharapkan	Sistem akan menghapus data petugas
Hasil yang Didapatkan	Sistem berhasil menghapus data petugas
Kesimpulan	Diterima.

Tabel 8. 7 Pengujian pada proses mengelola daftar status

Nama	Mengelola daftar status
Objek Uji	F_ADM_STATUS
Skenario Pengujian	Skenario 1:

	<p>Aktor mengosongkan field nama status</p> <p>Skenario 2:</p> <p>Aktor mengosongkan field keterangan</p> <p>Skenario 3:</p> <p>Aktor mengubah data petugas</p> <p>Skenario 4:</p> <p>Aktor menghapus data petugas</p>
Hasil yang Diharapkan	<p>Skenario 1:</p> <p>Sistem akan menolak masukan dari aktor</p> <p>Skenario 2:</p> <p>Sistem akan menerima masukan dari actor</p> <p>Skenario 3:</p> <p>Sistem akan menerima masukan data yang diubah oleh aktor.</p> <p>Skenario 4:</p> <p>Sistem akan menghapus data petugas</p>
Hasil yang Didapatkan	<p>Skenario 1:</p> <p>Sistem berhasil menolak masukan dari aktor</p> <p>Skenario 2:</p> <p>Sistem berhasil menerima masukan dari actor</p> <p>Skenario 3:</p> <p>Sistem berhasil menerima masukan data yang diubah oleh aktor</p> <p>Skenario 4:</p> <p>Sistem berhasil menghapus data petugas</p>
Kesimpulan	Diterima.

Tabel 8. 8 Pengujian pada proses mengelola daftar ruang rawat

Nama	Mengelola daftar ruang rawat
Objek Uji	F_ADM_RUANG

Skenario Pengujian	Skenario 1: Aktor mengosongkan field nama ruang rawat Skenario 2: Aktor mengosongkan field keterangan Skenario 3: Aktor mengubah data petugas Skenario 4: Aktor menghapus data petugas
Hasil yang Diharapkan	Skenario 1: Sistem akan menolak masukan dari aktor Skenario 2: Sistem akan menerima masukan dari actor Skenario 3: Sistem akan menerima masukan data yang diubah oleh aktor. Skenario 4: Sistem akan menghapus data petugas
Hasil yang Didapatkan	Skenario 1: Sistem berhasil menolak masukan dari aktor Skenario 2: Sistem berhasil menerima masukan dari actor Skenario 3: Sistem berhasil menerima masukan data yang diubah oleh aktor Skenario 4: Sistem berhasil menghapus data petugas
Kesimpulan	Diterima

Tabel 8. 9 Pengujian pada proses input data pasien

Nama	Input Data pasien
Objek Uji	F_PTGS_INPUT
Skenario Pengujian	Skenario 1:

	<p>Aktor mengosongkan salah satu <i>field</i> pada form tambah data pasien</p> <p>Skenario 2:</p> <p>Aktor mengisi <i>field</i> dengan data yang tidak sesuai dengan format yang telah ditentukan</p> <p>Skenario 3:</p> <p>Aktor mengisi seluruh field dengan format data yang benar</p>
Hasil yang Diharapkan	<p>Skenario 1-2:</p> <p>Sistem akan menolak masukan data dari aktor</p> <p>Skenario 3:</p> <p>Sistem akan menerima dan menyimpan masukan data dari aktor.</p>
Hasil yang Didapatkan	<p>Skenario 1-3:</p> <p>Sistem berhasil menolak masukan data dari aktor.</p> <p>Skenario 4-5:</p> <p>Sistem berhasil menerima dan menyimpan masukan data dari aktor.</p>
Kesimpulan	Diterima.

Tabel 8. 10 Pengujian pada proses melihat data pasien

Nama	Melihat data pasien
Objek Uji	F_PTGS_VIEW
Skenario Pengujian	Aktor memilih sub menu pasien
Hasil yang Diharapkan	Sistem menampilkan daftar pasien pada sub menu pasien
Hasil yang Didapatkan	Sistem berhasil menampilkan daftar pasien pada sub menu pasien
Kesimpulan	Diterima.

