Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.202071394
Vol. 7, No. 2, April 2020, hlm.235-248 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 30/E/KPT/2018 e-ISSN: 2528-6579

PENGEMBANGAN JADWAL *SHIFT* STAF *EDITOR* VIDEO PADA STASIUN TELEVISI NASIONAL TRANS7 BERBASIS ANDROID MENGGUNAKAN ALGORITMA *ANT COLONY* DENGAN *FIREBASE*

Rizal*[1], Eliyani[2]

[1,2]Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Mercu Buana
Email: [1]41516120070@student.mercubuana.ac.id, [2]eliyani@mercubuana.ac.id
*Penulis Korespondensi

(Naskah masuk: 10 Desember 2018, diterima untuk diterbitkan: 06 Februari 2020)

Abstrak

Perusahaan bisnis bidang *broadcasting* atau tepatnya stasiun televisi memiliki strategi untuk menayangkan tayangan acara yang paling ditunggu oleh penonton setianya, yaitu berupa tayangan acara yang *up to date* dan cepat dalam penyiarannya. Oleh sebab itu, diperlukan staf pada bagian pasca produksi untuk bekerja secara cepat dan sesuai dengan prosedur tayangan. Untuk mempersiapkan tayangan terbaru secara cepat, diperlukan sistem penjadwalan staf *editor* video yang jumlahnya besar secara cepat, tepat, dan dapat dilihat dengan instan oleh para *editor* video sehingga memudahkan *editing* video setiap harinya. Algoritma yang cocok untuk menghasilkan jadwal secara cepat dengan jumlah yang besar dengan tidak ada data yang bentrok adalah *ant colony* atau algoritma koloni semut. Algoritma *ant colony* ini mengacu pada cara hidup semut yang berkelompok dalam mencari makanan sehingga dapat kembali lagi ke tempat semula dengan jalur yang sama dan cepat. Data masukan yang digunakan dalam penelitian ini adalah nama *editor* dan ruangan, serta luaran berupa jadwal per *shift* untuk setiap ruangan untuk suatu periode tertentu. Penelitian ini menggunakan basis data *MySQL* dan *firebase*. Jadwal *editor* yang telah diolah pada aplikasi *back-end* kemudian diubah ke dalam bentuk aplikasi android, dengan demikian jadwal tersebut dapat dilihat oleh seluruh staf *editor* video melalui *smartphone* masing-masing. Pengujian dilakukan terhadap hasil perhitungan algoritma, fungsional sistem, integrasi, dan penerimaan yang dikembalikan pada sistem dengan mencari kesalahan pada *interface* perangkat lunak, dan pengujian penggunaan langsung kepada pengguna. Hasil pengujian menunjukkan bahwa algoritma *ant colony* dapat digunakan untuk menyusun jadwal *editor* video dengan cepat dan tepat, semua fitur berjalan dengan baik, dan tingkat kepuasan pengguna cukup tinggi.

Kata kunci: *jadwal, ant colony, android, firebase, edit*

DEVELOPMENT OF VIDEO EDITOR SHIFT SCHEDULE ON ANDROID BASED TRANS7 NATIONAL TELEVISION USING ANT COLONY ALGORITHM WITH FIREBASE

Abstract

Broadcasting or television business companies have a strategy to broadcast programs up to date and quickly. Therefore, staff in the post-production section are required to work quickly and in accordance with the show procedures. To prepare the latest shows quickly, a video editor staff scheduling system is needed. That scheduling system should be fast, precise, and can be seen instantly by video editors so that the editing processes can be done easily every day. The suitable algorithm to generate a schedule quickly in large numbers with no data clashing is ant colony. Ant colony algorithm refers to the way of ants when looking for food in a group, then they can return again to their original place with the same path and fast. The input data used in this study is the name of the editor and the room, and output in the form of a schedule per shift for each room for a certain period. This research uses MySQL and firebase databases. The editor's schedule that has been processed in the back-end application is then converted into an android application, therefore the schedule can be seen by all video editor staff via their own smartphones. Tests are carried out on the results of algorithm calculations, functional systems, integration, and feedback to the system by looking for errors on the software interface, and direct use testing to users. The test results show that the Ant Colony algorithm can be used to compile a schedule of video editors quickly and precisely, all features run well, and the level of user satisfaction is quite high.

Keywords: *schedule, ant colony, android, firebase, edit*

1. PENDAHULUAN

Sebagai perusahaan yang menayangkan berita maupun konten hiburan ke seluruh nusantara, perusahaan televisi nasional dituntut untuk senantiasa menayangkan video yang terbaru. Banyaknya video yang mesti tersedia dalam waktu relatif cepat, menuntut kecepatan kerja para pegawainya, termasuk staf di bagian pasca produksi yang salah satunya bertugas untuk melakukan pengeditan video. Pekerjaan pengeditan video ini dilakukan dua puluh empat jam dengan jumlah *editor* video yang tidak sedikit, sehingga dibutuhkan mekanisme penjadwalan *editor* video yang cepat, tepat, dan langsung dapat diketahui oleh *editor* video. Bisa jadi dalam satu minggu dibutuhkan lebih dari 500 jadwal *editor* video. Jika dilakukan secara manual, pengerjaan jadwal sebanyak itu tentulah tidak efisien.

Teknologi informasi telah banyak digunakan untuk membantu proses penjadwalan kerja. Di samping lebih cepat dan tepat, pemanfaatan teknologi informasi ini juga dapat mengurangi pemakaian kertas dan jadwal dapat diakses kapan pun dan di mana pun. Walau terdapat sisi negatif yang harus diwaspadai, menyangkut sisi keamanan, di mana jadwal tersebut bisa saja diakses oleh pihak yang tidak bertanggungjawab. Solusi sudah diberikan untuk masalah keamanan ini, yaitu antara lain dengan penambahan fitur seperti *log in* yang menggunakan *user* dan *password* (Montenegro, Pinto, & Fuentes, 2017).

Banyak algoritma yang sudah dikembangkan untuk memecahkan masalah penjadwalan. Penjadwalan bekerja dengan banyak data sehingga memicu logika perhitungan yang rumit dalam perencanaan waktu dan penempatan data. Di Indonesia terdapat tiga jenis jadwal kerja, yaitu *full time*, *part time* dan *shift time*. Pada jadwal keja terdapat empat informasi yang umumnya selalu tersedia, yaitu jam, ruangan, jenis pekerjaan, dan karyawan sebagai aktornya. Media penempatan jadwal ini berupa selembar atau beberapa lembar kertas yang disatukan sehingga membentuk kotak-kotak jadwal.

Penjadwalan memiliki dua pendekatan yaitu secara *list* dan secara kluster. Jadwal *list* dapat diartikan sebagai jadwal yang menyusun kepada nilai terbawah terhadap data yang dimasukkan, sedangkan jadwal secara kluster yakni penggabungan, pengelompokan serta pemesanan data yang sesuai dan masuk ke dalam suatu wilayah atau bagian (Wang & Sinnen, 2018).

