PENGEMBANGAN KAKAS PENGUMPULAN DATA DALAM FORMAT *SPREADSHEET*

Laporan Tugas Akhir

Disusun sebagai syarat kelulusan tingkat sarjana

Oleh

Feryandi Nurdiantoro

NIM : 13513042

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

Agustus 2017

PENGEMBANGAN KAKAS PENGUMPULAN DATA DALAM FORMAT *SPREADSHEET*

Laporan Tugas Akhir

Oleh

Feryandi Nurdiantoro

NIM : 13513042

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

Telah disetujui dan disahkan sebagai Laporan Tugas Akhir

di Bandung, 23 Agustus 2017

Mengetahui,

Pembimbing I,	Pembimbing II,
Tricya Esterina Widagdo, ST., M.Sc.	Yudistira Dwi Wardhana Asnar, Ph.D
NIP 197109071997022001	NIP 198008272015041002

LEMBAR PERNYATAAN

Dengan ini saya menyatakan bahwa:

1. Pengerjaan dan penulisan Laporan Tugas Akhir ini dilakukan tanpa menggunakan bantuan yang tidak dibenarkan.
2. Segala bentuk kutipan dan acuan terhadap tulisan orang lain yang digunakan di dalam penyusunan laporan tugas akhir ini telah dituliskan dengan baik dan benar.
3. Laporan Tugas Akhir ini belum pernah diajukan pada program pendidikan di perguruan tinggi mana pun.

Jika terbukti melanggar hal-hal di atas, saya bersedia dikenakan sanksi sesuai dengan Peraturan Akademik dan Kemahasiswaan Institut Teknologi Bandung bagian Penegakan Norma Akademik dan Kemahasiswaan khususnya Pasal 2.1 dan Pasal 2.2.

Bandung, 4 Agustus 2017

Feryandi Nurdiantoro
NIM 13513042

ABSTRAK

PENGEMBANGAN KAKAS PENGUMPULAN DATA DALAM FORMAT *SPREADSHEET*

Oleh

FERYANDI NURDIANTORO

NIM : 13513042

Pengumpulan data merupakan proses kerja yang sangat penting yang sering ditemui pada kehidupan sehari-hari. Pengumpulan data biasanya dilakukan menggunakan aplikasi *spreadsheet*. Hal ini disebabkan oleh mudahnya penggunaan *spreadsheet* sehingga banyak orang awam yang memilih menggunakan *spreadsheet* dibandingkan basis data. Pengumpulan data menggunakan aplikasi *spreadsheet* memiliki beberapa kelemahan seperti lemahnya validasi, terisolasinya data yang dikumpulkan, serta terdapat kemungkinan sulitnya berkolaborasi dalam pengumpulan data. Dari permasalahan tersebut, dibuat kakas pengumpulan data berbasis *spreadsheet* yang diharapkan dapat memudahkan pengguna dalam melakukan pengumpulan data dan mengatasi permasalahan-permasalahan yang sering terjadi.

Kakas pengumpulan data ini diimplementasi sebagai fitur tambahan pada aplikasi EtherCalc sehingga permasalahan kolaborasi akan ditangani oleh aplikasi EtherCalc tersebut. Selanjutnya, pengguna dapat mendefinisikan *metadata table* secara manual atau juga dilakukan secara otomatis oleh kakas menggunakan algoritma framefinder yang telah dibuat oleh penelitian lain. Dari *metadata table* tersebut, pengguna dapat melakukan perubahan aturan-aturan validasi yang dibagi menjadi tiga tipe validasi yakni, tipe data, domain data, dan relasi antar data. Kakas akan menerjemahkan *metadata table* yang dibuat serta mencocokkannya dengan data yang ada pada *spreadsheet*, lalu memasukkan data tersebut ke dalam basis data relasional.

Pengujian dilakukan pada fitur-fitur yang diimplementasikan pada kakas. Pengujian dilakukan dengan menggunakan beberapa kasus yang dibuat dan diujikan kebenaran hasil data masukan menjadi data pada basis data.

Kata kunci: *spreadsheet*, pengumpulan data, *data quality*, *data management*.

KATA PENGANTAR

Puji syukur saya panjatkan kehadirat Tuhan Yang Maha Esa, karena degan kelimpahan rahmat dan kemurahan hati-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul ”Pengembangan Kakas Pengumpulan Data dalam Format *Spreadsheet*”.

Dalam penyusunan Tugas Akhir ini penulis banyak mendapatkan masukan, kritik, dorongan, bantuan, bimbingan, serta dukungan baik secara fisik maupun moral dari berbagai pihak yang merupakan pengalaman yang berharga yang tidak dapat diukur secara materi dan dapat menjadi pembelajaran yang berharga dikemudian hari. Oleh karena itu dengan segala hormat dan kerendahan hati perkenankanlah penulis mengucapkan terima kasih kepada:

1. Bapak Yudistira Dwi Wardhana Asnar, Ph.D dan Ibu Tricya Esterina Widagdo, ST., M.Sc. selaku pembimbing yang senantiasa memberikan arahan dan masukan selama pengerjaan Tugas Akhir.
2. Bapak Adi Mulyanto, ST., MT. selaku dosen penguji yang atas saran dan masukkannya membuat Tugas Akhir ini menjadi lebih baik.
3. Kedua orang tua penulis. Terima kasih atas dukungan baik secara moral dan material sehingga penulis dapat melaksanakan Tugas Akhir ini hingga selesai dengan baik.
4. Ibu Dr. Fazat Nur Azizah ST, M.Sc. selaku dosen Tugas Akhir yang telah mendukung saya dan memberikan arahan dalam penyelesaian dan pengerjaan Tugas Akhir ini.
5. Ibu Dr. Eng. Ayu Purwarianti, ST., MT. selaku dosen mentor Imagine Cup yang ikut membantu dan memberikan dukungan agar penulis dapat menyelesaikan pengerjaan Tugas Akhir.
6. Bapak Dr.techn. Saiful Akbar ST, MT. dan Bapak Achmad Imam Kistijantoro, ST, M.Sc, Ph.D selaku Ketua Program Studi dari Teknik Informatika dan Sistem dan Teknologi Informasi yang ikut mendukung penulis dalam menyelesaikan Tugas Akhir ini.

7. Seluruh dosen, karyawan dan civitas program studi Teknik Informatika, Institut Teknologi Bandung.
8. Rekan-rekan yang telah penulis minta bantuan dalam penyelesaian administrasi Tugas Akhir ini disaat penulis berhalangan terutama Muhamad Visat Sutarno dan Fiqie Ulya Sidiastahta
9. Rekan-rekan laboratorium basis data yang saling mendukung dalam menyelesaikan Tugas Akhir ini pada khususnya Vanya Deasy Safrina, Albert Tri Adrian, Marco Orlando, Fiqie Ulya Sidiastahta, dan Wilhelmus Andrian.
10. Rekan-rekan seperjuangan Teknik Informatika yang menamakan dirinya Happy Anti Wacana yang selalu mendukung penulis untuk mengerjakan Tugas Akhir.
11. Rekan-rekan dari Binary 2013 dan HMIF yang telah memberikan dukungan dan bantuan dalam pengerjaan Tugas Akhir.
12. Pihak-pihak lain yang tidak dapat disebutkan satu-persatu.

Penulis menyadari bahwa Tugas Akhir ini masih jauh dari sempurna serta memiliki banyak kekurangan. Oleh karena itu, penulis sangat terbuka dalam menerima kritik dan saran yang membangun untuk Tugas Akhir ini. Semoga Tugas Akhir ini dapat bermanfaat bagi pembaca.

Bandung, Agustus 2017

Penulis

Daftar Isi

Daftar Gambar

Daftar Tabel

BAB I

PENDAHULUAN

Pada bab ini akan dibahas mengenai gambaran dasar dari pelaksanaan Tugas Akhir dalam bentuk penjelasan latar belakang yang mendasari pemilihan topik. Dari latar belakang tersebut, akan diurai kembali menjadi rumusan masalah, tujuan, batasan masalah, serta metodologi yang digunakan untuk keperluan Tugas Akhir ini.

I.1 Latar Belakang

Aplikasi *spreadsheet* merupakan aplikasi yang mudah ditemui dan wajar digunakan oleh banyak orang, baik secara personal maupun dalam sebuah organisasi komersial (Chan and Storey, 1996). Pada tahun 1979, aplikasi *spreadsheet* pertama dibuat dengan nama VisiCalc. Pengguna komputer pada saat itu dimanjakan dengan kapabilitas dan fleksibilitas aplikasi yang dapat melakukan operasi sederhana tanpa harus menggunakan komputer *mainframe*. Dengan semakin berkembangnya daya komputasi, saat ini telah banyak sekali muncul aplikasi *spreadsheet* baru dan penggunaannya juga semakin beragam.

Spreadsheet memiliki beberapa keunggulan dibandingkan dengan aplikasi pengolahan data jenis lain. Keunggulan yang paling terlihat adalah banyak orang yang mengetahui cara penggunaan aplikasi jenis *spreadsheet* dan terbiasa dalam menggunakannya. Di samping itu, *spreadsheet* memiliki banyak fitur dan kemampuan yang jarang diketahui orang awam jika digunakan dengan benar. Dengan keunggulan ini, *spreadsheet* sering kali dijadikan pilihan utama dalam pengolahan dan pengumpulan data. Beberapa orang mungkin menganggap, penggunaan *spreadsheet* adalah personal sehingga tidak membutuhkan tim atau bantuan orang lain dalam pembuatan suatu *spreadsheet*. Hal ini tidak dapat dibenarkan, karena jika melihat kasus penggunaannya pada organisasi bisnis yang besar, *spreadsheet* yang dihasilkan sangatlah kompleks dan besar dengan pengembangan yang membutuhkan banyak orang (Panko, 1998).

Penggunaan *spreadsheet* yang dapat ditemui pada perusahaan atau instansi adalah sebagai

media pengumpulan data. Data yang dikumpulkan biasanya dalam bentuk *data frame* yakni *spreadsheet* yang mempunyai dua struktur utama yaitu bagian *value* atau data dan bagian *atribute* atau label (Chen and Cafarella, 2013). Data yang telah dikumpulkan biasanya akan disebarkan kepada pihak-pihak yang membutuhkan atau disimpan sebagai arsip data yang akan digunakan kembali pada saat dibutuhkan.

Sebagai media pengumpulan data, *spreadsheet* memiliki permasalahan penyimpanan data. Hal tersebut penting untuk diperhatikan jika data yang dikumpulkan mungkin tidak hanya akan ditampilkan dalam bentuk *spreadsheet* namun juga dapat digunakan oleh aplikasi lain. Permasalahan lain yang mungkin muncul adalah kompleksitas dan besarnya ukuran data pada *spreadsheet* yang membuat penggunaan *spreadsheet* pada sebuah bisnis sangatlah rentan akan kesalahan. Sebuah kesalahan kecil dapat berakibat fatal dan memberikan kerugian seperti kehilangan pendapatan, kesalahan pemberian harga, penipuan, dan kegagalan sistem akibat ketergantungan berlebih antar *spreadsheet* (EuSpRIG, 2010b). Telah banyak bukti dan penelitian yang menunjukan bahwa kesalahan pada *spreadsheet* sangat mudah ditemui. Contoh kesalahan yang dapat terjadi adalah kesalahan tipe data, kesalahan masukan, tidak divalidasinya data masukan, serta permasalahan *single version of truth* dimana bisa terdapat dua atau lebih versi dari data yang sama.

Untuk dapat mengurangi kesalahan-kesalahan yang sering terjadi pada *spreadsheet* secara lebih mendasar, dibutuhkan bantuan perangkat lunak untuk dapat melakukan kontrol terhadap masukan pengguna, melakukan validasi, serta dapat melakukan penyimpanan data yang telah dikumpulkan. Pada tugas akhir ini akan difokuskan pada pengembangan sebuah aplikasi pengumpulan data berbentuk *spreadsheet* yang dapat membantu pengguna mengurangi terjadinya masalah-masalah yang telah disebutkan sebelumnya.

I.2 Rumusan Masalah

Seperti yang telah dijelaskan pada latar belakang, salah satu penggunaan aplikasi *spreadsheet* adalah sebagai media pengumpulan data. Beberapa permasalahan yang muncul dalam pengumpulan data adalah kolaborasi, validasi, dan penyimpanan data. Pengumpulan data dapat dilakukan oleh banyak orang secara bersama-sama sehingga dapat menyebabkan konflik pada data masukan. Konflik ini bisa terjadi karena terdapat

lebih dari satu versi file yang diubah secara bersama-sama. Di samping itu, data yang dimasukkan biasanya tidak melalui proses validasi sehingga dapat menyalahi domain data yang seharusnya. Setelah data dikumpulkan ke dalam bentuk *spreadsheet*, diperlukan media penyimpanan data sehingga data tidak terisolasi. Contoh dari media penyimpanan tersebut adalah menggunakan basis data.

Hal-hal tersebut merupakan beberapa masalah utama yang cukup penting untuk diselesaikan terutama dalam penggunaannya pada bisnis dan komersial sehingga akan dibentuk sebuah kakas yang membantu pengumpulan data pada format *spreadsheet*. Dalam rangka pembangunan kakas, terdapat beberapa permasalahan yang menjadi perhatian pada tugas akhir ini, yaitu:

1. Bagaimana cara melakukan validasi terhadap data masukan?
2. Bagaimana cara data pada *spreadsheet* dapat disimpan pada basis data?
3. Bagaimana cara melakukan penyimpanan data yang telah dibuat?
4. Bagaimana proposal perubahan alur kerja yang terjadi akibat penggunaan kakas yang dibuat?

I.3 Tujuan

Tujuan yang ingin dicapai dalam Tugas Akhir ini adalah mengembangkan perangkat lunak pengumpulan data berbentuk *spreadsheet* yang dapat mengubah data dalam format *spreadsheet* menjadi data relasional serta melakukan validasi pada data masukan. Dengan adanya perangkat lunak ini diharapkan dapat dilakukan pengumpulan data yang disertai validasi data masukan.

I.4 Batasan Masalah

Dalam pengerjaan Tugas Akhir ini, terdapat beberapa batasan-batasan yang perlu diperhatikan. Batasan tersebut ditujukan untuk memperjelas dan memfokuskan objek penelitian dan pengembangan tugas akhir. Batasan-batasan masalah pengerjaan tugas akhir adalah sebagai berikut,

1. Kakas hanya dapat berjalan pada aplikasi EtherCalc.
2. Jenis *spreadsheet* yang ditangani adalah *data frame* dan relasi.

3. Data disimpan dalam bentuk basis data relasional.
4. Basis data yang didukung untuk digunakan dalam penyimpanan data hanya MySQL.
5. Struktur basis data dan tingkat normalisasi yang terbentuk merupakan tanggung jawab pengguna dan tidak ditangani oleh kakas.
6. Kakas hanya mendukung penggunaan satu *sheet* dalam satu *spreadsheet*.
7. Validasi yang dilakukan hanya validasi tipe, domain, dan relasi data.
8. Validasi tipe hanya mencakup tipe *integer*, *double*, *string*, dan *boolean*.
9. Validasi domain hanya mencakup aturan *range*, lebih besar, lebih kecil, sama dengan, dan nilai diskrit. Atribut hanya dapat menerima satu aturan validasi domain.

I.5 Metodologi

Metodologi yang digunakan dalam pengerjaan Tugas Akhir ini yakni:

1. Studi Literatur

   Pengerjaan tugas akhir diawali dengan mencari dan mempelajari referensi berupa jurnal ilmiah dan aplikasi-aplikasi yang telah ada sebelumnya yang dapat membantu pengembangan kakas yang dibuat pada tugas akhir ini. Literatur yang dicari dan dipelajari berkaitan dengan topik tugas akhir yaitu mengenai *spreadsheet*, penggunaannya pada bisnis, kesalahan yang sering dilakukan dalam pembuatan, teknik pendeteksian data yang ada pada *spreadsheet*, serta hal-hal lain yang masih berkaitan dengan topik tugas akhir ini.

2. Analisis Masalah

   Pada tahap ini dilakukan analisis permasalahan yang berkaitan dengan topik yang diangkat pada tugas akhir ini. Diantaranya adalah memecah permasalahan menjadi aplikasi *spreadsheet* yang dikembangkan, penanganan konflik saat berkolaborasi, jenis *spreadsheet* yang ditangani, metode interaksi pengguna, penentuan bagian data dan label, representasi tabel, validasi data, penyimpanan data, interaksi antar *spreadsheet*, dan proposal perubahan alur kerja.

3. Perancangan Solusi

   Pada tahap ini dilakukan perancangan solusi yang dapat menyelesaikan masalah-

masalah yang telah dijelaskan pada bagian analisis masalah. Bagian perancangan ini juga menjelaskan arsitektur yang digunakan untuk membangun perangkat lunak berdasarkan spesifikasi dan metode yang digunakan.

4. Implementasi

   Pada tahap ini dilakukan pembangunan kakas sesuai dengan kebutuhan dan spesifikasi dari hasil analisis masalah serta rancangan solusi yang diajukan.

5. Pengujian dan Analisis Hasil

   Pada tahap ini dilakukan pengujian dengan menggunakan data set uji yang sesuai dengan batasan masalah ke dalam kakas yang diimplementasikan. Selanjutnya dilakukan analisis hasil pengujian dan penarikan kesimpulan.