Tabel 8. 11 Pengujian pada proses edit data pasien

Nama	Edit data pasien
Objek Uji	F_PTGS_EDIT
Skenario Pengujian	<p>Skenario 1: Aktor mengosongkan salah satu <i>field</i> pada form edit data pasien</p> <p>Skenario 2: Aktor mengisi <i>field</i> dengan data yang tidak sesuai dengan format yang telah ditentukan</p> <p>Skenario 3: Aktor mengisi seluruh field dengan format data yang benar</p>
Hasil yang Diharapkan	<p>Skenario 1-2: Sistem akan menolak masukan data yang diubah dari aktor</p> <p>Skenario 3: Sistem akan menerima dan menyimpan masukan data yang telah diubah dari aktor.</p>
Hasil yang Didapatkan	<p>Skenario 1-3: Sistem berhasil menolak masukan data yang diubah dari aktor.</p> <p>Skenario 4-5: Sistem berhasil menerima dan menyimpan masukan data yang telah diubah dari aktor.</p>
Kesimpulan	Diterima.

Tabel 8. 12 Pengujian pada proses delete pasien

Nama	Delete pasien
Objek Uji	F_PTGS_DELETE
Skenario Pengujian	Aktor menghapus daftar petugas

Hasil yang Diharapkan	Sistem akan menghapus data petugas
Hasil yang Didapatkan	Sistem berhasil menghapus data petugas
Kesimpulan	Diterima.

Tabel 8. 13 Pengujian pada proses melihat grafik harian dan bulanan

Nama	Melihat grafik harian dan bulanan
Objek Uji	F_PTGS_DASHBOARD
Skenario Pengujian	Aktor memilih sub menu dashboard
Hasil yang Diharapkan	Sistem akan menampilkan halaman dashboard yang menunjukkan pantauan grafik harian dan bulanan
Hasil yang Didapatkan	Sistem berhasil menampilkan halaman dashboard yang menunjukkan pantauan grafik harian dan bulanan
Kesimpulan	Diterima.

Tabel 8. 14 Pengujian pada proses tracking riwayat pasien

Nama	<i>Tracking</i> riwayat Pasien
Objek Uji	F_PTGS_TRACKING

Skenario Pengujian	Skenario 1: Aktor mengosongkan <i>field</i> pencarian No KTP Skenario 2: Aktor memasukkan No KTP yang tidak terdaftar sebagai pasien Skenario 3: Aktor memasukkan No KTP yang terdaftar sebagai pasien
Hasil yang Diharapkan	Skenario 1-2: Sistem akan menolak masukan data dari aktor dan memberi perintah untuk mengisi field tersebut Skenario 3: Sistem akan menampilkan No KTP yang diinputkan
Hasil yang Didapatkan	Skenario 1-2: Sistem berhasil menolak masukan data dari aktor dan memberi perintah untuk mengisi field tersebut Skenario 3: Sistem berhasil menampilkan No KTP yang diinputkan
Kesimpulan	Diterima.

Tabel 8. 15 Pengujian pada proses melihat daftar status

Nama	Melihat daftar status
Objek Uji	F_PTGS_VIEWSTATUS
Skenario Pengujian	Aktor memilih sub menu pasien dan menekan tombol "Daftar Status"
Hasil yang Diharapkan	Sistem akan menampilkan daftar status pasien
Hasil yang Didapatkan	Sistem berhasil menampilkan daftar status pasien

Kesimpulan	Diterima.
-------------------	-----------

Tabel 8. 16 Pengujian pada proses edit status pasien

Nama	Edit status pasien
Objek Uji	F_PTGS_EDITSTATUS
Skenario Pengujian	Aktor menekan tombol “Ubah Status”
Hasil yang Diharapkan	Sistem akan melakukan update data status pasien
Hasil yang Didapatkan	Sistem berhasil melakukan update data status pasien
Kesimpulan	Diterima.

8.2 Hasil Pengujian Menggunakan *Browser Compatibility Testing*

Compatibility Testing dilakukan dengan menggunakan perangkat lunak SortSite. Pengujian ini bertujuan untuk memastikan bahwa Sistem Informasi Pencatatan COVID-19 ini dapat berjalan di beberapa browser yang berbeda dengan baik. Browser yang digunakan ditunjukkan pada Tabel 8.16

Tabel 8. 17 Browser Compatibility Testing

No.	Nama Browser	Versi Browser
1	Internet Explorer	11
2	Edge	14
3	Firefox	76
4	Safari	13
5	Opera	68

6	Chrome	81
7	iOS	<= 11, 12, dan 13
8	Android	<= 3 dan 4*

Pada pengujian *Compatibility Testing* ini menggunakan perangkat lunak SortSite yang membagi masalah menjadi 3 jenis yaitu *critical issues*, *major issues* dan *minor issues*. Dari pengujian yang dilakukan dapat diketahui bahwa tidak terdapat *critical issues*, kesalahan mayor, maupun kesalahan minor. Dikarenakan pada sistem sudah mendukung penggunaan tampilan sistem seperti format CSS, HTML, dan Bootstrap. Hasil dari pengujian *compatibility testing* menyatakan bahwa sistem mampu berjalan dengan baik sesuai dengan prosedur yang telah ditentukan. Berikut adalah hasil pengujian *Compatibility* pada gambar 8.1

Address: C:\Users\LAPTOP\AppData\Local\Temp\SortSite3368\PowerMapper\Map2\map.BUG.htm

file:///C:/Users/LAPTOP/Downloads/covid

Summary Issues Errors Accessibility Compatibility Search Standards Usability

This tab shows pages that exhibit browser-specific behavior, or trigger browser bugs.