Optimalisasi pada penjadwalan mengaitkan beberapa metode pemecah masalah seperti metode tabu *search* atau metode optimasi pencarian yang berbasis pada *local search* dengan menggunakan algoritma heuristik (*Shifting Bottleneck Procedure*). Hasil yang diperoleh menggunakan algoritma ini belum cukup untuk penempatan jadwal karena dengan proses yang cepat akan membuat beberapa data mengalami kerusakan atau *bug* (Mellado, Cubillos, & Cabrera, 2016). Algoritma lainnya yang menerapkan penjadwalan seperti algoritma genetik yang mengambil beberapa contoh data yang terurai menjadi data yang kompleks namun memakan waktu lama serta memiliki tingkatan yang rumit dalam pengembangannya. Kekurangan algoritma genetik yaitu sistem acak yang dilakukan dalam mutasi dan persilangan yang mengakibatkan kromosom pertama terbaik akan hilang (Liu, Chen, & Chou, 2014a)(Liu, Chen, & Chou, 2014b)(Parera, Sukmana, & Wardhani, 2016).

Metode baru untuk memecahkan penjadwalan telah dikembangkan dengan memperkenalkan algoritma *ant colony* sebagai pemecahan masalah. Dengan algoritma *ant colony*, tugas beban kerja dari proses pengumpulan jadwal akan lebih stabil dan cepat (Chen & Zhang, 2013)(Nosheen, Bibi, & Khan, 2013). Algoritma *ant colony* pernah dibandingkan dengan algoritma genetik dengan menerapkan pada penjadwalan tugas. Namun tingkat kecepatan pencarian global algoritma *ant colony* menghasilkan waktu terlambat dibanding algoritma genetik yang dikombinasikan pada *node-node* tertentu. Namun kecepatan tersebut tidak sebanding dengan hasil waktu yang efektif stabil (Liu et al, 2008). Begitu juga dengan penelitian yang dilakukan (Widyawati, 2018) yang menemukan bahwa algoritma *ant colony* menunjukkan penghematan waktu terhadap jadwal produksi yang digunakan pada kasus *job shop scheduling problem*.

Algoritma *ant colony* atau algoritma koloni semut merupakan algoritma yang meniru pada perilaku kebiasaan yang sering dilewati semut. Algoritma *ant colony* ini berkesinambungan dengan makhluk alam yang memiliki sifat alamiah yaitu sifat alami dari semut ketika mencari sarang atau tempat makan. Namun dalam pencarian jalur tercepat dan terstruktur, semut tidak membutuhkan penglihatan yang terang. Dalam dunia nyata semut yang mencari makan berkeliaran secara acak serta meninggalkan zat pada jejak mereka untuk dikenali atau diikuti oleh semut lainnya serta agar dapat mengenal jalan pulangnya. Zat tersebut disebut dengan feromon, semut yang mengeluarkan zat tersebut akan dideteksi oleh semut lainnya sehingga memiliki jalur yang bisa diikuti meskipun jalur itu pendek atau panjang. Akan tetapi, semut lainnya pun ikut menyebarkan feromon, sehingga akan lebih mudah bagi semut lain mendapatkan jalur terdekatnya.

Penelitian ini akan mengimplementasikan algoritma *ant colony* sebagai metode pengolahan data jadwal. Proses pengolahan data membutuhkan perangkat yang mendukung dari *memory* internal ataupun RAM (*Random Access memory*) pada *smartphone*. Tidak semua pengguna memiliki *smartphone* yang berkapasitas besar untuk *memory* internal dan RAMnya. Oleh sebab itu, pengolahan data pada jadwal *shift* staf *editor* video dilakukan diluar aplikasi yang digunakan pengguna. Pengolahan dilakukan dalam bagian *back-end*, dan

untuk tampilan dilakukan di bagian *front-end* merupakan cara yang tepat untuk meringankan beban kerja dari *smartphone* pengguna. Teknik ini digunakan juga oleh (Haekal & Eliyani, 2017) saat menghubungkan aplikasi Sikasir yang tersimpan di *Cloud* untuk terhubung ke android. Untuk penyimpanan jadwalnya menggunakan fitur basis data terbaru yakni *firebase database* di samping tetap menggunakan fitur *database* lokal. *Firebase* dari perusahaan Google dikembangkan untuk *developer* program dan dapat menampung data yang dihasilkan dari aplikasi pengolah. *Firebase database* tidak menggunakan perintah SQL (*Structured Query Language*) melainkan menggunakan JSON (*JavaScript Object Notation*) atau format pertukaran data yang mudah dibaca seperti pohon yang memiliki batang, akar dan daun. *Database* ini hanya untuk penyimpanan data publik dalam *cloud service* (Li et al, 2018).

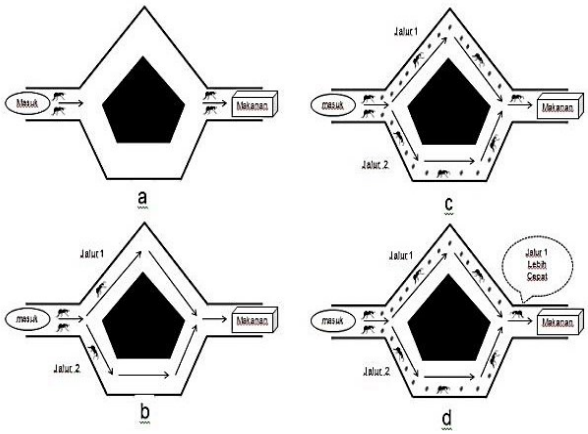
2. METODOLOGI PENELITIAN

Penelitian ini termasuk dalam penelitian kuantitatif. Dilakukan wawancara untuk menentukan data masukan dan informasi yang diharapkan.

a. Algoritma

Algoritma yang digunakan pada penelitian ini yaitu algoritma *ant colony* dengan jumlah iterasi sesuai dengan permintaan dari bakal pengguna aplikasi. Pengguna aplikasi dapat mengedit data yang diketikkan.

Algoritma *Ant Colony* disajikan pada Gambar 1 dibawah ini.

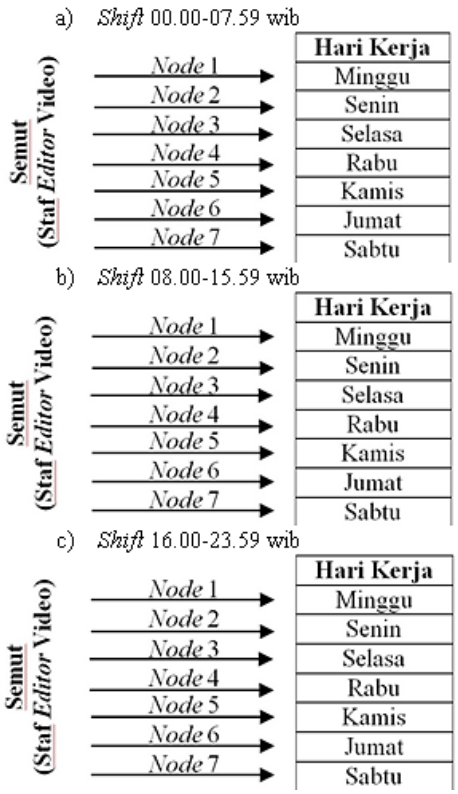


Gambar 1. Langkah Algoritma *Ant Colony* (Alfrerio Risvan Effriandi & Artha Zenda, 2010)

Gambar 1 mengilustrasikan tahapan di mana semut pada akhirnya mampu menemukan jejak tercepat dan kemudian diikuti oleh semut lainnya:

a. Langkah pertama semut berjalan mengelilingi rute/jalur.
b. Ketika semut-semut menemukan jalur yang berbeda contohnya seperti pada simpang, mereka akan berpisah dengan melewati kedua jalur tersebut.
c. Semut akan berpisah dengan melewati jalan atas dan jalan bawah, ketika mereka berjalan, mereka melakukan peninggalan jejak dengan zat feromon. Lalu jejak feromon tersebut diikuti oleh semut-semut lainnya, sehingga semut di belakangnya yakni semut jalur bagian atas dan bawah akan terlihat siapa yang datang terlebih dahulu dengan jalur tercepat.
d. Dengan mengikuti jejak tercepat menuju titik seberang, maka semut yang sudah sampai di depan akan memberitahu semut yang di belakang untuk mengikuti jejak tercepat dan semut lain akan mengikutinya.