I.6 Sistematika Pembahasan

Penulisan tugas akhir ini terdiri dari 5 bab, yaitu: BAB I Pendahuluan, BAB II Tinjauan Pustaka, BAB III Analisis dan Perancangan, BAB IV Rancangan, Implementasi, dan Pengujian, dan BAB V Penutup.

Bab satu membahas mengenai latar belakang permasalahan, rumusan masalah, tujuan, batasan masalah, metodologi serta sistematika pembahasan yang digunakan. Bab ini juga menjelaskan secara umum isi dari tugas besar serta gambaran dasar dari pelaksanaan tugas akhir.

Bab dua menjelaskan mengenai dasar teori yang digunakan di dalam menyelesaikan permasalahan yang diangkat. Teori yang digunakan berasal dari literatur dan referensi yang berhubungan dengan permasalahan yang diangkat seperti hal-hal yang berkaitan dengan *spreadsheet*, penggunaannya pada bisnis, masalah yang sering terjadi dalam pembuatan, metode *quality control* yang dapat dilakukan untuk mencegah kesalahan, serta metode pendeteksian bagian label dan data yang pernah dibuat pada penelitian lain. Dasar teori ini menjadi dasar analisis dan rancangan solusi pada bab selanjutnya.

Bab tiga memaparkan analisa kebutuhan dan permasalahan yang dipilih yakni tingginya tingkat kesalahan yang terjadi pada *spreadsheet*. Dari hasil analisa yang dilakukan, di dapatkan bentuk solusi umum yang dapat digunakan untuk mengatasi permasalahan tersebut. Selanjutnya solusi umum tersebut dibuat rancangan dan arsitekturnya agar dapat

diimplementasikan.

Bab empat memperlihatkan rancangan perangkat lunak yang dibuat serta hasil implementasinya. Pada akhir bab akan ditunjukkan hasil pengujian yang dilakukan kepada kakas yang dibuat dan pembahasan dari pengujian tersebut. Pengujian dilakukan untuk mengetahui keberhasilan kakas yang dibuat untuk menyelesaikan permasalahan yang di definisikan pada rumusan masalah.

Bab lima berisikan kesimpulan terhadap hasil implementasi dan solusi yang dipaparkan untuk menyelesaikan permasalahan. Di samping itu, terdapat bagian saran yang memaparkan saran pengembangan dan perbaikan yang dapat dilakukan untuk memperkaya fitur dan menyelesaikan permasalahan yang lebih luas.

BAB II

STUDI LITERATUR

Pada bab ini akan dideskripsikan kajian literatur yang terkait dengan persoalan Tugas Akhir. Studi literatur ini akan dijadikan dasar di dalam melakukan penyelesaian persoalan yang telah didefinisikan.

II.1 *Spreadsheet*

Bagian ini akan membahas mengenai definisi umum *spreadsheet* serta teknologi yang sering digunakan di dalam membuat *spreadsheet*. Dengan adanya definisi umum ini diharapkan dapat menyamakan persepsi mengenai *spreadsheet* yang dimaksud pada Tugas Akhir ini. Teknologi yang dijelaskan pada bagian ini merupakan teknologi yang memungkinkan untuk dijadikan dasar pengembangan perangkat lunak pada Tugas Akhir ini.

II.1.1 Definisi Umum

Secara harafiah, *spreadsheet* adalah suatu perangkat lunak yang dapat melakukan kalkulasi terhadap angka serta mengorganisir informasi yang ada di dalamnya berdasarkan kolom dan baris (Dictionary, 2016). Konsep dasar pada aplikasi *spreadsheet* modern adalah sebuah aplikasi yang berupa sekumpulan sel terdiri dari baris dan kolom yang disebut *sheet* yang dapat digambarkan sebagai matriks yang besar (Ronen et al., 1989).

Sel-sel pada *spreadsheet* dapat diisi data berupa data mentah maupun formula. Data mentah dapat berupa angka, teks, tanggal, dan nilai mata uang. Formula merupakan perintah yang dapat dimengerti komputer untuk menghitung dan memanipulasi data pada sel. Data hasil pengolahan dan masukan pada *spreadsheet* ditampilkan dalam bentuk sel yang namanya terdiri dari nama kolom dan nilai baris (Contoh: A1 untuk kolom pertama dan baris pertama). Di samping itu, sel tersebut juga dapat memiliki *properties* berupa *value* yang diisikan, format sel, serta format data yang digunakan.

II.1.2 Teknologi *Spreadsheet*

Perkembangan teknologi *spreadsheet* digital modern dimulai pada tahun 1978, saat Bricklin mengembangkan *working prototype* dari konsep dasar *spreadsheet* menggunakan Integer BASIC. Pada tahun yang sama, Frankston dan Fylstra bergabung dan membentuk sebuah perangkat lunak bernama VisiCalc (Visible Calculator) yang merupakan sebuah perangkat lunak *spreadsheet* pertama yang bekerja dengan baik dan sukses dipasaran. Setelah keberhasilan VisiCalc, mulai muncul aplikasi serupa yang semakin baik salah satunya adalah Lotus. Dengan berkembangnya daya komputasi dan munculnya konsep *graphical user interface*, Microsoft mengembangkan Microsoft Excel yang merupakan *spreadsheet* pertama yang menggunakan antarmuka grafis dan menggunakan *mouse* sebagai alat kontrol (Power, 2004).

Saat ini, perangkat lunak berupa *spreadsheet* sangat banyak variasi dan tipenya. Perangkat lunak *spreadsheet* ini dapat dibagi berdasarkan konektivitasnya yakni *offline spreadsheet* dan *online spreadsheet*. Selain itu, perangkat lunak *spreadsheet* dapat juga dibagi berdasarkan keterbukaan dari *source code* yakni *open source* dan *closed source*. Bagian ini akan membahas masing-masing perangkat lunak tersebut secara umum.

II.1.2.1 Microsoft Excel

Microsoft Excel adalah perangkat lunak yang dikembangkan oleh Microsoft yang menyediakan fitur dasar dari *spreadsheet* serta dengan fitur-fitur lainnya yang selalu ditambahkan pada setiap iterasi pengembangan Excel. Microsoft Excel dapat dimiliki oleh pengguna melalui pembelian paket Microsoft Office yang berisikan produk esensial Microsoft lainnya (Microsoft, 2015b).

Sejak Microsoft Excel 2007, Microsoft menggunakan format Office Open XML (OOXML) sebagai format penyimpanan (Microsoft, 2015a). Office Open XML dikembangkan oleh Microsoft mulai dari tahun 2000 dengan diimplementasinya dukungan XML pada Microsoft Office 2000. Pada awal penggunaan aplikasi *office*, terdapat permasalahan *data interoperability* antar mesin dan sulitnya manipulasi data. Office Open XML diharapkan dapat menyelesaikan permasalahan ini dengan membentuk standar yang dapat diimplementasi berbagai aplikasi *office* (Microsoft, 2006).

II.1.2.2 LibreOffice Calc

LibreOffice adalah perangkat lunak yang dikembangkan oleh komunitas dan proyek dari organisasi non-profit bernama The Document Foundation. LibreOffice adalah perangkat lunak yang gratis dan *open source* yang awalnya didasarkan pada perangkat lunak serupa yakni OpenOffice.org dan merupakan pengembangan lanjutan dari OpenOffice yang paling aktif. Hampir serupa dengan Microsoft Office, LibreOffice memberikan 6 perangkat lunak yang ada di dalamnya yakni: Writer (pemrosesan teks), Calc (*spreadsheet*), Impress (presentasi), Draw (grafik dan vektor), Base (basisdata), dan Math (editor formula) (Foundation, 2016).

LibreOffice Calc memiliki berbagai kemampuan yang dimiliki oleh kebanyakan *spreadsheet*. Di dalam melakukan penyimpanan file, Calc menggunakan format OpenDocument. Format OpenDocument dikembangkan oleh Organization for the Advancement of Structured Information Standards (OASIS) yang bertujuan untuk membentuk *open standard* bagi *office document* (OASIS, 2016).

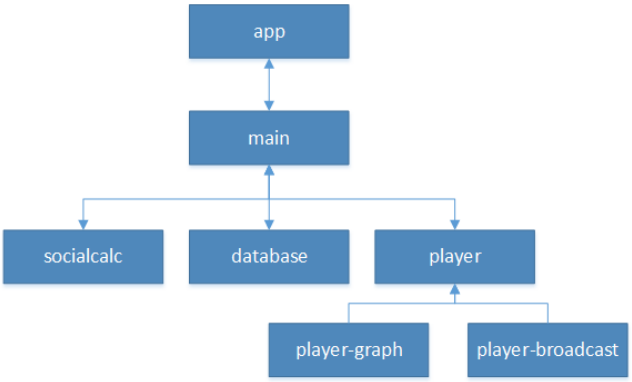
II.1.2.3 EtherCalc

EtherCalc merupakan perangkat lunak *spreadsheet online* yang *open-source* yang dikembangkan oleh Audrey Tang. EtherCalc merupakan pengembangan yang didasarkan dari perangkat lunak serupa yakni WikiCalc dan SocialCalc. WikiCalc merupakan aplikasi *spreadsheet* yang mengandalkan komputasi server untuk dapat berkolaborasi, sedangkan SocialCalc merupakan aplikasi *spreadsheet* yang menggunakan kemampuan javascript untuk melakukan komputasi pada *client-side*. EtherCalc dikembangkan di atas Node.js dan menggunakan javascript sebagai alat komputasi.

Arsitektur aplikasi EtherCalc berjalan di atas aplikasi SocialCalc. Fungsi-fungsi dasar yang ada pada *spreadsheet* hampir seluruhnya menggunakan implementasi SocialCalc. EtherCalc menjadikan aplikasi dalam bentuk Node.js, memiliki fitur kolaborasi yang lebih baik, dan beberapa fitur lain seperti pembuatan grafik data. Pada Gambar II.1 dapat dilihat interaksi antar modul yang terjadi pada EtherCalc.

Modul-modul tersebut memiliki tugas antara lain sebagai berikut:

1. Modul App



Gambar II.1: Arsitektur Module pada EtherCalc

Merupakan modul utama yang berisikan variabel konstanta dan pengaturan yang dibutuhkan oleh aplikasi.

2. Modul Main

   Modul ini bertugas sebagai modul pertama yang menerima perintah dari pengguna dan mengatur API yang ada pada aplikasi. Modul ini mengatur hubungan antara modul player, modul database, dan modul socialcalc.

3. Modul Socialcalc

   Merupakan modul yang bertugas merepresentasikan *spreadsheet* pada aplikasi. Di dalam modul ini terdapat fungsi-fungsi yang dapat digunakan untuk memanipulasi sel-sel yang ada pada *spreadsheet*.

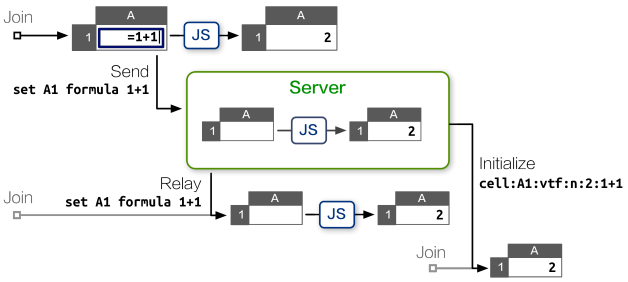
4. Modul Database

   Modul ini bertugas untuk memanipulasi basis data redis yang menyimpan perubahan-perubahan yang terjadi pada *spreadsheet*.

5. Modul Player

   Modul ini memiliki tugas untuk menjembatani *front-end* dan *back-end* dari aplikasi.

Perangkat lunak hanya akan melakukan panggilan ke *server* saat melakukan suatu aksi dan *server* yang bertugas untuk menyimpan kumpulan aksi tersebut agar pengguna lain yang ikut berkolaborasi dapat melihat *spreadsheet* yang sama satu dengan yang lain (Tang, 2014). Gambar II.2 merupakan gambaran umum cara kerja EtherCalc dalam berkolaborasi.



Gambar II.2: Ilustrasi Cara Kerja Kolaborasi pada EtherCalc

II.1.2.4 Google Sheet

Google Sheet merupakan salah satu perangkat lunak pada *office suite* miliki Google. Google Sheet dapat berjalan di atas tiga *platform* yang berbeda yakni: sebagai *web application*, Chrome apps, dan *mobile apps*. Sheet memiliki kemampuan berkolaborasi secara *real-time* dan menyediakan fitur-fitur *spreadsheet* yang ada pada umumnya. Sheet memiliki fitur *revision history* sehingga setiap orang dapat melihat perubahan yang terjadi, *add-ons* yang dapat menambahkan fitur kepada Sheet melalui program yang dibuat oleh komunitas serta fitur *chat* dengan kolaborator. Google Sheet dikembangkan menggunakan bahasa *javascript* yang memberikan kemampuan penyimpanan dan kolaborasi secara *real-time* (Google, 2016).

II.1.2.5 OnlyOffice

Merupakan perangkat lunak sebagai *service* yang dikembangkan oleh Ascensio System SIA. OnlyOffice merupakan *office suite* yang berjalan di atas *web application* yang dipadukan dengan sistem *customer relationship management*. OnlyOffice tidak hanya terdiri dari *online office editor* namun juga terdapat fitur managemen dokumen, manajemen proyek, *mail service*, serta manajemen pelanggan seperti kontak, *invoice*, *opportonities*, dan *task*. OnlyOffice ditujukan kepada bisnis yang membutuhkan perangkat lunak yang dapat mengorganisir kebutuhan bisnis di dalam satu aplikasi yang saling terintegrasi (SIA, 2016).

II.2 Penggunaan *Spreadsheet*

Spreadsheet dapat digunakan untuk melakukan kalkulasi terhadap suatu rumus atau formula yang sulit jika dikalkulasikan dengan cara manual. Di samping itu, *spreadsheet*

dapat juga digunakan untuk melakukan ramalan terhadap suatu perubahan variabel masukan. Pada perkembangannya, *spreadsheet* memiliki fitur-fitur tambahan seperti visualisasi data dan ekstraksi data penting dari kumpulan data yang ada.

Penelitian tentang penggunaan *spreadsheet* pada bisnis pernah dilakukan sebelumnya pada tahun 2014. Subjek yang diteliti adalah akuntan manajemen (Bradbard et al., 2014). Pada penelitian tersebut, didapatkan gambaran umum mengenai penggunaan *spreadsheet* secara umum. Menurut hasil penelitian tersebut beberapa fitur yang sering digunakan oleh pengguna *spreadsheet* secara terurut dari yang paling sering digunakan adalah sebagai berikut,

1. Menghitung fungsi matematika dasar (tambah, kurang, kali, bagi, dan lainnya)
2. Mengelola *worksheet* dan *workbook* (menambahkan, menghapus, merubah nama, dan lainnya)
3. Melakukan perubahan format dasar (menebalkan, memberi garis bawah, format angkat, dan lainnya)
4. Melakukan pengurutan data, penghitungan subtotal, serta meringkas data
5. Menggunakan fitur *cell addressing* baik absolut maupun relatif
6. Penggunaan fungsi kondisi (IF, COUNTIF), fungsi logika (AND, OR), fungsi pencarian (VLOOKUP, HLOOKUP), menautkan *workbook* lain, serta fungsi pembulatan (ROUND, CEILING, FLOOR)

Penggunaan *spreadsheet* sangat bergantung kepada domain bisnis atau organisasi yang menggunakan. Pada bisnis yang berorientasi komersial, *spreadsheet* dapat digunakan sebagai alat bantu perhitungan laba, pengeluaran, investasi, dan pajak. Pada organisasi-organisasi non komersial, *spreadsheet* dapat digunakan sebagai salah satu bentuk basis data yang menangani penyimpanan, pengelolaan, dan pengumpulan data yang mudah dan cepat.

II.3 Kesalahan dalam Penggunaan *Spreadsheet*

II.3.1 Kualitas Data

Kualitas Data (*Data Quality*) adalah tingkat kemampuan data untuk memenuhi kebutuhan penggunaannya (*usage requirement*) sehingga data dapat digunakan dengan baik (Khatri and Brown, 2010). Dimensi yang ada pada kualitas data dapat dibagi menjadi:

1. Akurasi, merujuk kepada tingkat kebenaran dari data.
2. Aktualitas, menunjukan bahwa data yang dicatat merupakan data terbaru.
3. Kelengkapan, menunjukkan bahwa nilai-nilai yang diperlukan tercatat (tidak hilang).
4. Kredibilitas, menunjukkan kepercayaan terhadap sumber serta isinya.

Tingkatan nilai untuk dimensi tersebut dapat berbeda pada setiap kasus yang ada. Contohnya, akurasi 85% untuk data nama dan alamat dokter merupakan nilai yang cukup baik bagi perusahaan asuransi yang menargetkan doktor sebagai konsumen potensial, namun tidak cukup baik untuk perusahaan obat yang ingin melakukan *recall* terhadap obat yang terdistribusi. Kualitas data yang buruk dapat menyebabkan akibat yang fatal dalam bisnis baik secara operasional maupun strategis.

II.3.2 Tingkat Kesalahan dalam Penggunaan *Spreadsheet*

Penelitian telah dilakukan oleh Panko (Panko, 1998) untuk mengetahui banyaknya kesalahan yang terjadi pada pengembangan *spreadsheet* terutama pada sektor bisnis. Dari penelitian ini, didapatkan bahwa 20% hingga 40% *spreadsheet* mengandung kesalahan. Pada kasus tertentu, bahkan ditemukan 90% *spreadsheet* yang diteliti memiliki kesalahan (Freeman, 1996).