Browser	IE	Edge	Firefox	Safari	Opera	Chrome	iOS	Android
Version	11	84	78	13	69	84	≤ 11 12 13	84
Critical Issues	✓	✓	✓	✓	✓	✓	✓	✓
Major Issues	✓	✓	✓	✓	✓	✓	✓	✓
Minor Issues	✓	✓	✓	✓	✓	✓	✓	✓

Key

- Missing content or functionality
- Major layout or performance problems
- Minor layout or performance problems

Priority	Description and URL	Guideline and Line#	Count
Expand all 0 issues			

Gambar 8. 1 Hasil Compatibility Testing

BAB 9 PENUTUP

9.1 Kesimpulan

Proses pengembangan Sistem Informasi Pencatatan COVID-19 (SIPCOP) telah dilalui dengan melakukan analisis kebutuhan dan perancangan sistem, kemudian dilanjutkan dengan melakukan implementasi dan pengujian sistem. Berdasarkan kegiatan tersebut, dapat diambil kesimpulan sebagai berikut :

1. Pada proses analisis kebutuhan didapatkan 2 aktor, 16 kebutuhan fungsional dan 1 kebutuhan non fungsional. Kebutuhan fungsional adalah kebutuhan pokok dari sistem. Kebutuhan fungsional dibagi berdasarkan *role* atau *user* dalam sistem yang terdiri dari Admin dan Petugas yang masing-masing memiliki beberapa fungsi yang telah dijabarkan lebih detail dalam *Use Case Diagram* dan *Activity Diagram*. Kebutuhan non fungsional adalah kebutuhan yang diperlukan oleh sistem itu sendiri atau bisa dikatakan sebagai batasan layanan dari sistem.
2. Pada proses perancangan sistem dilakukan tiga buah perancangan yang terdiri dari Pemodelan *Sequence Diagram*, Pemodelan PDM (*Physical Data Model*), dan Perancangan UI (*User Interface*). Pemodelan *Sequence Diagram* bertujuan untuk menjelaskan hubungan dan interaksi antar objek dengan *user* dalam sistem sehingga mempermudah *user* dalam memahami cara sistem bekerja. *Sequence Diagram* terdiri dari 7 urutan yaitu Login Admin, input petugas, input ruang, input status, login petugas, input pasien, dan tracking riwayat pasien. Pemodelan PDM bertujuan untuk menjelaskan hubungan antar objek data dalam database yang terdiri dari 7 tabel. Perancangan UI ditujukan sebagai acuan dalam membuat tampilan dari Sistem Informasi Pencatatan COVID—19 (SIPPCOP) yang terdiri dari login admin, login petugas, dashboard, input data pasien, input data petugas, kelola ruang, kelola status, dan tracking riwayat pasien.
3. Pada proses implementasi diterapkan sistem yang telah dirancang, yaitu implementasi *user interface*, implementasi *database*, dan implementasi kode program. Implementasi *user interface* dilakukan dengan memperhatikan kebutuhan user dalam sistem yang berbeda-beda antara admin dan petugas. Implementasi *database* dilakukan dengan menggunakan MySQL dan phpMyAdmin yang terdiri atas 7 tabel yang dibutuhkan dalam sistem. Implementasi kode program dilakukan dengan menggunakan bahasa pemrograman PHP yang terdiri atas beberapa method dengan fungsi antara lain login, input petugas, input pasien, mengelola ruang rawat, mengelola status, melihat grafik, melihat tracking riwayat pasien.

4. Pada proses pengujian dilakukan metode *Black Box Testing* yang menguji validitas dari Sistem Informasi Pencatatan COVID-19 (SIPCOP) dengan spesifikasi yang telah ditentukan tanpa memperhatikan internal sistem. Hasil dari pengujian *Black Box Testing* didapatkan bahwa setiap fungsionalitas sistem telah terpenuhi, yakni proses *Login* admin dan petugas, input petugas, mengelola ruang rawat, mengelola status, input pasien, melihat grafik, melihat tracking riwayat pasien. Sedangkan untuk hasil pengujian *Compatibility Testing* dibantu dengan *tools SortSite* dan didapatkan bahwa sistem telah memenuhi kriteria *compatibility*.