Pada penelitian ini, pasukan semutnya adalah staf *editor* video. Pada satu ruangan hanya dapat ditempati oleh tiga *editor* video, dan satu ruangan tersebut digunakan untuk tiga *shift* kerja, yaitu *shift* jadwal kerja pukul: (a) 00.00-07.59; (b) 08.00-15.59; dan (c) 16.00-23.59. Logikanya disajikan pada Gambar 2.



Gambar 2. Logika *Ant Colony* untuk *editor* video (Alfrerio Risvan Effriandi & Artha Zenda, 2010)

Pada penerapan algoritma *ant colony*, *node-node* tersebut atau jalur tersebut diimplementasikan sebagai susunan *array* ke dalam tabel hari. Untuk menyusun serta mengolah data yang dimasukkan dari hasil yang diketikkan oleh pengolah jadwal, implementasi algoritma *ant colony* pada penelitian ini adalah sebagai berikut:

1. Langkah awal identifikasi jarak yang digunakan untuk penentuan jarak yakni dari *node* 1 ke *node* 2 (langkah 1 dan 2 untuk jalurasemut).
2. Langkah selanjutnya adalah inisialisasi parameter awal bagi feromon pada waktu ke –*t*. Pada implementasinya menggunakan *index array* dengan menghubungkan semua hari dan *editor* yang diketikkan yang membentuk sebuah *list*.
3. Menentukan jumlah semut dengan meletakkan *m* semut pada *n nodes*. Semut pada data yaitu data *editor* video yang akan diletakkan pada hari sesuai urutan hari kerja lima hari dalam satu minggu.
4. Kemudian dilakukan iterasi pada setiap langkah agar tidak ada jadwal yang bentrok dan/atau tidak terdapat ruangan yang kosong. Iterasi dilakukan sampai seluruh *editor* video mendapatkan *shift* kerja pada hari tersebut untuk periode selama satu minggu.
5. Langkah selanjutnya adalah memutakhirkan feromon untuk periode berikutnya. Feromon berupa jejak untuk menemukan hari kerja dan periode kerja.
6. Langkah terakhir adalah mencetak hasil atau jadwal yang sudah dibentuk.

b. *Database Firebase*

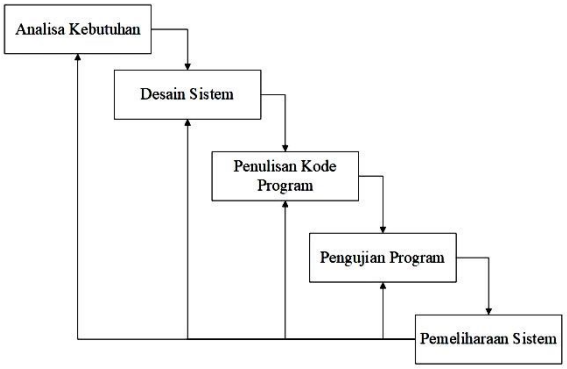
Firebase dalam penelitian ini digunakan sebagai *database* cadangan untuk menampilkan hasil jadi dari pengembangan jadwal *shift editor* video. *Firebas*e berfungsi jika *database* lokal dalam keadaan tidak terkoneksi atau terdapat gangguan. Jika data sudah selesai diolah oleh Koordinator *editor* video selaku pengolah jadwal, maka *database* utama tidak lagi dibutuhkan.

Firebase memiliki sistem yang stabil sebagai penyimpan dan penyalur *database*. Untuk mengirimkan data jadwal yang sudah diolah maupun melihat hasil jadwal dibutuhkan koneksi internet. Permintaan serta pengiriman data ke *Firebase* bersifat *realtime* sehingga dapat langsung dikirim dari sistem yang mengolah data jadwal dan diterima oleh sistem yang menampilkan data jadwal tersebut. Dengan demikian, sangat membantu dalam proses kejar tayang yang menjadi kebutuhan stasiun televisi.

Penelitian ini masih menggunakan *Firebase* yang gratis sehingga hanya mendapatkan kuota tidak melebihi 1 *gigabyte* untuk setiap proyek.

c. Metode pengembangan perangkat lunak

Pengembangan perangkat lunak pada penelitian ini menggunakan Metode SDLC (*Systems Development Life Cycle*) atau Metode *waterfall* seperti diilustrasikan pada Gambar 3. Dimulai dari (1) analisa kebutuhan sistem dengan melakukan proses wawancara di salah satu Stasiun Televisi Nasional yaitu Trans 7, kemudian dilakukan (2) desain aplikasi yang dibuat secara rinci menggunakan UML (*Unified Modeling Language*), kemudian (3) penulisan kode program yang berkaitan dengan algoritma *ant colony* dan dilengkapi dengan beberapa fitur lanjutan seperti *log in* menggunakan Bahasa Java IntelliJ IDEA versi *trial* dan android studio versi tidak berbayar, (4) pengujian program, merupakan proses verifikasi antara pengembang dan pengguna aplikasi yaitu Koordinator *editor* sebagai pembuat jadwal dan *editor* video, dan langkah terakhir yaitu (5) pemeliharaan sistem yang dilakukan pada saat bersamaan dengan penulisan kode program.



Gambar 3. Metode *Waterfall* (Pressman, 2010)

3. HASIL DAN PEMBAHASAN

3.1. Analisa kebutuhan

Setelah dilakukan analisis data yang diperoleh dari wawancara kepada pihak terkait di Stasiun Televisi Nasional Trans7, kebutuhan sistem untuk aplikasi penjadwalan *editor* video ini adalah:

1. Koordinator sebagai pengolah data melakukan:
   - Buka aplikasi jadwal
   - *Log in* sebagai Koordinator
   - Manajemen data ruangan
   - Manajemen data *editor* video
   - Buat jadwal
   - Lihat jadwal
2. *Editor* video sebagai pengguna aplikasi melakukan:
   - Buka aplikasi jadwal
   - Lihat jadwal

3.2. Desain Sistem

a. Pemodelan proses

Perancangan perangkat lunak jadwal *shift editor* video menggunakan UML yang dibagi menjadi lima model, yaitu Proses Bisnis, Diagram Konteks, Diagram Alir Data, Diagram Aktivitas, dan Diagram Urutan (*sequence*).

1. Proses Bisnis
   Proses bisnis aplikasi adalah:
   a) Aplikasi terbuka maka muncul halaman untuk *log in* dan lihat jadwal.

b) Koordinator melakukan pengetikan pada halaman *log in* untuk memulai memasukkan data.
c) Sistem menyediakan halaman untuk memasukkan data jadwal dengan tanda '+' untuk menambah seperti manajemen ruangan dan manajemen *editor*, untuk edit data dapat menekan *list* datanya.
d) Sistem merekam data masukan, namun jika ada salah ketik atau belum *input* maka akan muncul notifikasi data belum lengkap.
e) Sistem mengirimkan data yang sudah valid ke *database* utama.
f) Koordinator menekan tombol halaman lihat jadwal, lalu menekan tanda '+' untuk membuat jadwal, selanjutnya Koordinator mengisi *field* yang berada pada *form* jadwal dan menekan tombol *create* untuk mengolah jadwal. Jadwal akan terlihat dalam bentuk tabel.