Penelitian yang dilakukan oleh Panko juga menemukan 88% dari 113 *spreadsheet* yang diaudit melalui 7 lebih studi yang diteliti. Beberapa hasil yang telah di rangkum oleh penelitian tersebut mengunakan *spreadsheet* yang digunakan di dunia nyata dapat dilihat pada Tabel II.1.

Tabel II.1: Studi terhadap Kesalahan pada *Spreadsheet*

Pembuat	**Jumlah yang Diaudit**	**Rata-rata Sel**	**Persentase Error**	**Keterangan Kesalahan**
Butler (1992)	273	-	11%	Kesalahan perhitungan pada pajak
Dent (1994)	Tidak diketahui	-	30%	Menggunakan angka yang ditulis manuals yang mengakibatkan perhitungan berikutnya salah
Hicks (1995)	1	3856	100%	Kesalahan interpretasi pada data
Coopers & Lybrand (1997)	23	150+	91%	Kesalahan perhitungan yang meleset hingga 5%
Lukasic (1998)	2	2270 - 7027	100%	Kesalahan akibat melebih-lebihkan perhitungan hingga 16%
Clermont, Hanin, & Mittermeier (2002)	3	-	100%	Kesalahan akibat perhitungan sel kosong
Lawrence and Lee (2004)	30	2182	100%	Kesalahan perhitungan dan formula
Powell, Lawson, and Baker (2007)	25	-	64%	Kesalahan perhitungan dan formula
Powell, Baker & Lawson (2007)	50	-	86%	Kesalahan perhitungan dan formula

Dari kumpulan data di atas, dapat dilihat bahwa di dalam pembentukan *spreadsheet* pada bidang bisnis, tidak mungkin terlepas dari kesalahan. Dengan tingginya tingkat kesalahan ini, bisnis dapat mengalami kerugian secara material maupun moral yang cukup besar (EuSpRIG, 2010a). Hal ini mengindikasikan bahwa tingginya tingkat kesalahan harus dapat diselesaikan agar tidak terjadi kerugian di dalam penggunaan *spreadsheet* terutama dalam bisnis.

II.3.3 Tipe Kesalahan dalam Penggunaan *Spreadsheet*

Tingkat fleksibilitas *spreadsheet* yang tinggi memberikan keleluasaan kepada penggunanya untuk melakukan banyak manipulasi dan pengelolaan data. Tingginya fleksibilitas ini dapat berakibat mudahnya *human error* terjadi pada saat penggunaan *spreadsheet* yang menyebabkan terjadinya kesalahan-kesalahan pada data. Tipe-tipe kesalahan pada *spreadsheet* dapat dibagi menjadi dua jenis tipe kesalahan yakni kesalahan kuantitatif, dan kesalahan kualitatif (Panko, 1998).

II.3.3.1 Kesalahan Kualitatif

Kesalahan kualitatif merupakan kesalahan yang berhubungan dengan kualitas *spreadsheet* tersebut lebih menitikberatkan pada kebiasaan dan prosedur yang salah di dalam pembuatan *spreadsheet*. Beberapa kesalahan yang dapat diklasifikasikan sebagai kesalahan kualitatif adalah (Powell et al., 2009):

1. Melakukan *hard-code* pada suatu angka di dalam formula
2. Menggunakan formula yang panjang dalam perhitungan
3. Susunan data yang tidak direncanakan dengan baik
4. Tidak adanya dokumentasi mengenai *spreadsheet* yang dibuat

Kesalahan ini tidak langsung mengakibatkan nilai hasil keluaran yang salah namun menurunkan kualitas dari *spreadsheet* tersebut (Rajalingham et al., 2001). Di samping itu, kesalahan kualitatif dapat menyebabkan kesalahan kuantitatif terutama pada saat penggunaan fungsi analisis *what-if* pada *spreadsheet* (Panko, 1998).

II.3.3.2 Kesalahan Kuantitatif

Kesalahan ini mengakibatkan *spreadsheet* mengeluarkan hasil dan nilai yang salah di dalam operasi perhitungannya. Kesalahan jenis ini dapat dibagi menjadi empat jenis kesalahan (Howe and Simkin, 2006), yakni:

1. Kesalahan *clerical* dan non-material merupakan kesalahan yang tidak berpengaruh pada hasil dan perhitungan, namun tetap merupakan kesalahan. Contohnya adalah kesalahan ejaan.
2. *Rule violations* merupakan kesalahan yang melanggar aturan yang telah ditetapkan. Contohnya nilai seseorang tidak boleh melebihi angka 100.
3. Kesalahan *data-entry* merupakan kesalahan yang terjadi saat memasukan data. Contohnya data yang seharusnya angka, namun dimasukkan berupa teks.
4. Kesalahan *formula* merupakan kesalahan yang terjadi akibat penggunaan *formula*. Kesalahan yang terjadi bisa berupa referensi tidak sesuai, penggunaan konstanta yang salah, atau kesalahan logika.

Pada penelitian yang dilakukan oleh Howe, diketahui bahwa faktor yang berpengaruh terhadap ditemukannya kesalahan *formula* adalah umur, indeks prestasi, serta pengalaman pemrograman. Sedangkan kesalahan lain, tidak dipengaruhi oleh faktor-faktor eksternal yang diuji. (Howe and Simkin, 2006) Hal ini juga terlihat pada penelitian yang dilakuakn oleh Bishop, dimana profesional lebih baik dalam mendeteksi kesalahan *formula* hingga lebih baik 16% dibandingkan orang kebanyakan. (Bishop and McDaid, 2008) Sehingga, dapat terlihat bahwa dengan pelatihan yang baik, kesalahan formula dapat berkurang, namun kesalahan lain tidak berkurang cukup signifikan.

II.4 Data pada *Spreadsheet*

II.4.1 Tipe Struktur Data pada *Spreadsheet*

Pada penelitian yang dilakukan oleh Chen dan Cafarella, *spreadsheet* dapat dibagi menjadi 2 jenis yakni; *data frame* dan *non-data frame*. *Data frame* merupakan tipe *spreadsheet* yang terdiri dari 2 komponen utama: area nilai dan area atribut atau metadata (biasanya berada di atas dan atau di kiri area nilai). *Non-data frame* adalah tipe *spreadsheet* selain tipe *data*

frame yang telah didefinisikan sebelumnya. Tipe *non-data frame* dapat dibagi menjadi beberapa jenis yakni:

1. Relasi merupakan tipe *spreadsheet* yang dapat langsung diubah ke model relasional.
2. Formulir merupakan *spreadsheet* yang tidak ditujukan sebagai penyimpanan dan didesain untuk diisi oleh manusia.
3. Diagram yang digunakan sebagai visualisasi data, biasanya berisi banyak data tanpa skema informasi yang detil.
4. Daftar atau *List* merupakan catatan sejumlah nama atau hal (tentang kata-kata, nama orang, barang, dan sebagainya) yang disusun berderet dari atas ke bawah (dan Pengembangan Bahasa et al., 1991).
5. Jadwal merupakan *spreadsheet* digunakan sebagai pembuatan dan pengelolaan jadwal.
6. Silabus merupakan kerangka unsur kursus pendidikan, disajikan dalam aturan yang logis, atau dalam tingkat kesulitan yang makin meningkat (dan Pengembangan Bahasa et al., 1991).
7. *Scorecard* yakni suatu alat manajemen yang biasanya berguna untuk membantu manajer melacak aktivitas yang dilakukan oleh stafnya.

Penelitian ini menggunakan sampel 200 *spreadsheet* yang dilabeli oleh ahli dan didapatkan bahwa 50.5% *spreadsheet* merupakan tipe *data frame*, dimana 32.5% memiliki label atribut dibagian atas atau bawah. Sedangkan 49.5% *spreadsheet* bertipe *non-data frame* terdiri dari 22.0% relasi, 10.5% formulir, 3.5% diagram, 3% berupa *list*, dan 10.5% lainnya (Chen and Cafarella, 2013).

II.4.2 Pengolahan Data pada *Spreadsheet*

Extract-Transform-Load (ETL) adalah proses yang digunakan sebagai metode integrasi data dari beberapa sumber dan aplikasi. ETL biasanya digunakan pada saat melakukan proses *data warehouse* dimana data dari sumber eksternal diambil, lalu ditransformasikan ke bentuk yang sesuai dengan kebutuhan (di dalam prosesnya bisa terkadung pengecekan kualitas), dan memasukannya ke dalam basisdata yang telah ditentukan (Bansal, 2014).

Terdapat tiga fase pada proses ETL yakni:

1. *Extract*, fase pertama ini adalah proses yang melakukan ekstraksi data dari sumber yang dipilih. Data biasanya tersedia dalam format *flat file* seperti csv, xls, dan txt atau melalui klien RESTful.
2. *Transform*, pada fase ini data dibersihkan agar sesuai dengan skema tujuan. Beberapa cara untuk mentransformasikan data adalah dengan menormalisasi data, menghapus duplikasi, melakukan pengecekan terhadap batasan-batasan, melakukan *filtering*, melakukan pengurutan dan pengelompokan, atau fungsi-fungsi lain yang didefinisikan.
3. *Load*, pada fase ini data yang telah ditransformasikan dimasukan ke dalam *data mart* atau *data warehouse* yang ditentukan.

II.4.2.1 Metode ETL pada *Spreadsheet*

Pada penelitian yang dilakukan oleh Chen, pengolahan data pada sebuah *spreadsheet* bertipe *data frame* dapat dibagi menjadi tiga proses yakni: *frame finder*, *hierarchy extractor*, dan *tuple builder*. Proses *frame finder* dilakukan dengan cara mengidentifikasi *data frame* serta mencari lokasi dari atribut dan nilai. Pada proses *hierarchy extractor*, atribut yang ada pada *data frame* yang ditemukan dicari hirarkinya setelah itu proses *tuple builder* membentuk *tuple* relasional untuk setiap nilai yang ada. Proses ini tidak membedakan *spreadsheet* tipe *data frame* atau bukan, sehingga diasumsikan jika *tuple* yang dihasilkan memiliki kualitas yang baik, dapat dikatakan bahwa *spreadsheet* masukan bertipe *data frame* dan sebaliknya (Chen and Cafarella, 2013).

II.4.2.1.1 *Frame Finder*

Tujuan dari proses *frame finder* (Chen and Cafarella, 2013) adalah mengidentifikasi wilayah nilai dan wilayah atribut yang dapat berupa *left attribute* maupun *top attribute*. Untuk mensimplifikasi permasalahan, Chen menganggap bahwa *data frame* tidak akan berada sejajar secara horisontal, namun hanya secara vertikal. Sehingga proses ini dapat disimplifikasi menjadi labeling terhadap baris per baris. Label yang akan diberikan adalah *title*, *header*, *data*, dan *footnote*.

Pelabelan dapat dilakukan dengan algoritma *conditional random field* (CRF) karena terdapat keterkaitan antara satu baris terhadap baris yang lain dalam penggunaan baris. Contohnya, jika baris telah teridentifikasi sebagai *header*, maka besar kemungkinan bahwa baris selanjutnya adalah *data* atau *header*. CRF memiliki kemampuan untuk melakukan *machine learning* yang memperhitungkan label pada elemen sebelumnya.

II.4.2.1.2 *Hierarcy Extractor*

Proses *hierarcy extractor* (Chen and Cafarella, 2013) bertujuan untuk mendapatkan hirarki dari atribut-atribut yang ada. Masukan dari proses ini adalah *data frame* dengan *top attribute* dan *left attribute* dan keluarannya berupa hirarki untuk masing-masing atribut atas dan kiri tersebut. Proses ini dapat dilakukan melalui dua algoritma: *classification* dan *enforced-tree classification*.

Classification dilakukan dengan cara berbeda untuk *left attribute* dan *top attribute*. Pada *left attribute*, klasifikasi dapat dilakukan dengan dua cara: pengecekan terhadap *formatting* pada sebuah sel dan kedekatan sel secara geometris. Semakin mirip *formatting* sebuah sel, maka semakin mungkin bahwa kedua sel bukan merupakan pasangan *parent* dan *child* dan semakin dekat sel secara geometris, kemungkinan kedua sel meruapakan pasangan *parent* dan *child* semakin besar. Sedangkan pada *top attribute* dapat dilakukan pengecekan posisi antar baris atribut bagian atas.

Kelemahan pada metode klasifikasi sebelumnya adalah terdapat kemungkinan tidak dapat dibentuk pohon dari hasil klasifikasi. *Enforced-tree classification* mencoba untuk menyelesaikan permasalahan ini dengan dua langkah tambahan yakni: memastikan bahwa suatu atribut hanya dapat memiliki satu *parent* dimana yang terpilih menjadi *parent* adalah atribut dengan probabilitas tertinggi, dan memastikan bahwa tidak ada *cycle* yang terbentuk dengan cara menghapus keterhubungan dengan nilai probabilitas terkecil. Klasifikasi ini tetap menggunakan metode klasifikasi fitur yang dilakukan pada algoritma *classification*.

II.4.2.1.3 *Tuple Builder*

Proses ini dilakukan dengan cara mengiterasi setiap *value (v)* dan mencari atribut akar pada atribut bagian kiri dan atas dari *v*. Setelah dibentuk *relational tuple* untuk nilai

v dengan atribut bagian kiri dan atas tersebut. Tingkat akurasi dari proses ini sangat bergantung dari dua proses sebelumnya.

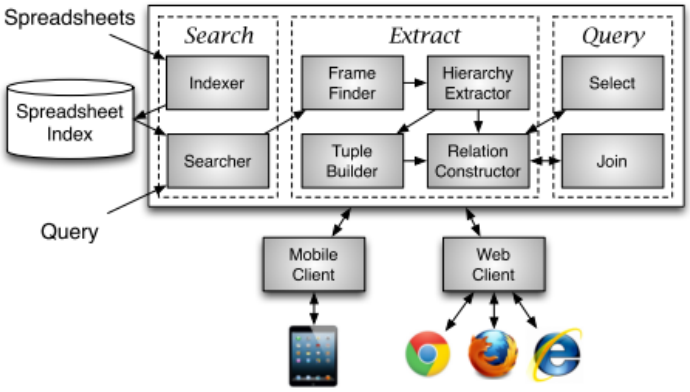
II.5 Studi dan Penelitian Terkait

II.5.1 *Senbazuru: A Prototype Spreadsheet Database Management System*

Senbazuru (Chen et al., 2013) merupakan prototipe yang dikembangkan dengan tujuan untuk mempermudah pencarian, pengaksesan, pengubahan, dan melakukan *query* terhadap *spreadsheet*. Pengembangan ini ingin menyelesaikan permasalahan dimana data *spreadsheet* sangat tersebar diberbagai tempat sehingga untuk mendapatkan informasi yang diinginkan atau membandingkan antar informasi di dalam beberapa *spreadsheet* sangatlah sulit.

Di dalam pengembangan prototipe ini, kesulitan teknikal yang harus dihadapi adalah proses ekstraksi dan perbaikan data. Di dalam melakukan ekstraksi data, harus dilakukan beberapa proses berikut: mendeteksi mana atribut dan nilai, mengidentifikasi hirarki atribut, membentuk *relational tuple*, dan membentuk *tuple* tersebut menjadi tabel relasional. Dari hasil dari ekstrasi ini, masih sangat mungkin memiliki kesalahan sehingga proses perbaikan diperlukan di dalam pembentukan tabel relasional ini. Proses perbaikan pada protitipe ini dilakukan manual dengan bantuan pengguna.

II.5.1.1 Arsitektur Sistem



Gambar II.3: Arsitektur Sistem Senbazuru (Chen et al., 2013)

Spreadsheet Database Management System (SSDBMS) yang dikembangkan pada prototipe ini memiliki tiga proses utama yakni: *search*, *extract*, dan *query*. Proses pencarian

dilakukan terhadap repositori *spreadsheet* yang ada di internet, lalu data *spreadsheet* tersebut diekstraksi dan dijadikan tabel relasional, dan setelah itu pengguna dapat melakukan *query* terhadap tabel yang telah dibentuk. Gambaran arsitektur Senbazuru dapat dilihat pada Gambar II.3. Arsitektur ini terbagi menjadi tiga modul utama yakni Search, Extract, dan Query.

II.5.1.1.1 *Search*

Komponen pencarian ini memudahkan pengguna untuk menemukan dataset yang tepat menggunakan bantuan internet. Saat prototipe ini dikembangkan, Senbazuru telah mengindeks 1800 *spreadsheet* yang didapatkan dari U.S. Census Bureau. Pengindeksan menggunakan bantuan *library* Python yakni xlrd untuk mengekstraksi teks dari sel lalu menggunakan Apache Lucene untuk melakukan indeks pada teks. Pencarian menggunakan metode *term frequency–inverse document frequency* (TF-IDF) untuk mendapatkan relevansi dokumen.

II.5.1.1.2 *Extract*

Proses ekstraksi data pada *spreadsheet* dilakukan melalui empat tahapan yakni:

1. *Frame Finder*

   Tahap ini dilakukan untuk mencari *frame* pada *spreadsheet* bertipe *data frame*. Dengan menggunakan algoritma *conditional random field* (CRF) untuk memberikan label pada setiap baris yang tidak kosong pada *spreadsheet*. Tahap ini akan menghasilkan *data frame* yang selanjutnya akan digunakan pada tahap selanjutnya, baris lain yang dianggap bukan *data frame* akan diabaikan.