9.2 Saran

Berdasarkan perancangan Sistem Informasi Pencatatan COVID-19 (SIPCOP) berbasis web yang telah dilakukan bersama dengan pihak Dinas Komunikasi dan Informatika Kabupaten Gresik untuk memenuhi permintaan dari Bupati Gresik agar dikembangkannya sistem informasi pada Rumah sakit darurat, peneliti ingin memberikan saran yang dapat digunakan sebagai acuan penelitian selanjutnya :

1. Hasil perancangan dan implementasi Sistem Informasi Pencatatan COVID-19 (SIPCOP) berbasis web diharapkan kedepannya dapat dilakukan pengembangan lebih lanjut oleh pihak Dinas Komunikasi dan Informatika Kabupaten Gresik.
2. Untuk pihak Dinas Komunikasi dan Informatika Kabupaten Gresik agar lebih mematuhi model pengembangan sistem yang telah ditentukan sehingga sistem dapat lebih siap untuk digunakan.

DAFTAR PUSTAKA

- Alfandi, Aris Styo. 2019. "Mengenal Lebih Dekat Dengan Visual Studio Code." <https://alfanbro99.wordpress.com/2019/02/28/mengenal-lebih-dekat-dengan-visual-studio-code/> (September 30, 2020).
- Ansori. 2020. "Pengertian Activity Diagram : Tujuan, Simbol, Dan Contohnya." <https://www.ansoriweb.com/2020/03/pengertian-activity-diagram.html> (September 17, 2020).
- CNN. 2020. "Jokowi Umumkan 2 WNI Positif Corona." *CNN Indonesia* 2020. <https://www.cnnindonesia.com/nasional/20200302111534-20-479660/jokowi-umumkan-dua-wni-positif-corona-di-indonesia> .
- Destiningrum, Mara, and Qadhli Jafar Adrian. 2017. "Sistem Informasi Penjadwalan Dokter Berbassis Web Dengan Menggunakan Framework Codeigniter (Studi Kasus: Rumah Sakit Yukum Medical Centre)." *Jurnal Teknoinfo* 11(2): 30.
- Dezaneru, Rio. 2016. "Pengembangan Sistem Informasi." <https://riodezaneru.wordpress.com/2016/05/17/pengembangan-sistem-informasi/>.
- Dini, Nikita. 2017. "Pengertian Class Diagram, Kegunaan, Dan Contoh Menurut Para Ahli." <https://modulmakalah.blogspot.com/2017/01/Pengertian.Class.Diagram.Kegunaan.dan.Contoh.Menurut.Para.Ahli.html> (September 17, 2020).
- Fatimah, Umi. 2013. "Compatibility Testing." <http://fatimahumi.blogspot.com/2014/03/compatibility-testing.html> (September 30, 2020).
- Jayan. 2010. "CSS (Cascading Style Sheet)." : 2. <https://aguslilinkecil.wordpress.com/2014/08/27/css-cascading-style-sheet/> (September 30, 2020).
- Jayanti, Dwi, and Iriani Siska. 2014. "Sistem Informasi Penggajian Pada CV . Blumbang Sejati Pacitan." *Sistem Informasi Penggajian* 6(3): 36–43. <http://ijns.org/journal/index.php/speed/article/view/1041/1029>.
- Jogianto. 1999. "KONSEP MVC (MODEL-VIEW-CONTROLLER) DALAM MEMBANGUN FRAMEWORK." <https://www.kapalomen.com/2016/07/konsep-mvc-model-view-controller-framework.html> (September 30, 2020).