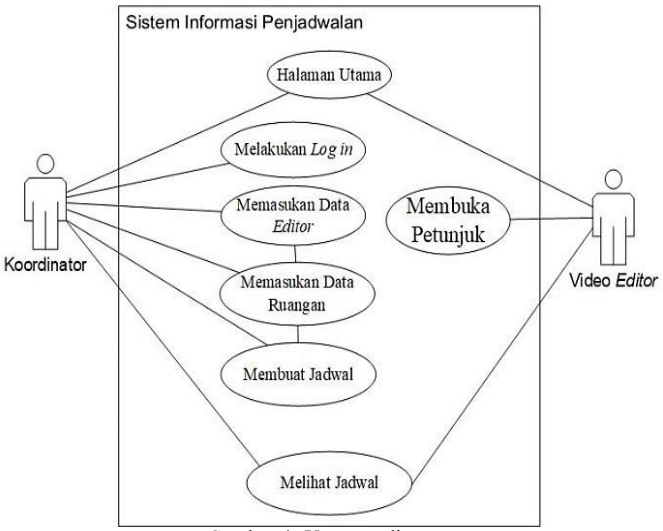
Diagram *use case* yang menggambarkan proses bisnis tersebut disajikan pada Gambar 4.

2. Diagram konteks

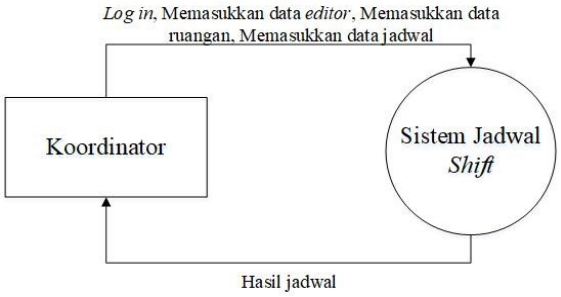
Jadwal *shift editor* video ini terdiri dari dua sistem yaitu sistem Koordinator dan sistem jadwal *shift* seperti disajikan pada Gambar 5. Sistem Koordinator merupakan aktor dalam pengolahan jadwal. Isi atau *record* dari *field-field* yang sudah dimasukkan oleh Koordinator akan disimpan pada sistem jadwal *shift*. Sistem jadwal *shift* akan mengirimkan hasil olahan jadwal kembali ke Koordinator.

3. Diagram Alir Data

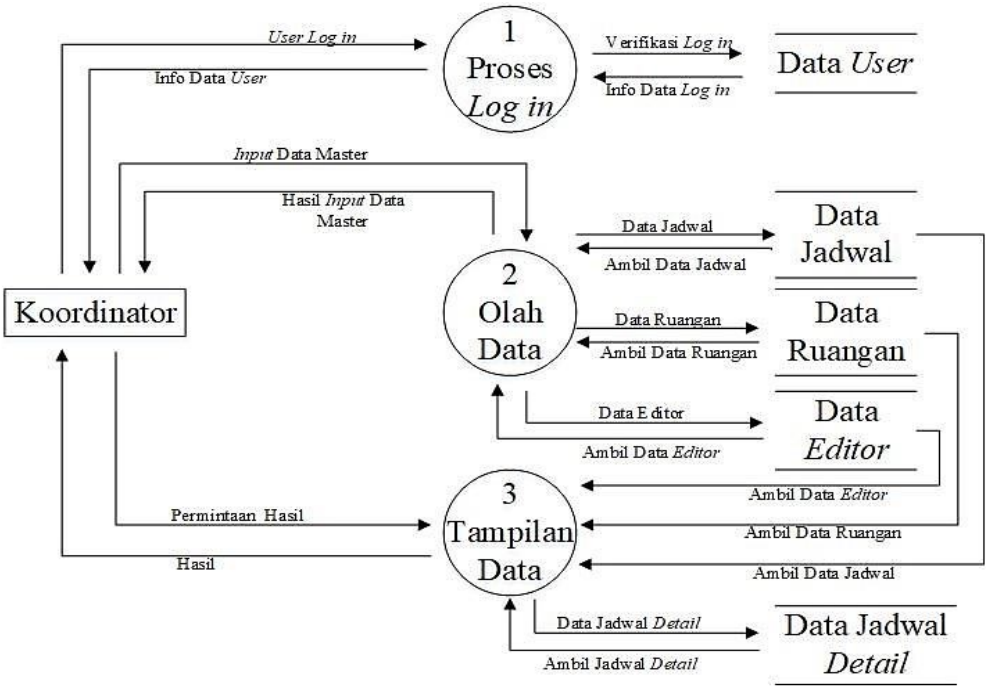
Gambar 6 menampilkan diagram alir data. Koordinator selaku pengolah jadwal melakukan *log in* untuk melanjutkan edit data *editor*, ruangan dan jadwal. Setelah data berhasil dimasukkan maka sistem tampilan data akan menyimpan data jadwal dan memberikan hasil berupa *output* ke halaman lihat jadwal yang dapat dilihat oleh Koordinator maupun *editor* video.



Gambar 4. *Use case* diagram



Gambar 5. Diagram konteks



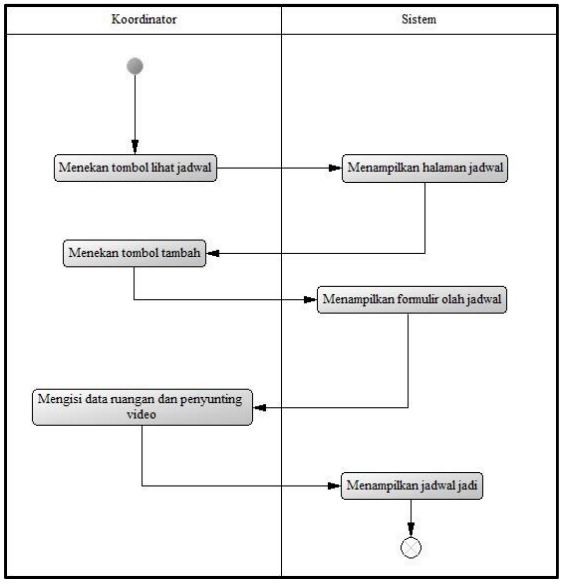
Gambar 6. Diagram alir data

4. Diagram Aktivitas

Aktivitas tambah data *editor* dilakukan setelah *log in* sebagai Koordinator dan tampilan akan muncul jika *database* utama terhubung dengan aplikasi. Koordinator menekan menu *home* pada kiri atas menu navigasi dan menekan tombol '+' pada kanan bawah agar sistem menampilkan *form* tambah *editor*. Koordinator dapat mengisi *form* data *editor* tersebut yang terdiri dari nomor induk karyawan, nama dan nomor telepon. Dengan menekan tombol *add*, sistem menyimpan data *editor* tersebut. Kemudian dilanjutkan dengan aktivitas *input* data ruangan.

Pada aktivitas mengisi data ruangan, Koordinator melakukan penekanan tombol menu *home* kembali, dan dapat menekan tombol manajemen ruangan. Sistem menampilkan *form input* ruangan untuk Koordinator mengisi nama ruangan. Dengan menekan tombol *add,* sistem langsung menyimpan data ruangan.