2. *Hierarchy Extractor*

   Tahap selanjutnya adalah ekstraksi hirarki pada wilayah atribut dari *data frame* yang ditemukan. Pada setiap atribut, akan dicari atribut mana yang dideskripsikan oleh atribut lainnya dan seterusnya hingga terbentuk hirarki dari atribut-atribut yang ada. Kesalahan pada pembentukan hirarki sangat mungkin terjadi sehingga pengguna akan diberikan kemampuan untuk memperbaiki hirarki yang salah pada bagian *repair interface*. Setelah perbaikan dilakukan oleh pengguna, Senbazuru akan menjalankan kembali CRF untuk melakukan pembelajaran terhadap label baru yang

diberikan.

3. *Tuple Builder*

   Bagian ini melakukan pembentukan *tuple* antara wilayah nilai dan wilayah atribut yang sesuai.

4. *Relation Constructor*

   Tahap ini melakukan transalasi dari *tuple* yang terbentuk menjadi tabel relasional dengan cara membentuk kluster terhadap atribut yang satu jenis. Contohnya, terdapat atribut *Male*, *total*, dan *Female*, ketiga atribut tersebut memiliki jenis yang sama sehingga harus digabungkan menjadi satu kolom yakni *gender*. Pada Senbazuru, teknik pengklusteran ini menggunakan bantuan koleksi skema dari Freebase dan YAGO.

II.5.1.1.3 *Query*

Setelah proses sebelumnya selesai, maka pengguna dapat memasukan perintah relasional terhadap data *spreadsheet* yang telah diubah menjadi tabel relasional. Pada prototipe yang dikembangkan, perintah yang diimplementasikan adalah *join* dan *select*.

II.5.1.2 Hasil Penelitian

Senbazuru merupakan prototipe untuk manajemen basis data berbasis *spreadsheet* yang dapat melakukan pencarian data pada internet melalui kata kunci yang diberikan. Prototipe ini berhasil melakukan ekstraksi data secara otomatis walaupun tidak terlepas dari kesalahan. Kesalahan yang terjadi masih harus seringkali dilakukan perbaikan secara manual. Namun dengan penggunaan algoritma CRF, protitipe dapat mengurangi kesalahan yang terjadi. Prototipe ini juga ditujukan sebagai demo kepada peserta konferensi VLDB dan diharapkan dapat menarik perhatian komunitas basis data.

II.5.2 *Spreadsheet As a Relational Database Engine*

Penelitian (Tyszkiewicz, 2010) pernah dilakukan terhadap pembuatan *spreadsheet* menjadi mesin basis data relasional. Penelitian ini dilatarbelakangi dengan tingginya penggunaan *spreadsheet* pada banyak bidang dan kurangnya kualitas data yang ada di dalamnya yang dapat menyebabkan kesalahan-kesalahan terjadi pada perhitungan dan prediksi. Solusi

yang dipaparkan pada penelitian ini adalah dengan menggabungkan *spreadsheet* dan *database engine* dengan menggunakan formula sebagai ganti dari *SQL query*.

II.5.2.1 Cara Kerja Sistem

Pada implementasinya, sebuah *workbook* akan memiliki satu *worksheet* untuk setiap tabel data dan satu *worksheet* untuk setiap *view* pada basis data. Dengan menggunakan *external compiler* yang menerima masukan berupa SQL yang akan mengubahnya ke dalam bentuk *spreadsheet*. Program tersebut akan mengubah SQL menjadi beberapa formula yang diterima oleh *spreadsheet* tersebut.

Terdapat dua bagian utama pada *spreadsheet* hasil implementasi yakni: tabel data dan bagian *view*. Bagian tabel data adalah tempat pengguna untuk memasukkan, mengubah, serta menghapus data. Secara teori, bagian tabel data tidak memiliki formula, namun di dalam implementasinya mengikuti implementasi SQL dimana perlu dilakukan validasi data dan verifikasi terhadap *primary key*, *foreign key*, dan batasan lain yang ada diperintah *create table*.

Bagian *view worksheet* tidak dapat diubah oleh pengguna dan berisikan formula-formula yang independen terhadap data yang dimasukkan oleh pengguna. Di samping itu, bagian *view* berisikan kolom-kolom yang berguna sebagai *intermediate result* yang selanjutnya akan digunakan oleh formula lain. Pada awalnya sel-sel akan berisikan "" (*string* kosong) yang merepresentasikan sel yang belum digunakan.

II.5.2.2 Hasil Penelitian

Formula pada excel dilakukan menggunakan *linear scan* dan hal ini dapat mengurangi kinerja. Beberapa cara untuk meningkatkan kinerja adalah mengeksploitasi *lazy evaluation* dari *if statement*, mengkomputasi hanya beberapa sel tetangga yang berkaitan, serta menggunakan file atau *workbook* lain untuk membagi *query* dan membangkitkannya ketika dibutuhkan.

Pada tes kinerja yang dilakukan pada penelitian ini dapat disimpulkan bahwa untuk operasi dasar dan *query* sederhana penggunaan *spreadsheet* dapat digunakan dengan baik dengan waktu yang cukup cepat. Tingkat efektifitas penggunaan pada arsitektur ini cukup

rendah, namun hasil tes tersebut menunjukan bahwa arsitektur ini memiliki potensial yang dapat dikembangkan.

BAB III

ANALISIS MASALAH PENGUMPULAN DATA PADA *SPREADSHEET*

Pada bab ini diuraikan analisis persoalan pengumpulan data pada *spreadsheet* yang telah diuraikan pada Bab I. Hasil dari bab ini digunakan untuk merancang kakas yang akan diimplementasikan seperti yang dijelaskan pada Bab IV.

III.1 Permasalahan Pengumpulan Data Pada *Spreadsheet*

Pada Subbab I.2 telah dijelaskan bahwa terdapat tiga permasalahan yang muncul dalam pengumpulan data menggunakan media *spreadsheet* yakni:

1. Kolaborasi

   Dalam pembuatan suatu *spreadsheet*, kadang dibutuhkan banyak pihak untuk dapat melengkapi seluruh data yang dibutuhkan. Hal ini membuat pengembangan dan pengumpulan data membutuhkan kontribusi banyak orang (Panko, 1998). Untuk itu dibutuhkan fitur kolaborasi pada aplikasi *spreadsheet* agar banyak pihak dapat melakukan pengumpulan data dan pengeditan secara bersamaan.

2. Validasi

   Data yang dikumpulkan pada suatu *spreadsheet*, sangat rentan akan kesalahan. Bahkan pada *spreadsheet* yang dibuat dengan sangat hati-hati, masih dapat ditemui sekitar 1 persen atau lebih kesalahan pada formula yang dibuat (Panko, 1998). Sehingga dibutuhkan mekanisme tambahan pada saat pengumpulan data untuk melakukan validasi terhadap masukan.

3. Penyimpanan Data

   Setelah data dikumpulkan, permasalahan selanjutnya yang muncul adalah penyimpanan data tersebut. Data biasanya akan tetap tersimpan dalam bentuk berkas *spreadsheet*, namun bentuk ini sulit untuk digunakan oleh aplikasi lain. Bentuk yang biasanya digunakan oleh aplikasi lain untuk dapat mengambil dan

menyimpan data adalah bentuk relasional. Sehingga dibutuhkan mekanisme penyimpanan data ke dalam bentuk relasional agar data dapat dengan mudah digunakan pada aplikasi lain.

Ketiga permasalahan tersebut yang dijadikan dasar pengembangan kakas pada Tugas Akhir ini dan akan dibahas pada subbab-subbab berikutnya.

III.2 Analisis Rancangan Kakas pada Aplikasi *Spreadsheet*

Dari permasalahan yang telah didefinisikan pada subbab III.1, dapat ditentukan beberapa hal yang harus dianalisis dalam merancang kakas pada aplikasi *spreadsheet* ini, diantaranya adalah:

1. Penentuan aplikasi *spreadsheet* yang akan dikembangkan dan ditambahkan fiturnya. Aplikasi harus memiliki fitur kolaborasi sehingga banyak pengguna dapat mengakses suatu *spreadsheet* secara bersamaan.
2. Penanganan konflik yang terjadi pada saat kolaborasi.
3. Jenis *spreadsheet* yang menjadi fokus utama untuk dapat ditangani oleh kakas.
4. Pemilihan metode interaksi kakas pada aplikasi *spreadsheet* dimana pengguna dapat membentuk struktur tempat pengguna lain memasukkan masukan. Contohnya dalam bentuk tabel atau formulir. Pengguna juga melengkapi domain dan batasan data yang dimasukkan.
5. Setelah pengguna selesai merancang struktur masukan, diperlukan penentuan bagian label dan data yang berkaitan dengan label tersebut. Bagian label dan data akan dijadikan menjadi bentuk relasional yang dapat disimpan dalam basis data.
6. Setelah label dan data ditentukan, tabel harus direpresentasikan dalam bentuk relasional sehingga dapat dimasukkan ke dalam basis data.
7. Setelah data dikumpulkan, data perlu divalidasi dan dicek kesesuaiannya sesuai dengan batasan yang diberikan sehingga memenuhi domain yang ditentukan pada basis data.
8. Setelah proses validasi terpenuhi, diperlukan cara penyimpanan data sesuai dengan relasi *tuple* yang telah ditentukan.

9. Interaksi antar *spreadsheet* yang ingin memasukkan data pada suatu tabel yang sama perlu ditangani dan ditentukan aturannya.

Pada subbab-subbab selanjutnya, akan dibahas analisis untuk kelima poin di atas yang selanjutnya akan dijadikan dasar rancangan kakas yang akan dibuat pada aplikasi *spreadsheet*.

III.3 Analisis Perbandingan Aplikasi *Spreadsheet* yang Dikembangkan

Dari studi yang telah dilaksanakan pada subbab II.1.2, aplikasi *spreadsheet* dapat dibagi berdasarkan konektivitasnya yakni *offline spreadsheet* dan *online spreadsheet* dan dapat juga dibagi lagi berdasarkan keterbukaan dari *source code* yakni *open source* dan *closed source*. Pada Tabel III.1 dijabarkan parameter yang mendasari pemilihan aplikasi yang akan dikembangkan.

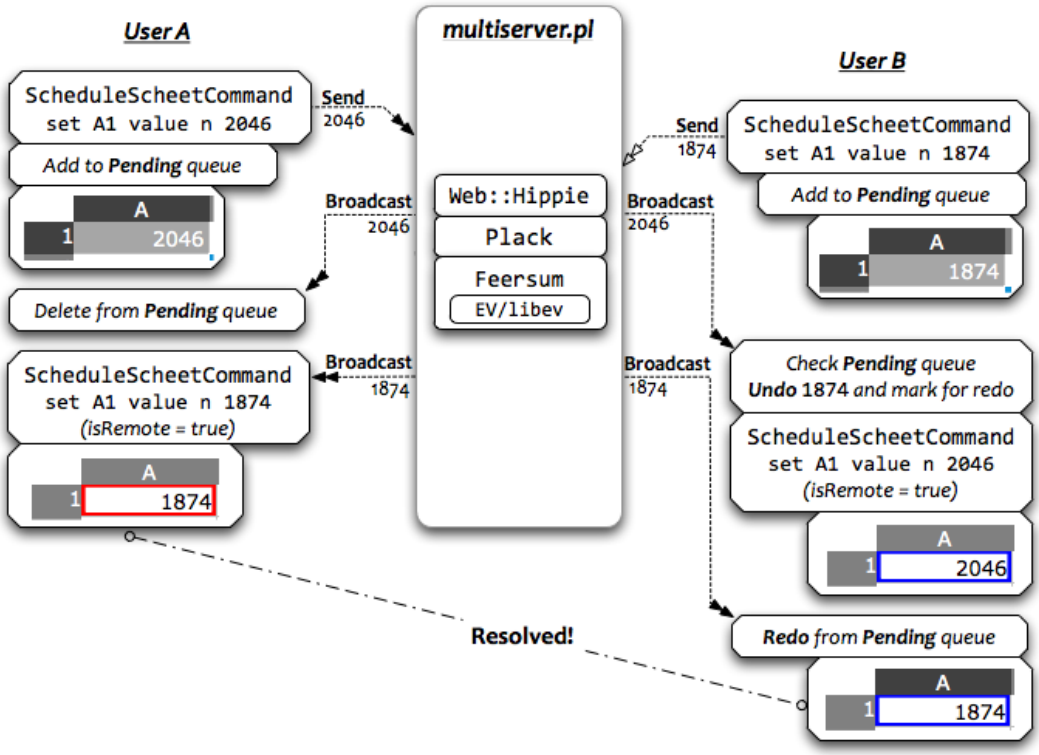
Tabel III.1: Perbandingan Aplikasi *Spreadsheet*

Parameter	**Offline Application**	**Web/Online Application**
Aksesibilitas	Dibutuhkan instalasi aplikasi / plugin yang bersangkutan.	Dapat menggunakan web browser yang tersedia.
Kolaborasi	Tidak tersedia secara online dan kolaboratif secara *default*.	Secara *default*, dapat diakses konkuren dan kolaboratif.
Fitur	Sudah banyak jenis formula yang didukung.	Tidak semua formula yang ada didukung.

Dari perbandingan di atas, diambil solusi dengan menggunakan aplikasi *spreadsheet* berbasis web yang dapat diakses secara konkuren dan memiliki *portability* yang lebih baik. Di samping itu, untuk dapat mengembangkan fitur aplikasi dengan lebih baik, akan dipilih aplikasi yang berbentuk *open source*. Alasan lain dipilihnya *open source* adalah aplikasi berbasis *open source* dapat dipasang pada *platform* yang diinginkan oleh pengguna. Pengguna atau perusahaan yang menggunakan aplikasi dapat membangun infrastrukturnya sendiri sehingga dapat mengatur privasi dan performa aplikasi sesuai dengan keinginan pengguna. Sehingga dipilih aplikasi yang memiliki tipe *online spreadsheet* dan *open source* yaitu EtherCalc. Fitur kolaborasi yang dibutuhkan akan

ditangani oleh aplikasi EtherCalc ini dan bukan merupakan bagian dari fitur yang akan dikembangkan.

III.4 Analisis Penanganan Konflik saat Kolaborasi



Gambar III.1: Mekanisme Penanganan Konflik (Tang, 2014)

Penanganan konflik pada saat lebih dari satu pengguna memasukkan data pada suatu *spreadsheet* yang sama telah ditangani oleh EtherCalc menggunakan mekanisme *conflict resolution* yang berasal dari SocialCalc. Mekanisme penanganan konflik menggunakan fitur *undo*/*redo* yang telah ada. Gambaran mekanisme tersebut dapat dilihat pada Gambar III.1. Berdasarkan gambaran tersebut dan *website* resmi dari EtherCalc, langkah-langkah penanganan konflik yang terjadi adalah sebagai berikut:

1. Ketika klien mengirimkan suatu perintah. Perintah tersebut dimasukkan ke *pending queue*.
2. Ketika klien menerima *broadcast* perintah, akan dilakukan pengecekan *pending queue*.
3. Jika *pending queue* kosong, maka perintah akan langsung dijalankan.

4. Jika perintah baru sama dengan *pending queue*, perintah akan dijalankan dan dihapus dari *pending queue*.
5. Jika tidak, maka dilakukan pengecekan apakah terjadi konflik.
6. Jika terjadi konflik, maka akan dilakukan perintah *Undo* dan ditandai untuk dilakukan *Redo* pada saat konflik terselesaikan.
7. Setelah semua konflik terselesaikan, jalankan perintah seperti biasa.
8. Jika masih terdapat perintah yang ditandai untuk dilakukan *Redo*, maka perintah tersebut dijalankan kembali.

III.5 Analisis Jenis *Spreadsheet* yang Ditangani

Pada Subbab II.4 telah dijelaskan jenis-jenis *spreadsheet* yang biasanya dibuat. Persentase jenis *spreadsheet* terbesar yang dibuat berjenis *data frame* dengan persentase 50.5%. *Data frame* merupakan tipe *spreadsheet* yang terdiri dari 2 komponen utama: area nilai dan area atribut atau metadata (biasanya berada di atas dan atau di kiri area nilai). Karena tingginya angka penggunaan menggunakan tipe ini, kakas akan melakukan penanganan pada *spreadsheet* bertipe *data frame*.

Persentase kedua ditempati oleh jenis *spreadsheet* bertipe relasi yang memiliki persentase 22.0%. Tipe relasi ini merupakan tipe *spreadsheet* yang dapat langsung diubah ke model relasional. Biasanya berupa satu *header* dan kumpulan baris data. Dengan tingginya persentase tipe ini serta miripnya tipe relasi dan *data frame*, tipe ini juga dipilih untuk dapat ditangani oleh kakas.

Tipe lainnya yang teridentifikasi adalah tipe formulir, diagram, dan jadwal. Ketiga jenis ini memiliki penanganan yang berbeda-beda dan tidak memiliki sifat yang sama dengan tipe *data frame* dan relasi. Sehingga, tipe ini tidak dijadikan fokus dalam pengembangan kakas untuk dapat ditangani. Dengan menangani hanya tipe *data frame* dan relasi diharapkan telah dapat menangani 72.5% kasus penggunaan *spreadsheet* yang biasanya ditemui.