- Kurniawati, Peni. 2018. "Pengujian Sistem." <https://medium.com/skyshidigital/pengujian-sistem-52940ee98c77> (September 30, 2020).
- Liputan6. 2020. "'Gresik Catat Rekor Harian Tertinggi Pasien COVID-19.'" <https://surabaya.liputan6.com/read/4296159/gresik-catat-rekor-harian-tertinggi-pasien-covid-19> (September 2, 2020).
- Maulana, Robi, and R Fitria Rachmawati. 2017. "Membangun Website E-Commerce Menggunakan Framework Codeigniter Pada Chemistry Merch." 5(2): 86–96.
- Mubyarsah, Latu Ratri. 2020. "'Kasus Positif Covid-19 Di Gresik Meningkat, Rumah Sakit Overload.'" <https://www.jawapos.com/surabaya/09/07/2020/kasus-positif-covid-19-di-gresik-meningkat-rumah-sakit-overload/> (September 2, 2020).
- Pressman. 2012. "METODE WATERFALL." <https://raharja.ac.id/2020/04/04/metode-waterfall/>.
- Proboyekti, Umi. 2007. "Rekayasa Kebutuhan." <http://lecturer.ukdw.ac.id/othie/index.php?itemid=36>.
- Putra, Nanda Perdana. 2020. "'Kasus COVID-19 Di Jawa Timur Tertinggi Di Indonesia, Dipicu Masyarakat Yang Tak Disiplin?'"
- Riadi, Muchlisin. 2013. "Use Case Diagram." <https://www.kajianpustaka.com/2013/12/use-case-diagram.html>.
- Rizky, Dimas. 2019. "Metode Waterfall." [https://medium.com/dot-intern/sdlc-metode-waterfall-5ae2071f161d#:~:text=Metode waterfall merupakan suatu metode,%2Cimplementasi%2Cpengujian dan pemeliharaan](https://medium.com/dot-intern/sdlc-metode-waterfall-5ae2071f161d#:~:text=Metode waterfall merupakan suatu metode,%2Cimplementasi%2Cpengujian dan pemeliharaan.).
- Sanjaya, Ridwan, and Sebri Hesinto. 2018. "Rancang Bangun Website Profil Hotel Agung Prabumulih Menggunakan Framework Bootstrap." *Jurnal Teknologi dan Informasi* 7(2): 57–64.
- Saptiyulda, Erafzon. 2020. "'Khofifah Minta Stadion Joko Samudro Tiru RS Darurat Lapangan Surabaya.'"
- Setiawan, Dimas. 2019. "Pengertian CodeIgniter Dan Konsep MVC (Model View Controller)." <https://kelasprogrammer.com/pengertian-codeigniter-konsep-mvc/> (September 30, 2020).
- Sora. 2015. "Pengertian UML Dan Jenis-Jenisnya." <http://www.pengertianku.net/2015/09/pengertian-uml-dan-jenis->

jenisnya-serta-contoh-diagramnya.html.

Suhartono, Joni. 2016. "SOFTWARE TESTING." <https://sis.binus.ac.id/2016/12/16/software-testing/> (September 30, 2020).

Syafitri, Irmayani. 2019. "Pengertian Framework Beserta Fungsi Dan Jenis-Jenis Framework." https://www.nesabamedia.com/pengertian-framework/#Pengertian_Framework (September 30, 2020).

Trifaris. 2020. "Aplikasi Berbasis Web." <https://trifaris.net/contoh-aplikasi-berbasis-web/> (September 17, 2020).

Usada, Elisa, Yana Yuniarsyah, and Noor Rifani. 2012. "Rancang Bangun Sistem Informasi Jadwal Perkuliahan Berbasis JQuery Mobile Dengan Menggunakan PHP Dan MySQL." *JURNAL INFOTEL - Informatika Telekomunikasi Elektronika* 4(2): 40.

WHO. 2020. "Corona Virus." https://www.who.int/health-topics/coronavirus#tab=tab_1 (September 2, 2020).

William. 2011. "Pengertian Javascript." <https://widuri.raharja.info/index.php?title=TA1333376511> (September 30, 2020).

LAMPIRAN

Lampiran 1 : Foto Kegiatan



Lampiran 1. 1 Foto kegiatan Praktik Kerja Lapangan bersama Pembimbing Lapangan



Lampiran 1. 2 Foto kegiatan bersama Kepala Diskominfo Kabupaten Gresik

Lampiran 2 : Form Validasi User Acceptance Testing

Form Validasi *User Acceptance Testing*

PENGEMBANGAN SISTEM INFORMASI PENCATATAN COVID-19 PADA RUMAH SAKIT DARURAT GELORA JOKO SAMUDRO

Nama : Jeffry Nasri Faruki						
Jabatan :						
No.	Pertanyaan	1	2	3	4	5
1.	Apakah sistem yang kami buat pada aplikasi sudah sesuai dengan apa yang diharapkan?				✓	
2.	Apakah desain pada aplikasi yang kami buat sudah sesuai dengan keadaan di lapangan?					✓
3.	Apakah dengan dibuatnya sistem ini mempercepat proses pekerjaan?					✓
4.	Apakah sistem yang kami buat telah berjalan dengan baik dan lancar?				✓	
5.	Apakah sistem ini sangat bermanfaat pada Rumah Sakit Darurat tersebut?					✓
Saran :						

Ket :

- 1 = sangat tidak setuju
- 2 = tidak setuju
- 3 = cukup
- 4 = setuju
- 5 = sangat setuju

Mengetahui



JEFFRY NASRI FARUKI, S.KOM