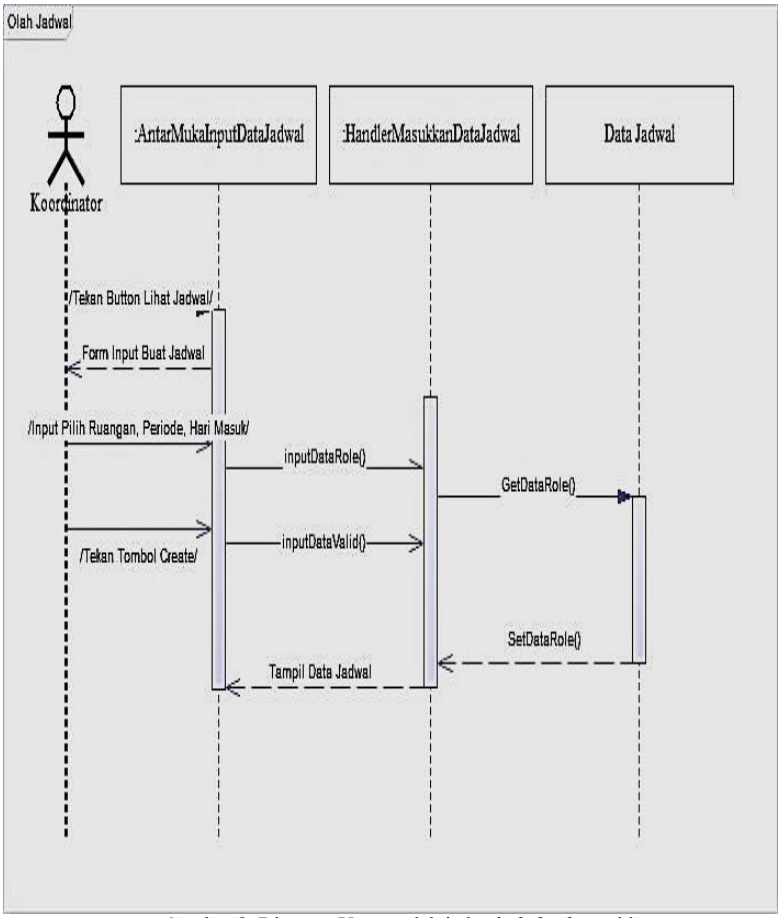
Langkah aktivitas selanjutnya adalah pengolahan jadwal. Untuk aktivitas ini, Koordinator menekan tombol lihat jadwal agar *form* olah jadwal tersedia. Setelah itu, dengan menekan tombol '+' pada kanan bawah, sistem akan menampilkan *form* buat jadwal yang terdiri dari daftar ruangan, periode, dan pilihan untuk *editor* dapat masuk di hari yang sama. Setelah Koordinator melakukan pengisian data, maka hasil berupa jadwal *shift* akan terlihat pada menu navigasi lihat jadwal. Diagram aktivitas olah jadwal ini disajikan pada Gambar 7.



Gambar 7. Aktivitas olah jadwal *shift editor* video

5. Diagram Urutan

Merujuk pada diagram aktivitas, terdapat dua data yang dimasukkan ke sistem yaitu data *editor* dan data ruangan yang berakhir pada proses penyimpan data. Proses olah jadwal dilakukan dengan aktor Koordinator dan beberapa *database* seperti jadwal, *role*, dan jadwal *detail*. Diagram urutan proses olah jadwal disajikan pada Gambar 8, namun pada Gambar 8, *database* hanya diwakilkan oleh data jadwal.



Gambar 8. Diagram Urutan olah jadwal *shift editor* video

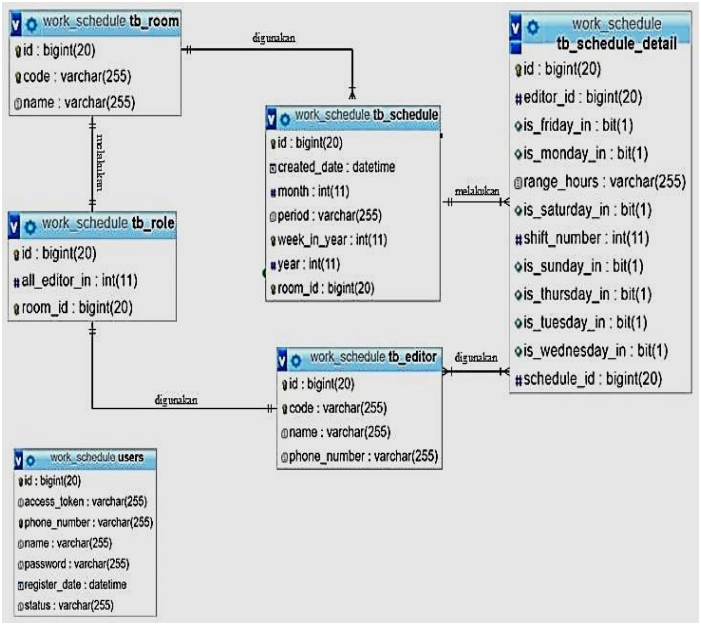
b. Pemodelan Data

ERD (*Entity Relationship Diagram*) pada *database MySQL* untuk penelitian ini disajikan pada Gambar 9.

Terdapat enam tabel *database*, yaitu:

- Tabel Ruangan memiliki:
  - id_Ruangan (*Primary key*)
  - nama_Ruangan
- Tabel *Editor* memiliki:
  - id_*Editor* (*Primary Key*)
  - nama_*Editor*
  - nomer_Telepon
- Tabel *Role* memiliki fungsi untuk menentukan semua *editor* masuk di hari yang sama:
  - id_Role (*Primary Key*)
  - all_*editor*_in
  - id_Ruangan (*Foreign Key*)
- Tabel Jadwal memiliki:
  - id_Jadwal (*Primary Key*)
  - id_Ruangan (*Foreign Key*)
  - create_Date
  - year
  - week_in_year
  - month
  - period
- Tabel Jadwal *Detail* merupakan tabel transaksi:
  - id_jadwal_*detail* (*Primary Key*)
  - id_*editor* (*Foreign Key*)
  - id_ruangan (*Foreign Key*)
  - sunday
  - monday
  - tuesday
  - thursday
  - wednesday
  - friday
  - saturday
  - shift_number
  - range_hours
- Tabel *User Log in* memiliki:
  - id_user (Primary Key)
  - nomer_telepon (Primary Key)
  - nama
  - password
  - register_date
  - status

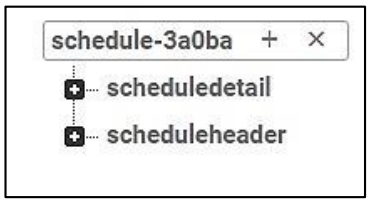
Tampilan data dalam *database* utama *MySQL* yang menyimpan tabel-tabel tersebut disajikan pada Gambar 10 dan tampilan *database Firebase* yang menjadi *database* cadangan untuk menampilkan hasil jadwal disajikan pada Gambar 11.