III.6 Analisis Metode Interaksi Pengguna

Di dalam pembangunan perangkat lunak *spreadsheet* untuk mengurangi kesalahan, dapat diidentifikasikan dua metode interaksi yang dapat diimplementasi. Model interaksi yang

pertama adalah menggunakan formulir sebagai basis masukan data dan metode yang kedua adalah menggunakan aplikasi *spreadsheet* secara langsung sebagai media input data.

III.6.1 Berbasis Formulir

Metode interaksi ini menggunakan *spreadsheet* sebagai tempat perancangan formulir. Pembuat formulir akan menentukan area label dan input secara manual serta diidentifikasikan berdasarkan warna atau properti lain yang unik pada sel tersebut. Formulir akan dibangkitkan oleh aplikasi agar menjadi bentuk HTML dan terhubung ke basis data. Pengisian data oleh pengguna dilakukan melalui formulir yang dibangkitkan dan dapat diakses melalui web. Beberapa cara dapat dilakukan untuk mengimplementasikan teknik ini antara lain, mengembangkan dari aplikasi *spreadsheet* yang telah ada menggunakan *plugin* atau membuat aplikasi baru yang dapat melakukan konversi otomatis menjadi formulir.

III.6.2 Berbasis *Spreadsheet*

Metode interaksi berbasiskan *spreadsheet* menggunakan antarmuka yang telah disediakan oleh aplikasi. Penggunaannya dilakukan dengan membuat format pengisian seperti membuat tabel pada *spreadsheet* pada umumnya. Pada tabel harus terdapat label dan data sehingga metadata minimal yang dibutuhkan dapat dicapai. Fitur-fitur yang ada pada *spreadsheet* juga tetap dapat digunakan saat pembuatan tabel yang diinginkan. Dari pembuatan tabel tersebut dilakukan identifikasi label dari data tersebut untuk selanjutnya diproses melalui penyaringan masukan dan dimasukan ke tabel yang sesuai yang ada di basis data. Untuk mengimplementasikan teknik ini harus mengubah kode pada program *spreadsheet* atau mengekstensi fitur yang ada menggunakan *plugin*.

III.6.3 Perbandingan Metode Masukan

Kedua metode interaksi pengguna tersebut memiliki beberapa perbedaan dan efek terhadap penggunaannya baik bagi aplikasi maupun pengguna. Pada Tabel III.2 dijabarkan perbandingan antara kedua metode interaksi pengguna tersebut.

Tabel III.2: Perbandingan Metode Interaksi Pengguna

Parameter	**Berbasis Formulir**	**Berbasis *Spreadsheet***
Fungsionalitas	Berhasil memisahkan bagian operasional dan data. Pengguna hanya memodifikasi dan melakukan input pada bagian data. Bagian operasional hanya dapat dimodifikasi oleh pembuat formulir tersebut.	Jika tidak menggunakan proteksi terhadap sel yang dapat ditulis, tidak terjadi pemisahan antara data dan operasional sehingga beberapa kesalahan yang terjadi pada saat input masih dapat terjadi.
Teknologi	Dibutuhkan adanya algoritma tambahan untuk menangani formulir dan masukannya, serta melakukan konversi dan *parsing* dari sel pada *spreadsheet* ke dalam bentuk formulir.	Dibutuhkan algoritma dan logika *parsing* yang lebih detil dan kompleks dalam menangani kasus-kasus table tertentu.
Antarmuka	Menggunakan antarmuka formulir yang kaku terurut dari atas ke bawah serta tidak dapat melihat hasil masukan secara langsung.	Struktur tabel atau masukan dapat disesuaikan dengan kebutuhan dan tidak perlu mempelajari hal lain jika sebelumnya telah menggunakan *spreadsheet* sebagai media untuk memasukan data.

Pada Tugas Akhir ini, akan dipilih metode masukan pengguna dengan berbasis *spreadsheet* sehingga pengguna dapat dengan lebih mudah di dalam menggunakannya karena tidak memerlukan pembelajaran kembali di dalam menggunakan aplikasi. Di samping itu, data yang dimasukan lebih mudah untuk dilihat secara menyeluruh dibandingkan dengan berbasis formulir yang hanya dapat menerima satu masukan dalam suatu formulir.

III.7 Analisis Penentuan Bagian Data dan Label

Pada pembuatan *spreadsheet* pada umumnya, didapatkan bahwa kebanyakan *spreadsheet* pada umumnya berbentuk tabel yang terdiri dari dua unsur utama yakni data dan label atau disebut juga tipe *data frame* (Chen and Cafarella, 2013). Bagian data merupakan bagian yang biasanya dinamis dan merupakan masukan pengguna. Bagian label merupakan bagian yang memberikan keterangan dan konteks mengenai data yang dimaksud. Pada Gambar III.2 dapat dilihat bahwa area dengan nomor 1 disebut sebagai label yang menjelaskan data dibawahnya yakni pada area nomor 2.

NIM	Nilai 1	Nilai 2	UTS	UAS	Nilai Akhir	Nilai
13513001	90	88	76	80	82.18181818	A
13513002	50	60	82	67	66.72727273	BC
13513003	0	56	77	65	54.81818182	C
13513004	35	55	50	70	55.45454545	C
13513005	90	100	100	110	101.8181818	A

1

2

Gambar III.2: Contoh *Data Frame* Sederhana

III.7.1 Secara Manual

Penentuan label dan data dilakukan oleh pengguna secara langsung saat pembuatan tabel. Pengguna sendiri yang menentukan area mana yang merupakan label dan data mana yang dijelaskan oleh label tersebut. Dengan metode manual ini, pengguna dapat menyesuaikan bentuk tabel sesuai keinginan mereka. Metode ini menyerahkan sepenuhnya tanggungjawab keterhubungan sel label dan sel data kepada pengguna. Metode yang digunakan adalah menerima masukan dari pengguna berupa suatu *script* yang mendefinisikan letak tabel, *header*, dan data pada *spreadsheet*.

III.7.2 Secara Otomatis

Penentuan secara otomatis dimulai dari pencarian tabel pada *spreadsheet*. Hal ini dilakukan karena tabel dapat lebih dari satu pada sebuah *spreadsheet*. Penentuan tabel dilakukan menggunakan algoritma k-Nearest Neighbour (kNN) sehingga ditemukan kedekatan antar sel, lalu mengelompokkannya sehingga diketahui koordinat sel mana saja yang merupakan suatu kesatuan tabel. Setelah tabel ditemukan, pencarian label dan data dapat menggunakan algoritma seperti yang telah dijelaskan pada Subbab II.4.2.1.

Mekanisme yang akan diimplementasi pada kakas ini hanya bagian *frame finder* yang bertujuan untuk mengidentifikasi wilayah nilai (data) dan wilayah atribut (label) yang

dapat berupa *left attribute* maupun *top attribute*. Teknik *frame finder* menggunakan algoritma pembelajaran mesin *conditional random field* (CRF) yang dapat mendeteksi kemungkinan jenis data dengan mengetahui urutan kemunculan dan ketetanggaan antar data. Algoritma ini dilatih menggunakan data latih yang berasal dari *Statistical Abstract of the United States* (SAUS) 2010 (Chen, 2015). Dengan ditemukannya tabel-tabel pada *spreadsheet* beserta label dan data yang ada pada tabel tersebut, seluruh *metadata* tersebut ditampilkan dalam bentuk *metadata table*.

III.7.3 Perbandingan Metode Penentuan

Untuk mengetahui perbandingan kedua metode, pada Tabel III.3 dijelaskan kelebihan dan kekurangan dua metode yang telah disebutkan sebelumnya.

Tabel III.3: Perbandingan Metode Penentuan Data dan Label

Keterangan	**Manual oleh Pengguna**	**Otomatis**
Kelebihan	Tingkat akurasi lebih tinggi karena data dan label yang ditentukan sesuai dengan keinginan pengguna.	Kenyamanan dalam penggunaan karena pengguna tidak perlu melakukan interferensi tambahan. Di samping itu, kemungkinan data dan label yang diambil dapat diubah ke dalam bentuk *tuple* relasional lebih tinggi.
Kekurangan	Interferensi pengguna yang banyak dan mungkinnya data dan label tidak dapat dijadikan bentuk *tuple* relasional.	Algoritma ini hanya optimal jika digunakan pada tabel yang terurut vertikal.

Dari perbandingan di atas, dapat dilihat terdapat kekurangan dan kelebihan di dalam menggunakan salah satu metode tersebut. Berdasarkan hal tersebut, yang akan dipilih sebagai metode penentuan data dan label adalah gabungan dari kedua metode tersebut. Kakas awalnya akan melakukan *parsing* terhadap struktur yang telah dibuat oleh pengguna, kemudian pengguna dapat melihat hasil dari *parsing* tersebut sehingga pengguna dapat memperbaiki jika terdapat hasil pelabelan yang salah. Dengan metode

gabungan ini diharapkan dapat meningkatkan akurasi dan mengurangi kesalahan *parsing* yang terjadi namun tetap memberikan kenyamanan terhadap pengguna serta menjaga agar hasil pelabelan tetap dapat diubah ke dalam bentuk *tuple* relasional.

III.8 Analisis Representasi Tabel pada *Spreadsheet*

Dalam pembuatan *spreadsheet*, terdapat beberapa jenis tabel yang biasanya dibuat oleh pengguna dalam merepresentasikan data yang ingin disimpan. Pada jenis-jenis tabel tertentu, dibutuhkan transformasi bentuk tabel tersebut agar dapat dimasukkan ke dalam basis data bentuk relasional. Tipe-tipe tabel yang biasanya muncul di dalam pembuatan *spreadsheet* beserta teknik transformasi yang akan digunakan adalah sebagai berikut:

1. Tabel dengan *header* berada di atas data

NIM	Nilai 1	Nilai 2	UTS	UAS	Nilai Akhir	Nilai
13513001	90	88	76	80	82.18181818	A
13513002	50	60	82	67	66.72727273	BC
13513003	0	56	77	65	54.81818182	C
13513004	35	55	50	70	55.45454545	C
13513005	90	100	100	110	101.8181818	A

Gambar III.3: Tabel dengan *Header* Berada Di atas Data

   Representasi tabel jenis ini dapat langsung dibuat ke dalam bentuk relasional tanpa harus mengubah struktur tabel. Contoh tabel jenis ini dapat dilihat pada Gambar III.3. Bagian *header* merupakan baris pertama pada tabel, dan baris kedua hingga seterusnya merupakan bagian data. Bentuk ini tidak membutuhkan transformasi agar dapat dimasukkan ke dalam basis data bentuk relasional.

2. Tabel dengan *header* berada di kiri dan atas data

	O2	N2	CO2	Ar	H20
Kadar di permukaan laut	21%	78%	0.04%	0.93%	1%
Kadar di bawah laut	0.04%	0%	0%	0%	100%
Kadar di ketinggian 10 km	10%	80%	1%	1%	0.80%
Kadar di stratosphere	5%	90%	1%	1%	0.40%

Gambar III.4: Tabel dengan *Header* Berada di Kiri dan Atas Data

   Tabel dapat memiliki lebih dari satu *header* di lokasi yang berbeda. Contoh tabel jenis ini dapat dilihat pada Gambar III.4. Representasi tabel dimana *header* bukan hanya di bagian atas saja, maka *header* yang dianggap hanya yang di bagian atas.

Bagian yang akan dianggap sebagai *header* adalah kolom yang keseluruhannya berisi *header* dan berada di atas data. Pada contoh di atas, *header* adalah baris pertama. Sedangkan *header* di kiri tabel yang satu baris dengan data akan dianggap data dengan nama *header* yang dapat ditentukan oleh pengguna. Dengan demikian, representasi tabel tersebut akan menjadi mirip dengan tipe tabel sebelumnya. Representasi tabel setelah transformasi dapat dilihat pada Gambar III.5.

Letak pengukuran	O2	N2	CO2	Ar	H20
Kadar di permukaan laut	21%	78%	0.04%	0.93%	1%
Kadar di bawah laut	0.04%	0%	0%	0%	100%
Kadar di ketinggian 10 km	10%	80%	1%	1%	0.80%
Kadar di stratosphere	5%	90%	1%	1%	0.40%

Gambar III.5: Transformasi Tabel dengan *Header* Berada di Kiri dan Atas Data

3. Tabel dengan *header* yang berhirarki

Record Number	Client ID	Sales			
		2011	2012	2013	2014
1	ID000123	802938	123513	625430	234500
2	ID000125	234324	342000	342600	324200
3	ID000222	237899	234520	245100	235000
4	ID000245	345690	2347	34528	34526

Gambar III.6: Tabel dengan *Header* yang Berhirarki

Tabel dapat memiliki *header* yang berhubungan satu dengan yang lain dan membentuk hirarki seperti pada contoh Gambar III.6. Tabel dengan *header* yang memiliki hirarki akan tetap dianggap memiliki satu baris *header*. Transformasi yang dilakukan adalah dengan menggabungkan kata-kata pada tiap *header* dan dijadikan sebagai satu nama *header* dan secara *default* dipisahkan dengan tanda '_' untuk masing-masing *header*. Penggunaan tanda '_' dan bukan spasi dilakukan agar pengguna dapat membedakan bagian yang merupakan hasil gabungan. Hasil transformasi tabel tersebut dapat dilihat pada Gambar III.7.

4. Tabel dengan data yang digabung

   Tabel dapat juga memiliki penggabungan sel pada bagian data seperti yang dapat dilihat pada contoh Gambar III.8. Penggabungan sel ini biasanya digunakan sebagai

Record	Client ID	Sales_2011	Sales_2012	Sales_2013	Sales_2014
1	ID000123	802938	123513	625430	234500
2	ID000125	234324	342000	342600	324200
3	ID000222	237899	234520	245100	235000
4	ID000245	345690	2347	34528	34526

Gambar III.7: Transformasi Tabel dengan *Header* yang Berhirarki

Kelompok	NIM	Nilai			
		Laporan	Program	Demo	Bonus
CIMOL TEAM	13042	20%	80%	0%	5%
	13022	30%	20%	45%	5%
	13056	50%	0%	55%	5%
Suatu Tim	13099	33%	25%	50%	0%
	13600	33%	25%	50%	0%
	13400	33%	50%	0%	0%

Gambar III.8: Tabel dengan Data yang Digabung

penanda bahwa data pada setiap baris atau kolom yang digabung adalah sama. Sehingga, transformasi yang akan dilakukan adalah dengan memecah sel yang digabungkan tersebut dan mengisi setiap sel dengan konten yang sama dengan sel yang digabungkan. Hasil transformasi tabel dapat dilihat pada Gambar III.9.

Kelompok	NIM	Nilai_Laporan	Nilai_Program	Nilai_Demo	Nilai_Bonus
CIMOL TEAM	13042	20%	80%	0%	5%
CIMOL TEAM	13022	30%	20%	45%	5%
CIMOL TEAM	13056	50%	0%	55%	5%
Suatu Tim	13099	33%	25%	50%	0%
Suatu Tim	13600	33%	25%	50%	0%
Suatu Tim	13400	33%	50%	0%	0%

Gambar III.9: Transformasi Tabel dengan Data yang Digabung

Setelah ditransformasikan, tabel akan dapat dengan mudah dijadikan *tuple* relasional sehingga dapat langsung dimasukkan ke dalam basis data relasional yang dipilih.

III.9 Analisis Validasi Data

Mekanisme validasi data bertujuan untuk menghindari kesalahan masukan. Berdasarkan pada penelitian (Panko, 1998), terdapat dua jenis tipe kesalahan yang bisa terjadi pada penggunaan *spreadsheet* yakni kesalahan kualitatif dan kesalahan kuantitatif. Dari kedua

tipe tersebut, Tugas Akhir ini lebih menekankan kepada kesalahan kuantitatif. Kesalahan kuantitatif diambil karena merupakan jenis kesalahan yang dapat ditangani oleh kakas, sedangkan jenis kesalahan kualitatif lebih berkaitan pada aturan dan konvensi pembuatan *spreadsheet*.

Berdasarkan penelitian (Howe and Simkin, 2006), diketahui bahwa pengurangan kesalahan *formula* sangat dipengaruhi oleh faktor-faktor eksternal yang dapat diselesaikan jika berlatih dan terbiasa. Namun, jenis kesalahan lain tidak dipengaruhi oleh faktor-faktor ini, sehingga tingkat kesalahan yang terjadi cukuplah konsisten tidak tergantung pengalaman. Sehingga, jenis kesalahan yang akan ditangani oleh kakas ini akan lebih difokuskan pada kesalahan-kesalahan yang sulit dihilangkan bahkan jika pengguna telah berpengalaman.

Jenis kesalahan yang akan ditangani adalah kesalahan *clerical* dan non-material, *rule violations*, dan kesalahan *data-entry*. Dari kesalahan-kesalahan tersebut, maka terdapat tiga hal utama yang divalidasi pada kakas ini, yakni:

1. Validasi tipe data. Untuk pengembangan Tugas Akhir ini, hanya diimplementasi empat jenis validasi tipe data yakni *integer*, *double*, *string*, dan *boolean*. Contoh kasus validasi adalah ketika data masukan yang seharusnya berupa *integer* tidak dapat menerima masukan berupa *string*.

2. Validasi domain masukan. Pada pengembangan Tugas Akhir ini, hanya diimplementasi validasi domain yang berupa aturan *range*, lebih besar, lebih kecil, sama dengan, dan nilai diskrit. Aturan validasi juga hanya terbatas hanya satu aturan pada satu atribut pada *metadata table*.