Gambar 9. ERD jadwal *shift editor* video

Table	Action	Rows	Type	Collation	Size	Overhead
tb_editor	Browse Structure Search Insert Empty Drop	81	InnoDB	latin1_swedish_ci	32 KiB	-
tb_role	Browse Structure Search Insert Empty Drop	28	InnoDB	latin1_swedish_ci	32 KiB	-
tb_room	Browse Structure Search Insert Empty Drop	28	InnoDB	latin1_swedish_ci	32 KiB	-
tb_schedule	Browse Structure Search Insert Empty Drop	20	InnoDB	latin1_swedish_ci	32 KiB	-
tb_schedule_detail	Browse Structure Search Insert Empty Drop	60	InnoDB	latin1_swedish_ci	32 KiB	-
users	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 KiB	-
6 tables	Sum	219	InnoDB	latin1_swedish_ci	176 KiB	0 B

Gambar 10. Tabel pada *database MySQL*



Gambar 11. Tabel pada *database Firebase*

Database utama merupakan *database MySQL* yang menyimpan berbagai tabel dan perintah SQL untuk mengolah jadwal *shift editor* video secara keseluruhan. *Database Firebase* hanya menyimpan dua *root* atau yang disebut dengan JSON (*Java Script Object Notation*). File yang disimpan dalam *database* pada pengembangan jadwal *shift* staf *editor* video memiliki 2 *file* yaitu jadwal *detail* dan jadwal *header*. Pada jadwal *header* berisi dari format atau tempat yang menyimpan hasil jadi olah data, sedangkan jadwal *detail* berisi data-data yang akan disimpan ke dalam jadwal *header*. Kedua susunan *file* tersebut akan saling berhubungan dan tidak bentrok.

Data yang digunakan dalam pengembangan jadwal *shift* staf *editor* video menggunakan data yang diambil dari contoh data yang sudah ada dari perusahaan, dalam bentuk bahasa *MySQL* disajikan pada Gambar 12. Data *editor* video yang dimasukkan sebanyak yang terdapat dalam sistem asli, yaitu 81 *editor* dengan dilengkapi data nomor induk karyawan, nama dan nomor telepon. Data ruangan yang dimasukkan sebanyak 26 ruangan seperti pada sistem nyata yang menjadi obyek pengamatan.

a. Pengujian dan Hasil

Beberapa tampilan aplikasi ini disajikan pada Gambar 13 sampai Gambar 18. Gambar 13 menampilkan halaman utama aplikasi, di mana pada *smartphone* yang memiliki operasi sistem android, tanpa melakukan perpindahan atau menekan tombol, jadwal akan muncul pada halaman utama saat membuka aplikasi jadwal *shift editor* video.

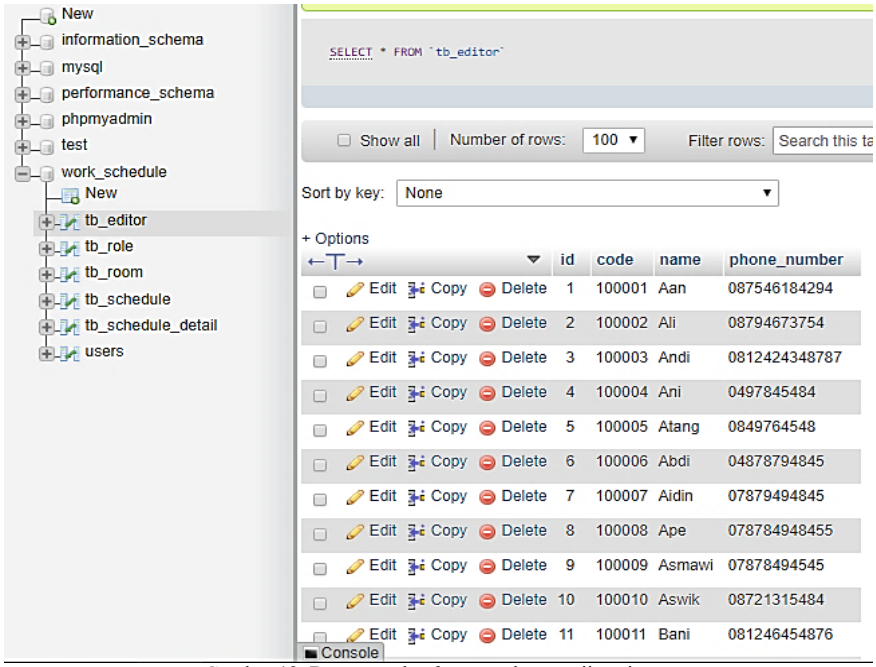
Gambar 14 menampilkan *form* manajemen ruangan untuk ditempati para *editor* video dan hanya

dituliskan dalam sebuah *text*, contoh: *Booth* A, *Booth* B dan seterusnya.

Tampilan hasil *input* ruangan disajikan pada Gambar 15.

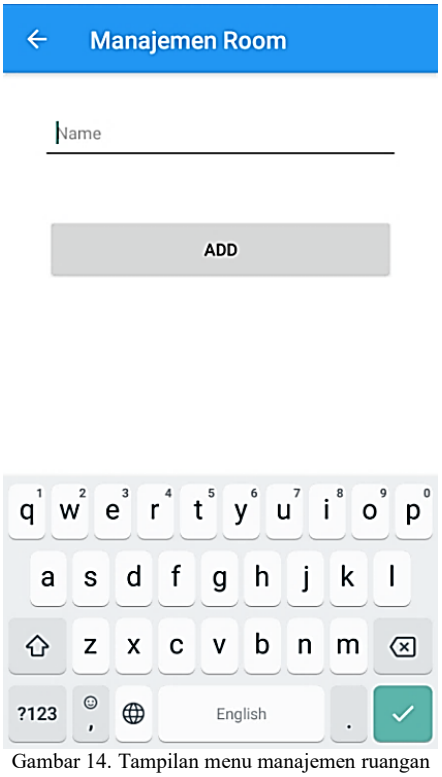
Gambar 16 menampilkan halaman manajemen *editor* di mana Koordinator dapat memasukkan nomor induk karyawan, nama dan nomor telepon *editor* video. Hasil *input* dari halaman tersebut disajikan pada Gambar 17.

Tampilan halaman untuk mengolah jadwal disajikan pada Gambar 18., terdiri dari data ruangan yang sudah dimasukkan sebelumnya, periode jadwal yang terdiri dari bulan dan tahun, serta pertanyaan kepada Koordinator yang diperlukan jika ada satu hari dalam seminggu *editor* dari semua *shift* harus masuk dan diperlukan jika ada rapat koordinasi kerja.