3. Validasi relasi data. Data masukan dapat berupa nilai yang harus ada pada tabel lain, contohnya terdapat 2 tabel yakni nilai mahasiswa dan identitas mahasiswa. Tabel nilai mahasiswa memiliki kolom NIM yang nilainya harus ada pada tabel identitas mahasiswa. Pada *spreadsheet* biasanya hal ini diatasi dengan menggunakan perintah VLOOKUP.

Kesalahan *clerical* dan non-material diharapkan dapat ditangani menggunakan validasi tipe data dan domain masukan. Kesalahan *data-entry* dan *rule violations* diharapkan dapat ditangani menggunkan validasi tipe data, validasi domain, dan validasi relasi data.

Kakas akan melakukan validasi terhadap ketiga hal tersebut dan menolak data masukan sehingga pengguna dapat memperbaiki data. Validasi akan dilakukan pada saat data akan dimasukkan ke dalam basis data. Letak validasi ini dapat dilihat pada alur pengumpulan data pada Subbab III.12.

III.10 Analisis Penyimpanan Data

Terdapat tiga permasalahan pada penyimpanan data yang perlu dianalisis yakni; jenis penyimpanan data yang digunakan, alur pengguna melakukan penyimpanan data tersebut, dan perubahan pada data di basis data. Kedua masalah ini akan dijelaskan pada dua subbab berikut.

III.10.1 Jenis Penyimpanan Data

Terdapat 2 jenis data yang harus disimpan ke dalam basis data di dalam membangun kakas menggunakan aplikasi *spreadsheet* ini, yakni:

1. Penyimpanan *File Spreadsheet*

   File spreadsheet yang dimaksud adalah data struktural seperti *value* pada suatu sel, *properties* suatu sel seperti warna, *border*, dan *alignment*, serta hal-hal lain yang berhubungan dengan bagaimana suatu *file spreadsheet* ditampilkan pada aplikasi. Tipe penyimpanan yang dapat digunakan untuk menampung data ini adalah NoSQL karena tipe ini cocok untuk menangani data yang kurang terstruktur seperti *file spreadsheet* yang struktur penempatan datanya berbeda tiap pengguna. Cara penyimpanan yang cocok untuk menangani struktur ini adalah *document based* atau *key-value*. Sehingga proses penyimpanan *file spreadsheet* akan menggunakan mekanisme yang sudah disediakan oleh aplikasi EtherCalc dengan menggunakan redis.

2. Penyimpanan Data dan Label

   Tipe penyimpanan yang cocok digunakan adalah basis data relasional (SQL) karena data yang dibuat dalam bentuk tabel pada umumnya dapat dikonversikan menjadi *tuple* relasional. Data dan label yang telah ditentukan menggunakan *metadata table* dijadikan bentuk yang dapat diterima basis data relasional. Label akan dijadikan nama kolom pada basis data, dan data akan dijadikan dalam bentuk baris-baris.

III.10.2 Alur Penyimpanan Data

Dengan adanya dua jenis penyimpanan yang digunakan seperti telah dijelaskan pada Subbab III.10.1 terdapat dua kemungkinan alur penyimpanan data, yakni:

1. Penyimpanan ke basis data dilakukan setiap mendapatkan masukan pengguna
2. Penyimpanan ke basis data menunggu perintah dari pengguna

Pada Tabel III.4 dijabarkan perbandingan antara kedua kemungkinan alur penyimpanan data tersebut.

Tabel III.4: Perbandingan Alur Penyimpanan

Parameter	**Dilakukan Setiap Mendapat Masukan**	**Menunggu Perintah Pengguna**
Sinkronisasi	Data yang ada pada basis data yang menyimpan *file spreadsheet* yakni redis dan data pada basis data relasional tersinkronisasi setiap saat.	Data diantara kedua basis data tidak selalu sinkron setiap saat.
Kinerja	Kinerja yang akan menurun diakibatkan kakas pendeteksian label dan data yang harus dilakukan setiap saat menerima masukan.	Kinerja yang lebih baik dibandingkan alur sebelumnya karena penyimpanan dilakukan *on demand*
Akses penyimpanan	Sulit untuk mengembangkan aplikasi menggunakan *user access control* karena sulit menentukan kapan dan oleh siapa data boleh dimasukkan ke dalam basis data.	Mengembangkan aplikasi dengan *user access control* akan lebih mudah untuk menentukan siapa yang memiliki akses untuk melakukan penyimpanan.

Dari pertimbangan tersebut, alur yang dipilih adalah penyimpanan ke basis data yang menunggu perintah dari pengguna karena kelebihannya pada sisi performa dan *user access*

control. Di samping itu, tidak sinkronnya antara kedua basis data bukan merupakan masalah karena data yang ingin dikumpulkan merupakan data akhir dan data tiap perubahan yang dilakukan tidak begitu dipedulikan.

III.10.3 Perubahan pada Data di Basis Data

Kasus yang dapat terjadi pada basis data tempat penyimpanan data yang berasal dari *spreadsheet* adalah diubahnya data tersebut secara manual oleh entitas lain. Pada kakas yang dibuat, kakas hanya bekerja sebagai media untuk mengumpulkan data sehingga perubahan yang terjadi pada basis data, tidak akan ditampilkan kembali menjadi bentuk *spreadsheet* pada aplikasi. Di samping itu, perubahan data langsung pada basis data tanpa melalui media yang disediakan sangat tidak disarankan.

III.11 Analisis Interaksi antar *Spreadsheet*

Interaksi yang akan dibahas terdiri dari dua jenis yakni interaksi data dan interaksi penyimpanan. Pada interaksi data akan dijelaskan bagaimana data pada dua atau lebih *spreadsheet* dapat saling berinteraksi dalam bentuk fragmen. Sedangkan pada interaksi penyimpanan akan dijelaskan bagaimana aplikasi menyelesaikan permasalahan konflik penyimpanan yang terjadi akibat interaksi dua atau lebih *spreadsheet*.

III.11.1 Interaksi Data

Data yang dikumpulkan pada dua atau lebih *spreadsheet* dapat diklasifikasikan menjadi empat tipe berdasarkan fragmen data yang dikumpulkan yakni; tidak berhubungan, horizontal, vertikal, dan gabungan.

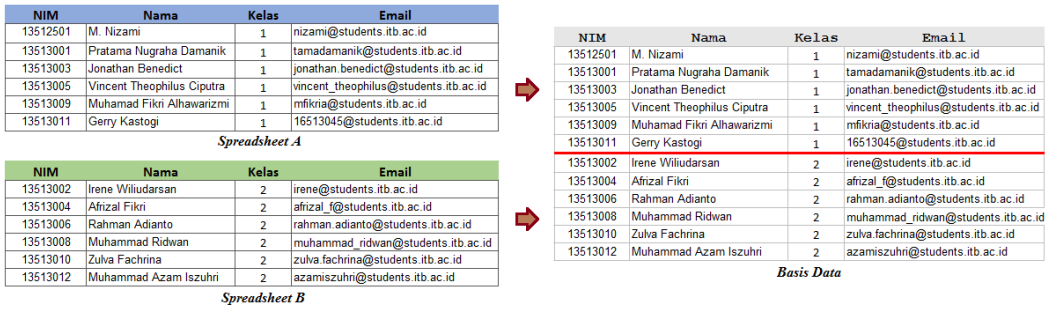
1. Fragmen Tidak Berhubungan

   Dua atau lebih *spreadsheet* dapat dikatakan tidak saling berpotongan fragmennya jika *spreadsheet* tersebut tidak menyimpan data pada tabel yang sama pada basis data. Pada kasus tersebut, data pada kedua atau lebih *spreadsheet* dapat dianggap tidak saling melengkapi dan dapat berdiri sendiri.

2. Fragmen Horizontal

   Fragmen horizontal terjadi ketika dua atau lebih *spreadsheet* menyimpan pada tabel basis data yang sama dimana data yang disimpan memiliki jumlah, nama, dan konteks kolom yang sama. Pada kasus tersebut, data pada dua atau lebih *spreadsheet*

akan digabungkan pada satu tabel yang sama secara horizontal. Contoh gambaran pengabungan data dapat dilihat pada Gambar III.10.



NIM	Nama	Kelas	Email
13512501	M. Nizami	1	nizami@students.itb.ac.id
13513001	Pratama Nugraha Damanik	1	tamadamanik@students.itb.ac.id
13513003	Jonathan Benedict	1	jonathan.benedict@students.itb.ac.id
13513005	Vincent Theophilus Ciputra	1	vincent_theophilus@students.itb.ac.id
13513009	Muhamad Fikri Alhawarizmi	1	mfikria@students.itb.ac.id
13513011	Gerry Kastogi	1	16513045@students.itb.ac.id

Spreadsheet A

NIM	Nama	Kelas	Email
13513002	Irene Wiliudarsan	2	irene@students.itb.ac.id
13513004	Afrizal Fikri	2	afrizal_f@students.itb.ac.id
13513006	Rahman Adianto	2	rahman.adianto@students.itb.ac.id
13513008	Muhammad Ridwan	2	muhammad_ridwan@students.itb.ac.id
13513010	Zulva Fachrina	2	zulva.fachrina@students.itb.ac.id
13513012	Muhammad Azam Iszuhri	2	azamiszuhri@students.itb.ac.id

Spreadsheet B

NIM	Nama	Kelas	Email
13512501	M. Nizami	1	nizami@students.itb.ac.id
13513001	Pratama Nugraha Damanik	1	tamadamanik@students.itb.ac.id
13513003	Jonathan Benedict	1	jonathan.benedict@students.itb.ac.id
13513005	Vincent Theophilus Ciputra	1	vincent_theophilus@students.itb.ac.id
13513009	Muhamad Fikri Alhawarizmi	1	mfikria@students.itb.ac.id
13513011	Gerry Kastogi	1	16513045@students.itb.ac.id
13513002	Irene Wiliudarsan	2	irene@students.itb.ac.id
13513004	Afrizal Fikri	2	afrizal_f@students.itb.ac.id
13513006	Rahman Adianto	2	rahman.adianto@students.itb.ac.id
13513008	Muhammad Ridwan	2	muhammad_ridwan@students.itb.ac.id
13513010	Zulva Fachrina	2	zulva.fachrina@students.itb.ac.id
13513012	Muhammad Azam Iszuhri	2	azamiszuhri@students.itb.ac.id

Basis Data

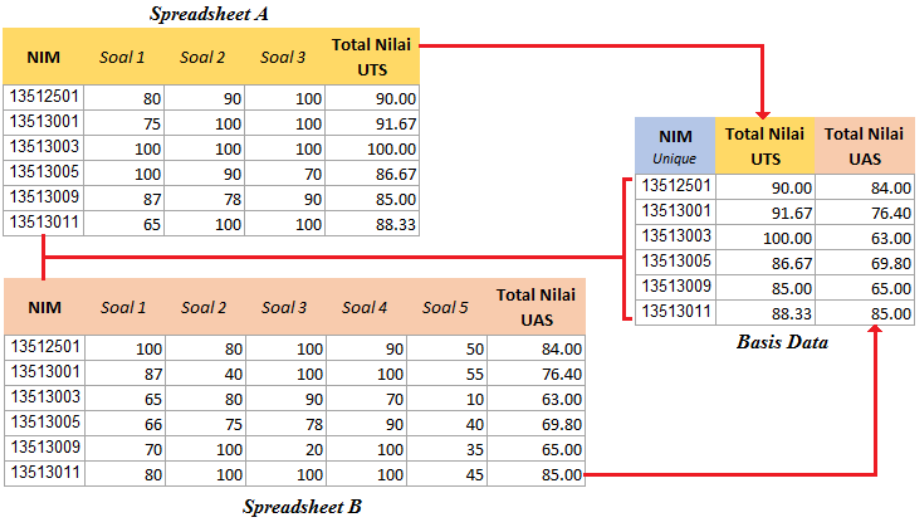
Gambar III.10: Contoh Penggabungan Horizontal

Pada gambar, data pada *spreadsheet* A dan *spreadsheet* B digabungkan pada satu tabel pada basis data secara horizontal (diilustrasikan dengan garis merah). Pada kasus tersebut, NIM tidak didefinisikan sebagai nilai yang unik, sehingga jika terdapat NIM yang sama pada lebih dari satu *spreadsheet* maka data akan tetap digabungkan secara horizontal pada data yang sudah ada bukan menulis kembali nilai pada NIM yang bersesuaian. Namun, jika nilai NIM didefinisikan sebagai unik, maka jika NIM terdapat di lebih dari satu *spreadsheet* akan ditimpa dan ditulis kembali nilainya pada NIM yang sesuai.

3. Fragmen Vertikal

   Fragmen vertikal terjadi ketika dua atau lebih *spreadsheet* menyimpan pada tabel basis data yang sama dimana data yang disimpan memiliki satu kolom unik dengan nama dan konteks yang sama dan kolom lainnya dapat berbeda dari segi jumlah , nama, dan konteks di masing-masing *spreadsheet*. Pada kasus ini, data pada dua atau lebih *spreadsheet* akan digabungkan pada satu tabel yang sama secara vertikal. Contoh gambaran pengabungan data dapat dilihat pada Gambar III.11.

   Penggabungan secara vertikal membutuhkan suatu kolom *key* dan unik yang dapat dijadikan patokan penggabungan data pada suatu baris data. Untuk masing-masing *spreadsheet* yang akan digabungkan datanya, harus memiliki kolom *key* dan unik yang sama dengan konteks data yang sama. Pada contoh pada gambar, kolom unik yang dipilih adalah kolom NIM. Penggabungan data dilakukan dengan mengambil data dari *spreadsheet* A yakni data nilai UTS, dan dari *spreadsheet* B yakni data nilai

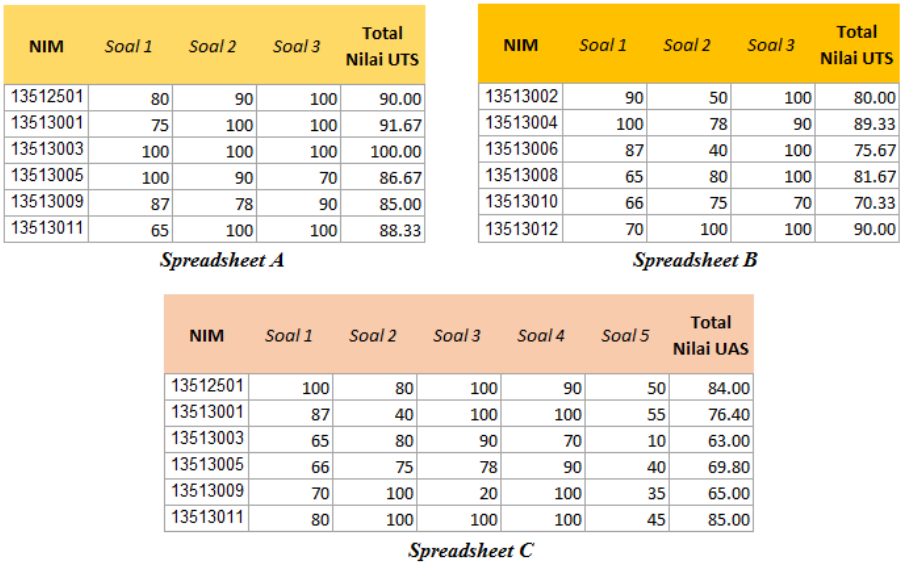


Gambar III.11: Contoh Penggabungan Vertikal

UAS, lalu dimasukkan ke dalam basis data pada tabel yang dipilih.

4. Fragmen Horizontal dan Vertikal

   Fragmen gabungan ini terjadi ketika data yang digabungkan ada yang memiliki perbedaan kolom, ada juga yang tidak, atau juga dapat merupakan gabungan semua kolom yang ada. Contoh kasus yang dapat terjadi dapat dilihat pada Gambar III.12.



Gambar III.12: Contoh Tabel Penggabungan

Terdapat tiga *spreadsheet* dimana data pada *spreadsheet* A dan *spreadsheet*

C memiliki jumlah kolom yang sama dan dapat digabung secara horizontal. Sedangkan, *spreadsheet* A dan *spreadsheet* B memiliki jumlah kolom yang berbeda namun kolom unik yang sama yakni NIM sehingga dapat digabungkan secara vertikal. Hasil penggabungan ketiga *spreadsheet* ini akan menjadi tabel yang dapat dilihat pada Gambar III.13.

NIM *Unique*	Total Nilai UTS	Total Nilai UAS
13512501	90.00	84.00
13513001	91.67	76.40
13513003	100.00	63.00
13513005	86.67	69.80
13513009	85.00	65.00
13513011	88.33	85.00
13513002	80.00	NULL
13513004	89.33	NULL
13513006	75.67	NULL
13513008	81.67	NULL
13513010	70.33	NULL
13513012	90.00	NULL

Gambar III.13: Contoh Penggabungan Horizontal dan Vertikal

Karena dari hasil penggabungan data tidak semua kolom dapat dipenuhi, maka terdapat kolom yang berisikan NULL sebagai penanda bahwa kolom tersebut kosong dan tidak mendapatkan data dari *spreadsheet.*

III.11.2 Interaksi Penyimpanan

Penanganan interaksi penyimpanan menjadi penting pada saat terdapat dua atau lebih *spreadsheet* memasukkan datanya pada suatu tabel yang sama. Kasus yang dapat terjadi adalah data yang telah sebelumnya dimasukkan oleh sebuah *spreadsheet* dapat ditimpa oleh data dari *spreadsheet* lainnya. Hal ini, jika memang tidak diharapkan, akan menyebabkan permasalahan kerusakan data. Pada kakas yang dibuat, diberikan penanganan berupa tombol pengecekan yang akan menjalankan fitur yang dapat melakukan pengecekan konflik data. Jika pengguna memang ingin melakukan penimpaan terhadap data yang sudah ada, maka peringatan dari fitur ini dapat diabaikan.

Urutan penyimpanan antar *spreadsheet* tidak akan mempengaruhi hasil dengan syarat data yang disimpan dari kedua *spreadsheet* merupakan data yang *disjoint* atau kelompok data yang terpisah satu dengan yang lainnya. Jika syarat tersebut tidak dipenuhi,

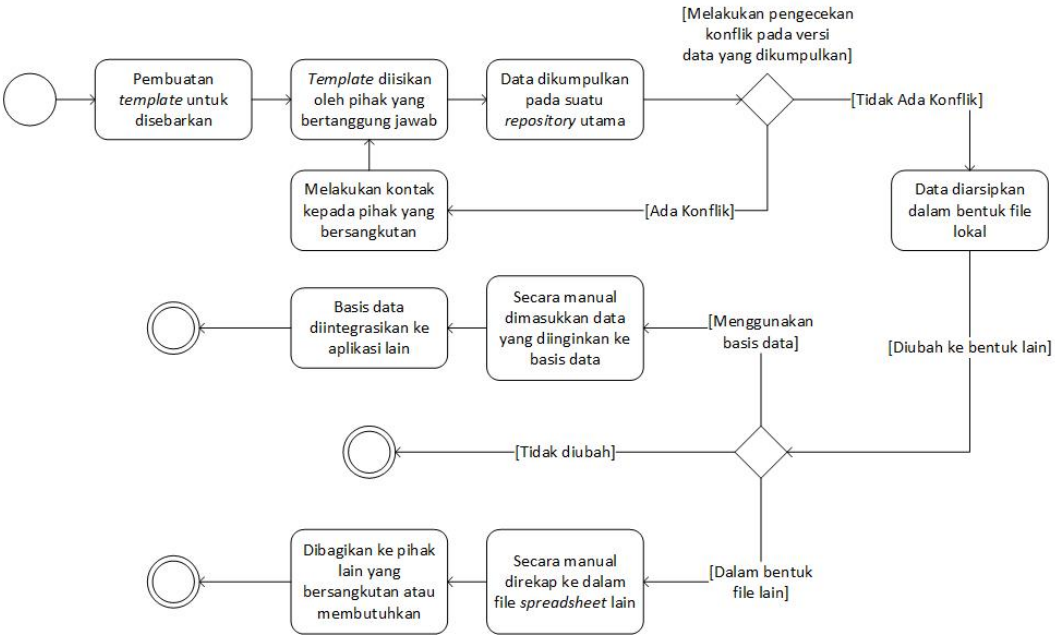
maka pengguna harus memasukkan data dengan urutan yang benar dan diperhitungkan agar data yang masuk sesuai dengan keinginan. Hal ini harus diperhitungkan dalam pengumpulan data yang dilakukan oleh pengguna.

III.12 Analisis Alur Kerja Manusia dan Kakas

Pada subbab ini, akan dijelaskan perbedaan alur kerja manusia dalam pengumpulan data menggunakan aplikasi yang telah ditambahkan dengan kakas yang dibuat dengan aplikasi biasa. Selain itu, akan dijelaskan juga perubahan alur kakas di dalam menangani masukan pengguna.

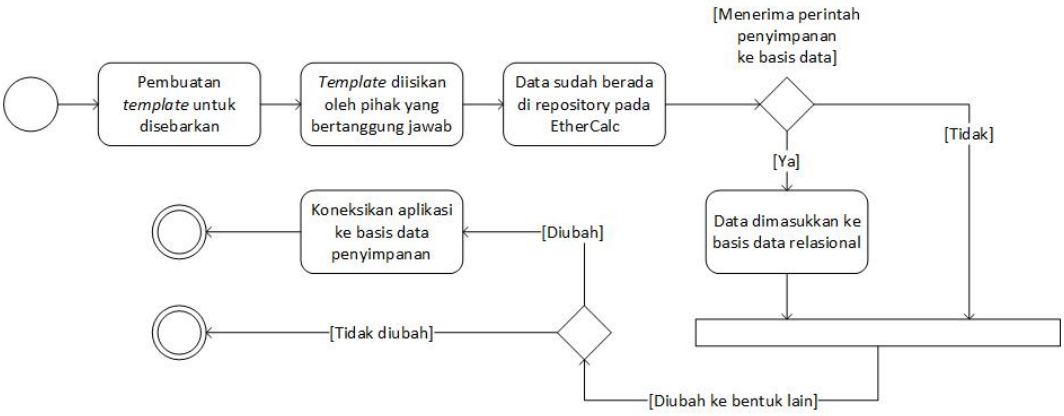
III.12.1 Alur Kerja Manusia

Terdapat perbedaan alur kerja yang terjadi jika pengguna menggunakan aplikasi yang dibuat pada Tugas Akhir. Alur kerja lama dapat dilihat pada Gambar III.14 dan alur kerja pada aplikasi *spreadsheet* yang dibagun dapat dilihat pada Gambar III.15.



Gambar III.14: Alur Pengumpulan Data Biasa

Seperti yang digambarkan pada alur tersebut, dengan menggunakan aplikasi dengan kakas yang dikembangkan akan mengurangi alur yang dibutuhkan untuk mengumpulkan data. Pengurangan alur terjadi pada bagian penanganan konflik pada versi-versi *spreadsheet* yang berbeda dan menjadi hanya dibutuhkan satu cara untuk mengubah data ke bentuk

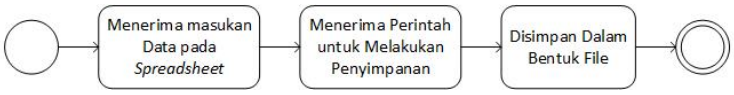


Gambar III.15: Alur Pengumpulan Data dengan Kakas yang Dibuat

lain, yakni menggunakan koneksi basis data.

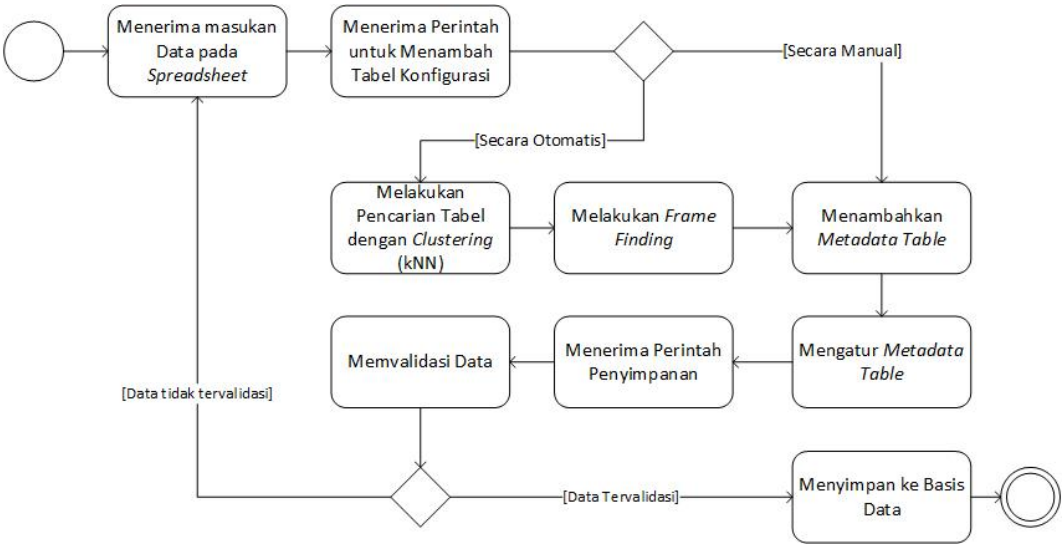
III.12.2 Alur Kerja Kakas

Kakas yang dibuat akan mengubah alur aplikasi *spreadsheet* yang umum dengan menambahkan mekanisme-mekanisme yang dibutuhkan. Alur aplikasi yang lama dapat dilihat pada Gambar III.16 dan aplikasi *spreadsheet* yang dibangun pada Tugas Akhir ini pada Gambar III.17.



Gambar III.16: Alur Aplikasi *Spreadsheet* Biasa

Dapat dilihat bahwa terdapat proses tambahan yang dilakukan pada aplikasi yang telah ditambahkan kakas yang dibuat pada Tugas Akhir ini. Proses tersebut terdiri dari pencarian bagian label dan data jika menggunakan pencarian otomatis, pengaturan *metadata table* untuk aturan masukkan ke basis data, serta proses validasi. Fokus kakas ini hanya sampai menyimpan data pada basis data, sehingga permasalahan *input* dan *output* berupa *file* tidak ditangani. Alur kolaborasi bukan merupakan fokus pengembangan kakas pada Tugas Akhir ini sehingga akan ditangani oleh aplikasi *spreadsheet* yang dipilih.



Gambar III.17: Alur Aplikasi *Spreadsheet* dengan Kakas yang Dibuat

BAB IV

RANCANGAN, IMPLEMENTASI, DAN PENGUJIAN

IV.1 Perancangan Perangkat Lunak

Subbab perancangan perangkat lunak menjelaskan deskripsi aplikasi, analisis kebutuhan fungsional dan non-fungsional, desain perangkat lunak, serta interaksinya.

IV.1.1 Deskripsi Umum Kakas

Kakas pengumpulan data yang dibuat merupakan pengembangan terhadap aplikasi *spreadsheet* kolaboratif yang sudah ada sebelumnya yakni EtherCalc. Kakas yang ditambahkan ini yang akan melakukan pengaturan koneksi ke basis data. Saat pengguna menginginkan penyimpanan data ke dalam basis data yang dituju, kakas ini akan melakukan pencarian bagian label dan data dan melakukan validasi masukan sebelum memasukkan data ke dalam basis data.

IV.1.2 Spesifikasi Kebutuhan

Pada subbab ini akan dipaparkan *use case* kakas yang akan dibuat serta kebutuhan fungsional dan non-fungsional dari kakas. Kasus penggunaan oleh pengguna diberi ID dengan format UC-XX dengan UC menyatakan *use case* dan XX menyatakan nomor. Pengguna adalah pihak yang menggunakan aplikasi *spreadsheet* yang sudah ditambahkan kakas pengumpulan data. Kasus penggunaan oleh pengguna dijelaskan pada Tabel IV.1.

Tabel IV.1: Kasus Penggunaan oleh Pengguna

ID	Deskripsi
UC-01	Pengguna dapat menentukan basis data tujuan dengan konfigurasi basis data yang diinginkan.
UC-02	Pengguna dapat menyimpan data yang dikumpulkan ke dalam basis data pada saat dibutuhkan.
UC-03	Pengguna dapat mendefinisikan *metadata table* yang ingin disimpan secara manual maupun otomatis oleh kakas.

ID	Deskripsi
UC-04	Pengguna dapat mengubah metadata pada *metadata table* yang ingin disimpan.
UC-05	Pengguna dapat mendefinisikan aturan validasi data.
UC-06	Pengguna dapat mengunggah berkas *spreadsheet* ke dalam aplikasi.

Untuk *use case* dengan kode UC-06, sudah ditangani oleh aplikasi *EtherCalc* sehingga tidak akan ditangani oleh kakas yang akan dikembangkan. Berdasarkan kasus penggunaan di atas, dirancang kebutuhan fungsional perangkat lunak yang diberi ID dengan format FR-XX dengan FR merupakan singkatan dari *functional requirement* dan XX menyatakan nomor kebutuhan. Kebutuhan fungsional dijelaskan pada Tabel IV.2.

Tabel IV.2: Kebutuhan Fungsional Kakas

ID	Deskripsi	ID Use Case Terkait
FR-01	Kakas dapat melakukan koneksi kepada basis data yang ditentukan oleh pengguna melalui data masukan berupa *host*, *port*, *username*, *password*, dan *database* dari basis data yang dituju.	UC-01
FR-02	Kakas dapat melakukan perintah basis data kepada basis data yang dituju.	UC-01, UC-02
FR-03	Kakas menyediakan fitur untuk dapat melakukan penyimpanan data saat pengguna ingin melakukan penyimpanan.	UC-02
FR-04	Kakas menyediakan fitur menambah dan mengurangi *metadata table* baik secara pendeteksian otomatis maupun manual.	UC-03
FR-05	Kakas dapat menampilkan hasil identifikasi label dan data dalam bentuk *metadata table.*	UC-03, UC-04
FR-06	Kakas menyediakan fitur bagi pengguna agar dapat mengubah hasil identifikasi label dan data.	UC-04

ID	Deskripsi	ID Use Case Terkait
FR-07	Kakas menyediakan fitur bagi pengguna agar dapat menambahkan dan mengubah batasan masukan pada suatu data.	UC-05
FR-08	Kakas dapat melakukan validasi data masukan sesuai dengan batasan yang diberikan oleh pengguna.	UC-05
FR-09	Kakas dapat memberitahukan kesalahan validasi yang terjadi.	UC-05

Selain kebutuhan fungsional, dijabarkan juga kebutuhan non-fungsional yang memiliki ID dengan format NF-XX dengan NF merupakan singkatan dari *non-functional requrirement* dan XX menyatakan nomor. Kebutuhan non-fungsional disajikan pada Tabel IV.3.

Tabel IV.3: Kebutuhan Non-fungsional Kakas

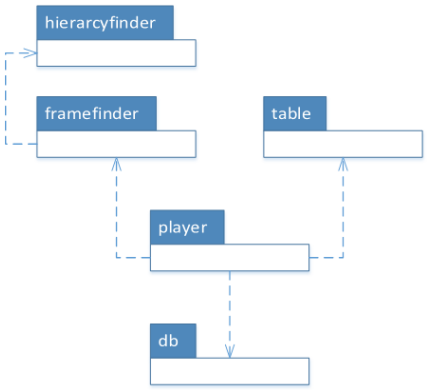
ID	Deskripsi	ID Use Case Terkait
NF-01	Data masukan pengguna disimpan secara persisten.	UC-01, UC-02
NF-02	Kakas dapat berjalan di atas aplikasi *spreadsheet* EtherCalc.	-

IV.1.3 Kebutuhan Modul

Pembangunan fitur ini di atas aplikasi EtherCalc terdiri dari lima buah modul, yaitu:

1. Modul `player`, bertugas sebagai jembatan antara *front-end* dan *back-end* dari fitur.
2. Modul `mysql`, bertugas untuk antarmuka baca tulis basis data.
3. Modul `framefinder`, bertugas untuk mendeteksi secara otomatis bagian label dan data pada tabel.
4. Modul `hierarchyfinder`, bertugas untuk mendeteksi secara otomatis tabel-tabel yang ada dalam suatu *sheet*.
5. Modul `table`, bertugas untuk mengelola *metadata table*, melakukan validasi data, dan membuat representasi relasional.

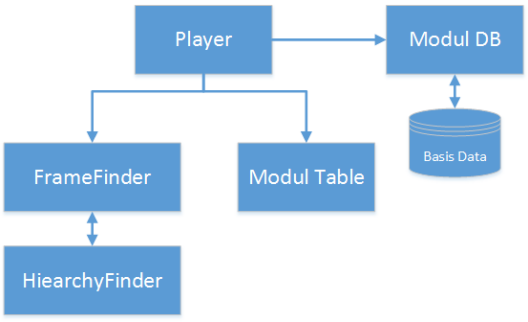
Ketergantungan antar modul dapat dilihat pada Gambar IV.1



Gambar IV.1: Ketergantungan Antar Modul

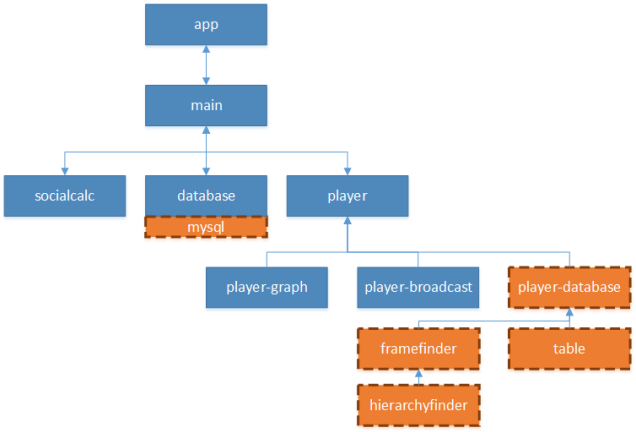
IV.1.4 Kolaborasi Antar Modul

Proses fitur ini akan dilakukan melalui modul `player` yang dapat menerima perintah pengguna melalui *front-end*. Selanjutnya modul `framefinder` akan melakukan pendeteksian label dan data secara otomatis pada masing-masing tabel yang terdapat pada *sheet*. Tabel-tabel tersebut didapatkan melalui modul `hierarchyfinder`. Selanjutnya, pada saat menerima perintah penyimpanan, modul `table` akan dipanggil oleh `player`. Jika data masukan sudah benar, maka modul `mysql` akan melakukan pennyimpanan ke dalam basis data. Kolaborasi antar modul disajikan pada Gambar IV.2.