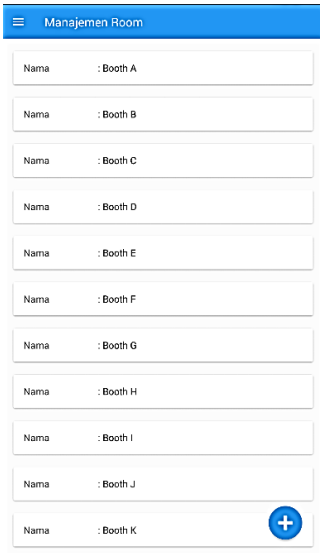


Gambar 12. Data contoh *editor* untuk pengujian sistem

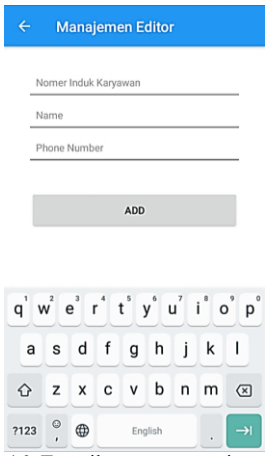
Gambar 13. Tampilan halaman utama



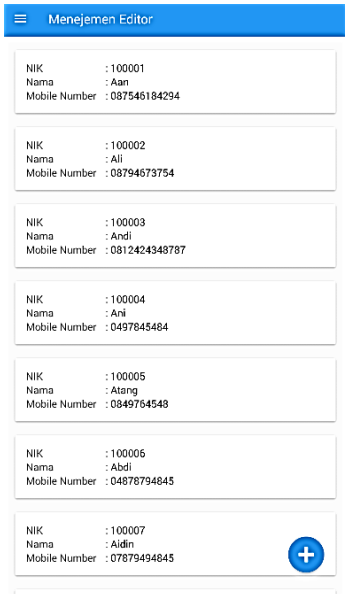
Gambar 14. Tampilan menu manajemen ruangan



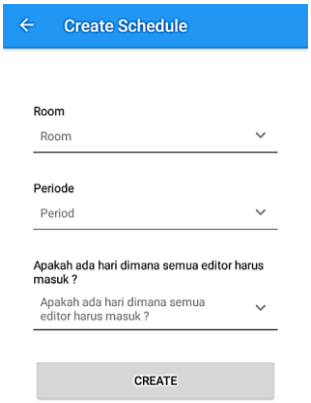
Gambar 15. Tampilan hasil *input* data ruangan



Gambar 16. Tampilan menu manajemen *editor*



Gambar 17. Tampilan hasil *input* data *editor*



Gambar 18. Tampilan halaman olah jadwal

3.3. Pengujian sistem

Pengujian aplikasi menggunakan perangkat sebagai berikut:

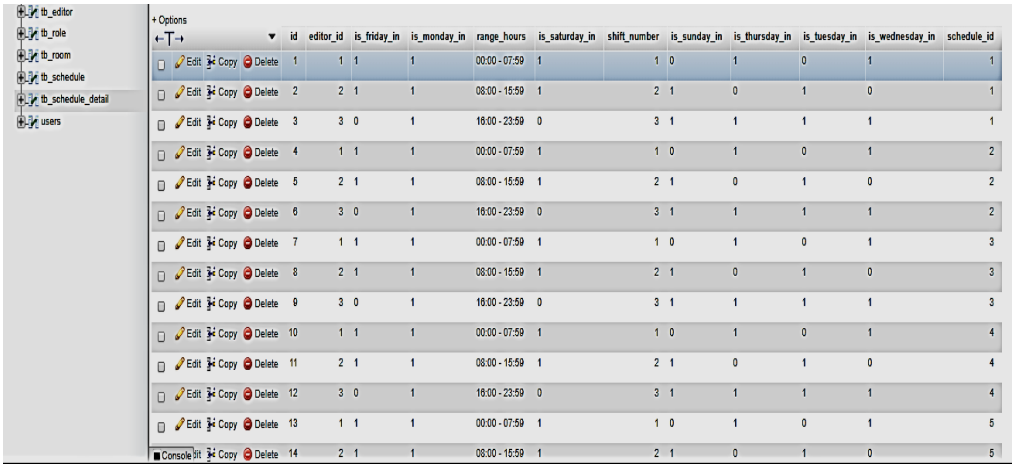
1. Sistem operasi android minimal API (*Application Programming Interface*) versi 19 atau versi sistem operasi android 4.4 (KitKat).
2. Memiliki memory internal minimal 1 *gigabyte*.
3. Memiliki kapasitas *memory* eksternal minimal 1 *gigabyte*.
4. *Smartphone* yang menggunakan aplikasi penjadwalan harus memiliki internet servis yang berguna untuk mengakses jadwal pada *Firebase*.

Pengujian algoritma a*nt colony* untuk penjadwalan kerja *editor* video dilakukan untuk 26 ruangan, 81 *editor*, periode satu pekan untuk tiga *shift* per hari. Jadwal yang dihasilkan disajikan pada Gambar 19.

Menggunakan aplikasi tersebut dihasilkan 390 jadwal dengan waktu olah data sekitar 1 sampai 2 menit.

Jadwal tersebut dapat dilihat pada *smartphone* dengan tampilan seperti pada Gambar 20 untuk satu ruangan (Ruangan A).

Selanjutnya dilakukan pengujian *black box* dan pengujian penggunaan. Pengujian *black box* dilakukan terhadap beberapa fitur untuk menguji fungsional sistem, integrasi, dan penerimaan yang dikembalikan pada sistem dengan mencari kesalahan pada *interface* perangkat lunak. Pengujian *black box* berfokus kepada tombol-tombol keseluruhan pada aplikasi jadwal *shift editor* video yang hasilnya disajikan pada Tabel 1.



Gambar 19. Pengujian sistem olah jadwal (*MySQL*)

Gambar 20. Hasil pengujian olah jadwal dalam satu ruangan

Tabel 1. Pengujian *Interface* dengan Teknik *Black Box*

No	Skenario	Kondisi	Kesimpulan
1	Halaman Utama	Ketika membuka aplikasi langsung ke halaman utama aplikasi.	*Valid*
2	*Log in* Koordinator	Koordinator sebagai administrator pengolah jadwal dapat masuk sebagai pengolah data.	*Valid*
3	Lihat Jadwal	Menampilkan jadwal yang telah selesai diolah.	*Valid*
4	Manajemen Ruangan	Halaman yang memandu koordinator untuk mengolah data ruangan seperti tambah, edit dan hapus data ruangan.	*Valid*
5	Manajemen *Editor*	Halaman yang memandu koordinator untuk mengolah data *editor* video seperti tambah, edit dan hapus data *editor*.	*Valid*
6	Petunjuk	Menampilkan petunjuk atau tata cara penggunaan aplikasi untuk melihat jadwal.	*Valid*
7	Keluar aplikasi	Menunjukkan aplikasi telah keluar.	*Valid*

Pengujian penggunaan langsung dilakukan pada pengguna aplikasi jadwal *shift editor* video yaitu aktor *editor* video dan koordinator *editor*. Tujuan dari pengujian ini adalah untuk mengetahui seberapa layak aplikasi jadwal *shift* ini dalam membantu proses menentukan jadwal *editor* video yang berkaitan dengan proses cepat tayang dalam dunia pertelevisian. Diharapkan tentu saja, aplikasi ini dapat membantu mempermudah pekerjaan *editor* video dan koordinator *editor*.

Pengujian dilakukan dengan memberikan kuesioner dan melakukan uji coba terhadap aplikasi yang telah dibangun. Untuk *editor* video diberikan empat pertanyaan, yaitu : (1) kemudahan mengakses aplikasi, (2) kemudahaan mengakses halaman petunjuk, (3) kemudahan melihat jadwal pada aplikasi, dan (4) kemudahan keluar dari aplikasi. Pengujian dilakukan terhadap 21 *editor* video. Hasil pengujian penggunaan oleh *editor* video disajikan pada Tabel 2. Rata-rata tingkat kemudahan dalam menggunakan aplikasi tersebut menurut para *editor* video sekitar 80% dalam rentang 0 hingga 100%. Kelemahan utamanya adalah belum terdapatnya halaman petunjuk penggunaan aplikasi yang menurut mereka tetap diperlukan. Saran-saran yang diberikan untuk pengembangan aplikasi disajikan pada juga pada Tabel 2.

Hasil pengujian penggunaan oleh Koordinator *editor* disajikan pada Tabel 3. Pengujian dilakukan terhadap satu-satunya Koordinator *editor* video yang menjabat di Stasiun Televisi Nasional Trans7. Pertanyaan diberikan melalui kuesioner yang diisikan setelah Koordinator menggunakan aplikasi tersebut.

Pertanyaan tersebut terdiri dari : (1) kemudahan mengakses halaman utama aplikasi, (2) kemudahaan *log in,* (3) kemudahan memasukkan data *editor*, (4) kemudahan memasukkan data ruangan, (5) kemudahan memasukkan jumlah *editor* untuk melihat hasil jadwal, (6) kemudahan untuk mengedit data, (7) tingkat kecepatan menentukan jadwal, (8) kelebihan aplikasi ini dibanding proses penjadwalan secara manual, dan (9) kemudahan keluar dari aplikasi. Kepuasan Koordinator *editor* dalam penggunaan aplikasi mencapai 90% dengan saran-saran perbaikan disajikan pada Tabel 3.

4. KESIMPULAN

Kesimpulan yang dapat diambil dari pengembangan jadwal *shift editor* video yang dibangun menggunakan algoritma *Ant Colony* dan *database* Firebase antara lain:

1. Implementasi algoritma *Ant Colony* yang didukung oleh *database* Firebase, dapat menghasilkan jadwal dengan tepat dan cepat.
2. Tingkat kepuasan pengguna yaitu para *editor* video dan Koordinator *editor* relatif tinggi sekitar 80% untuk *editor* video dan 90% untuk Koordinator *editor*.
3. *Firebase* mempermudah hasil aplikasi untuk secara *realtime* yang dilihat melalui *smartphone* android.

Tabel 2. Pengujian Penggunaan Aplikasi untuk *Editor* Video

No	Kuesioner	Hasil	Saran
1	Berapa persentase kemudahan mengakses aplikasi.	87	Sudah cukup mudah dalam penggunaan.
2	Berapa persentase kemudahaan mengakses halaman petunjuk.	52	Info petunjuk penggunaan aplikasi belum ada.
3	Berapa persentase kemudahan untuk melihat jadwal pada aplikasi.	89	Sudah cukup mudah karena tampil pada awal aplikasi dibuka.
4	Berapa persentase kemudahan untuk keluar dari aplikasi.	86	Tidak ada menu keluar
Persentase		$\frac{314}{4}$ = 79 %	

Tabel 3. Pengujian penggunaan aplikasi untuk Koordinator

No	Kuesioner	Hasil	Saran
1	Berapa persentase kemudahan mengakses halaman utama aplikasi.	90	Sudah baik karena aplikasi ringan.
2	Berapa persentase kemudahaan *log in* Koordinator.	90	Sudah cukup baik dengan *log in* nomor *handphone* dan *password* sangat unik.
3	Berapa persentase kemudahan untuk memasukkan data *editor*.	85	Perlu diperjelas pada bagian isi kolom.
4	Berapa persentase kemudahan untuk memasukkan data ruangan.	90	Sudah cukup mudah digunakan.
5	Berapa persentase kemudahan memasukkan jumlah *editor* untuk melihat hasil jadwal.	90	Sudah sangat terjangkau untuk melihat jadwal langsung.
6	Berapa persentase untuk mengubah, dan hapus masukkan data yang sudah masuk.	70	Belum ada menu hapus sehingga menyulitkan.
7	Berapa persentase tingkat kecepatan dalam menentukan jadwal.	75	Perlu *reload* halaman saat lihat jadwal.
8	Berapa persentase aplikasi ini dibanding proses manual dalam penjadwalan.	90	Lebih baik daripada proses pengolahan jadwal secara manual.
9	Berapa persentase kemudahan dalam keluar dari aplikasi	90	Sudah cukup baik karena sangat mudah keluar dari aplikasi.
Persentase		$\frac{770}{9}$ = 86%	

DAFTAR PUSTAKA

EFFRIANDI, E., ZENDA, Z. dan RACHMANSYAH, R., 2010. Rancang Bangun Aplikasi Penjadwalan Shift Kerja Lembur Menggunakan Algoritma Ant Colony Optimization (Studi Kasus : Pusri 1b). 1:1–12.

CHEN, C., NENG, W. dan ZHANG, J., 2013. Ant Colony Optimization for Software Project Scheduling and Staffing with an Event-Based Scheduler. *IEEE Transactions on Software Engineering* 39(1):1–17.

HAEKAL, H., MUHAMAD, M. dan ELIYANI, E., 2017. Token-Based Authentication Using JSON Web Token on SIKASIR RESTful Web Service. *2016 International Conference on Informatics and Computing, ICIC 2016* (Icic):175–79.

LI, L., JENG, W., YEN, C., LIN, Y.S., TUNG, S.C. dan HUANG, S.M., 2018. JustIoT Internet of Things Based on the Firebase Real-Time Database. *Proceedings - 2018 IEEE*

International Conference on Smart Manufacturing, Industrial and Logistics Engineering, SMILE 2018, 2018–Janua:43–47.

LIU, L.U., JING, J., CHEN, L., DUN,Y. dan DONG, G., 2008. The Research of Ant Colony and Genetic Algorithm in Grid Task Scheduling. *Proceedings - 2008 International Conference on MultiMedia and Information Technology, MMIT 2008,* 47–49.

LIU, L.U., KUAN, T., CHEN, Y.P. dan CHOU, J.H., 2014a. Developing a Multiobjective Optimization Scheduling System for a Screw Manufacturer: A Refined Genetic Algorithm Approach. *IEEE Access,* 2:356–64.

LIU, L.U., KUAN T., CHEN, Y.P. dan CHOU, J.H., 2014b. Solving Distributed and Flexible Job-Shop Scheduling Problems for a Real-World Fastener Manufacturer. *IEEE Access,* 2:1598–1606.

MELLADO, R., CUBILLOS, C. dan CABRERA, D., 2016. A Constructive Heuristic for Solving the Job- Shop Scheduling Problem. *Ieee Latin America Transactions,* 14(6):2758–63.

MONTENEGRO, J.A., PINTO, M., dan FUENTES, L., 2017. What Do Software Developers Need to Know to Build Secure Energy-Efficient Android Applications?. *IEEE Access,* 6:1428–50.

NOSHEEN, F., BIBI, S. dan KHAN, S., 2013. Ant Colony Optimization Based Scheduling Algorithm. *2013 International Conference on Open Source Systems and Technologies,* 18–22.

PARERA, P., SUNAN, S., SUKMANA, H.T. dan WARDHANI, L.K., 2016. Application of Genetic Algorithm for Class Scheduling (Case Study: Faculty of Science and Technology UIN Jakarta). *2016 4th International Conference on Cyber and IT Service Management,* 1–5.

PRESSMAN, R.S., 2010. *CS605-Software Engineering Practitioner's Approach*.

WANG, W., HUIJUN, H. dan SINNEN, O., 2018. List-Scheduling versus Cluster-Scheduling. *IEEE Transactions on Parallel and Distributed Systems,* 29(8):1736–49.

WIDYAWATI, W. dan Universitas Banten Jaya Serang, 2018. Penerapan Algoritma Ant Colony Optimization (Aco) Pada Job Shop Scheduling Problem (Jssp) Di Pt . Siemens Indonesia (Cilegon Factory). 1(1).

Halaman ini sengaja dikosongkan