Gambar IV.2: Kolaborasi Antar Modul

Pada aplikasi Ethercalc, modul yang ditambahkan akan berada di bawah modul `player` yang sudah ada. Ilustrasi penempatan modul yang dibuat pada aplikasi EtherCalc yang sudah ada dapat dilihat pada Gambar IV.3.



Gambar IV.3: Interaksi dengan Modul yang Sudah Ada

IV.1.5 Arsitektur

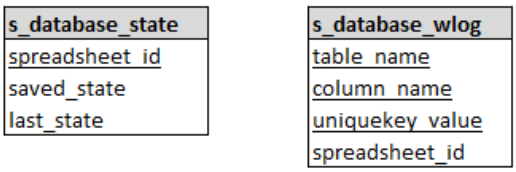
Aplikasi dibuat dengan menggunakan bantuan Docker sehingga diharapkan dapat dengan mudah dipasang pada berbagai *platform*. Aplikasi terdiri dari tiga *docker container* yakni untuk aplikasi utama, basis data redis, dan basis data MySQL. *File* `docker-compose.yml` yang dibuat untuk mendefinisikan *docker container* yang digunakan dapat dilihat pada Kode IV.1

```
ethercalc:
  build: .
  ports:
    - "80:8000"
  links:
    - redis:redis
    - mysql:mysql
  restart: always
redis:
  image: redis:latest
  volumes:
    - /var/lib/redis:/data
  command: redis-server --appendonly yes
  restart: always
mysql:
  image: mysql:5.7
  volumes:
    - dbdata:/var/lib/mysql
  restart: always
  environment:
    MYSQLROOTPASSWORD: root
    MYSQLDATABASE: TA
    MYSQLUSER: user
    MYSQLPASSWORD: user
```

Kode IV.1: Kode pada docker-compose.yml

Untuk menyimpan *state* konfigurasi tabel dan *mapping metadata table* dengan data pada basis data, digunakan dua tabel basis data yang sudah didefinisikan sebelumnya. Kedua

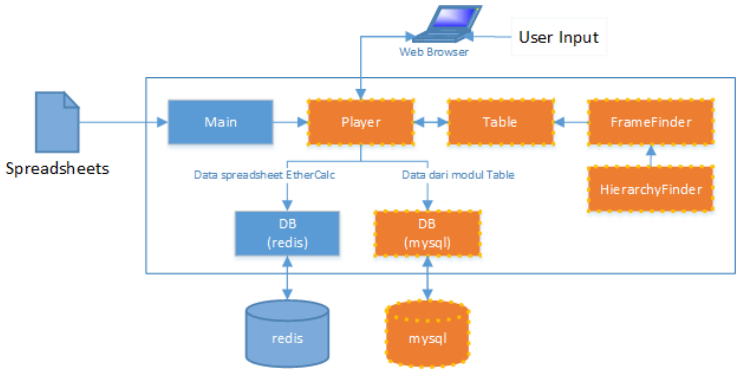
tabel basis data ini dinamakan s_database_state dan s_database_wlog, kedua tabel ini tidak memiliki keterhubungan. Atribut yang terdapat pada kedua tabel ini dapat dilihat pada diagram struktur basis data pada Gambar IV.4.



Gambar IV.4: Diagram Struktur Tabel Penyimpanan Konfigurasi

metadata table yang ditampilkan kepada pengguna akan berasal dari tabel s_database_state dan juga disimpan pada tabel tersebut. Kegunaan s_database_wlog adalah mencatat pemetaan data kepada *metadata table* sehingga saat *metadata table* dihapus, data yang berasal dari *metadata table* tersebut dapat juga dihapus.

Kakas dibuat di atas aplikasi EtherCalc, sehingga akan diimplementasi tambahan modul maupun menggunakan modul yang sudah ada pada EtherCalc. Interaksi antar modul yang dibuat dengan modul yang sudah ada serta dengan data masukan dapat dilihat pada Gambar IV.5.



Gambar IV.5: Diagram Arsitektur Aplikasi

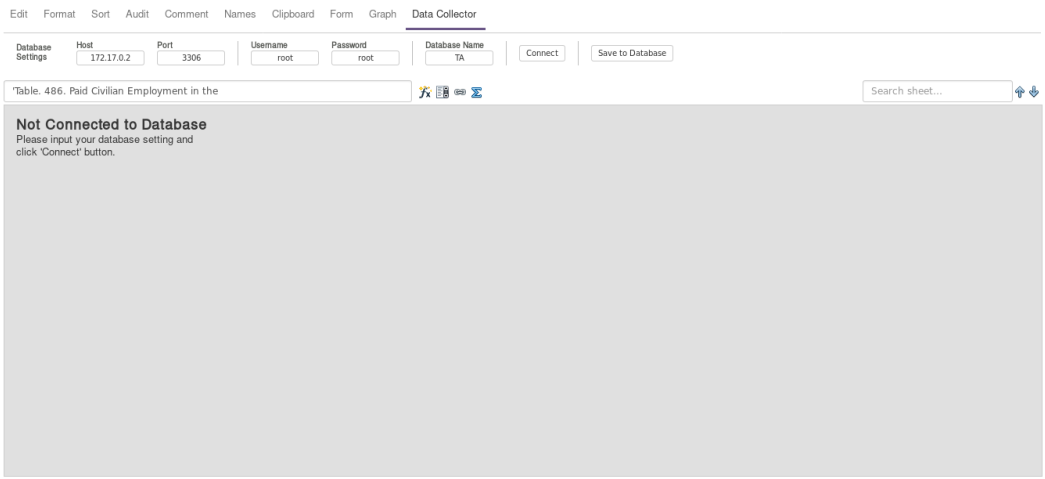
Pada diagram arsitektur terlihat bahwa masih terdapat dua basis data yakni redis dan MySQL. Hal ini dilakukan agar data mengenai format dan cara ditampilkan pada aplikasi EtherCalc tetap disimpan pada basis data redis sedangkan data yang ingin dikumpulkan dan digabungkan oleh pengguna dimasukkan pada basis data relasional MySQL.

IV.2 Implementasi

Implementasi dilakukan dengan membangun modul yang telah dijabarkan pada Subbab IV.1.3 dengan menggunakan bahasa Javascript, menyesuaikan dengan modul lain yang telah ada pada aplikasi EtherCalc. Pada bagian ini akan dijelaskan antarmuka dan modul-modul yang diimplementasikan.

IV.2.1 Antarmuka

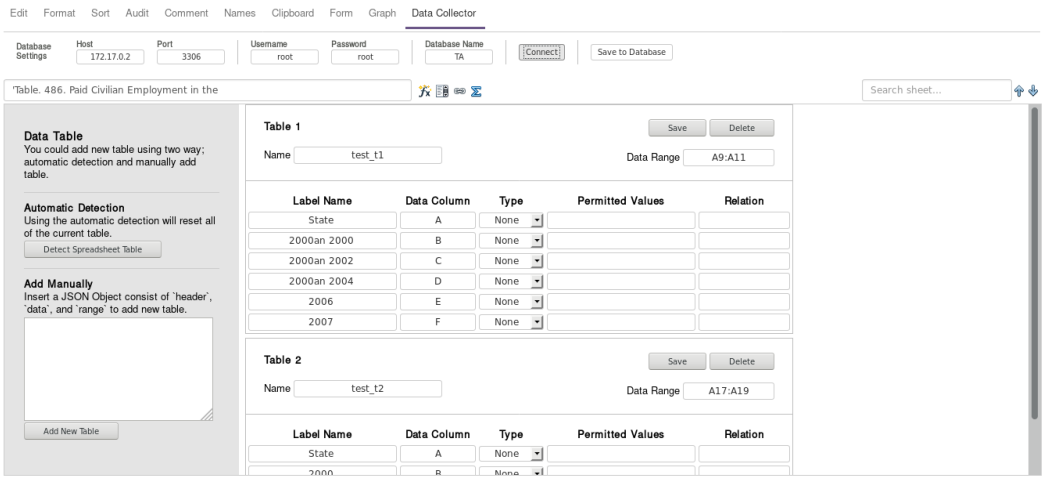
Antarmuka aplikasi pada Tugas Akhir ini mengikuti antarmuka yang telah ada sebelumnya pada aplikasi EtherCalc. Fitur baru yang ditambahkan akan menempati menu baru pada bagian atas antarmuka dengan nama 'Data Collector'. Antarmuka awal dapat dilihat pada Gambar IV.6.



Gambar IV.6: Antarmuka Awal

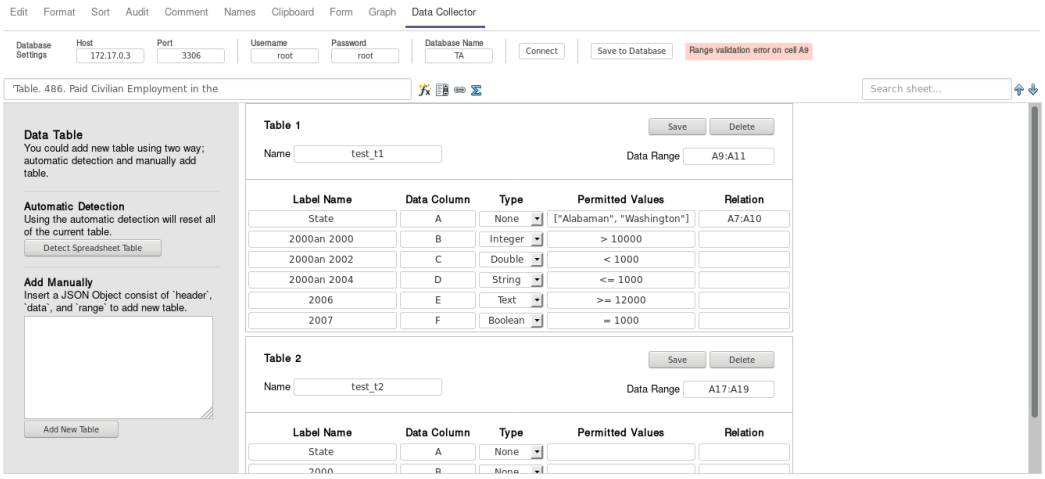
Terdapat dua bagian utama pada antarmuka ini, yang pertama adalah area pengaturan basis data dimana pengguna bisa dapat mengatur koneksi ke basis data yang diinginkan. Yang kedua adalah area tempat *metadata table* dapat ditambahkan dan dikurangi berdasarkan masukan pengguna. Pada bagian kiri area *metadata table*, terdapat dua pilihan metode untuk membuat *metadata table* yakni secara manual maupun otomatis. Kedua area ini dapat dilihat pada Gambar IV.7.

Pada bagian *metadata table*, pengguna dapat menentukan nama label, tipe data, nilai-nilai yang diizinkan, dan relasinya terhadap kumpulan sel lain untuk suatu kolom data. Pengguna dapat juga mengubah nama tabel tujuan ke basis data serta menghapus



Gambar IV.7: Antarmuka *metadata table*

metadata table. Jika terjadi kesalahan pada saat validasi, pengguna akan diberikan pesan kesalahan dan diharapkan untuk memperbaiki data pada sel yang dianggap memiliki kesalahan. Interaksi ini dapat dilihat pada Gambar IV.8.



Gambar IV.8: Antarmuka Perubahan *metadata table*

IV.2.2 Modul Main

Modul `Main` merupakan modul yang sudah ada pada aplikasi EtherCalc. Untuk mengimplementasi kakas, modul ini dikembangkan lagi dengan menambahkan beberapa *endpoint* API yang dibutuhkan. Seluruh API ini akan dipanggil melalui modul `Player`. API yang ditambahkan pada Tugas Akhir ini adalah:

1. `POST /_database/connect`

   Digunakan untuk melakukan pengecekan koneksi ke basis data. Parameter yang dibutuhkan adalah:

   - host: *host* basis data.
   - port: *port* basis data.
   - user: *user* basis dat yang digunakan.
   - password: *password* untuk *user* yang digunakan.
   - database: nama *database* yang digunakan.

2. `POST /_database/state`

   Digunakan untuk melakukan penyimpanan *state metadata table*. Parameter yang dibutuhkan adalah:

   - tables: data *metadata table* dalam bentuk JSON.
   - setting: pengaturan koneksi ke basis data yang dipilih berupa JSON Object.

3. `POST /_database/state/[spreadsheet_id]`

   Digunakan untuk mendapatkan data *state metadata table* pada suatu *spreadsheet*. Parameter yang dibutuhkan adalah::

   - setting: pengaturan koneksi ke basis data yang dipilih berupa JSON Object.

4. `POST /_database/create`

   Digunakan untuk melakukan penyimpanan data yang telah dikumpulkan sesuai dengan aturan pada *metadata table*. Parameter yang dibutuhkan adalah:

   - table: data dari *metadata table* yang telah diubah menjadi bentuk relasional dalam bentuk JSON.
   - setting: pengaturan koneksi ke basis data yang dipilih berupa JSON Object.

5. `GET /_framefinder/[spreadsheet_id]/[start_col]/[end_col]`
   `/[start_row]/[end_row]`

   Digunakan untuk mendapatkan hasil pencarian label dan data diantara kolom dan baris tertentu.

6. `GET /_hierarchical/[spreadsheet_id]`

   Digunakan untuk mendapatkan perkiraan kelompok sel yang merupakan suatu kesatuan tabel.

IV.2.3 Modul Player

Modul `player` merupakan modul yang menjembatani masukan pengguna dari yang berasal dari *front-end* sehingga dapat diterima oleh modul yang berada di *back-end*. Modul ini hanya terdiri dari satu kelas utama yakni kelas `player` yang berisi fungsi-fungsi yang dapat dipanggil oleh *front-end* yang dapat dilihat pada Lampiran A.

IV.2.4 Modul DB

Modul basis data digunakan sebagai antarmuka modul lain untuk melakukan operasi I/O basis data. Pada Tugas Akhir ini, basis data yang digunakan adalah MySQL. Modul ini hanya terdiri dari satu kelas utama yakni kelas `mysql`. Kelas ini memiliki tugas sebagai penghubung kakas ke basis data MySQL yang dipilih. Fungsi-fungsi yang terdapat pada kelas ini dapat dilihat pada Lampiran A.

Tabel akan dibuat pada basis data yang ditentukan, setiap tabel merepresentasikan suatu tabel pada *spreadsheet* yang ditentukan oleh pengguna. *Header* yang didefinisikan oleh pengguna pada *spreadsheet* akan dijadikan *column* pada tabel basis data, tipe yang dibentuk mengikuti masukan pengguna. Tiap baris data yang ada dibawah *header* pada *spreadsheet* akan ditranslasikan menjadi bentuk relasional agar dapat dimasukan ke dalam tabel.

IV.2.5 Modul Hierarchyfinder

Modul `hierarchyfinder` menggunakan algoritma kNN jenis *hierarchical clustering* untuk dapat mengetahui mana yang merupakan suatu kesatuan tabel pada suatu *sheet*. Modul ini dapat menentukan tabel-tabel yang terdapat pada suatu *sheet* yang selanjutnya akan dilakukan identifikasi label oleh modul `framefinder`. Algoritma *hierarchical clustering* yang digunakan menganggap setiap sel pada *spreadsheet* merupakan suatu node. Sel-sel yang bersebelahan dengan sel tersebut akan dianggap tetangga sehingga memiliki jarak sama dengan 0. Sel yang digabungkan dengan sel lain akan dihitung sebagai satu *node*. Aturan perhitungan jarak antar *node* dapat dilihat pada kode di Kode

IV.2.

```
# Fungsi cellDistance(v1, v2)
# Parameter pada fungsi adalah v1 (node 1) dan v2 (node 2)
t = new Table null, null
colD = t.GetCellCol(v1[0]) - t.GetCellCol(v2[0])
rowD = t.GetCellRow(v1[0]) - t.GetCellRow(v2[0])

# Jika sel saling bertetangga tetapi bukan secara diagonal
# Jika bertetangga, jarak kedua sel adalah 0
if colD == 0
  if rowD == 1 or rowD == -1
    return 0
if rowD == 0
  if colD == 1 or colD == -1
    return 0

# Jika tidak, cek apakah sel bertetangga secara diagonal
# Jika bertetangga, jarak kedua sel adalah 0
leftTop = [[v1[1], v1[2]], [v2[1], v2[2]]]
leftBot = [[v1[1], (v1[2] + v1[4])], [v2[1], (v2[2] + v2[4])]]
righTop = [[(v1[1] + v1[3]), v1[2]], [(v2[1] + v2[3]), v2[2]]]
righBot = [[(v1[1] + v1[3]), (v1[2] + v1[4])], [(v2[1] + v2[3]), (v2[2] +
    v2[4])]]

if (leftTop[0][0] == righTop[1][0] and leftTop[0][1] == righTop[1][1])
  return 0
if (leftBot[0][0] == righBot[1][0] and leftBot[0][1] == righBot[1][1])
  return 0

if (leftTop[1][0] == righTop[0][0] and leftTop[1][1] == righTop[0][1])
  return 0
if (leftBot[1][0] == righBot[0][0] and leftBot[1][1] == righBot[0][1])
  return 0

if (leftTop[0][0] == leftBot[1][0] and leftTop[0][1] == leftBot[1][1])
  return 0
if (righTop[0][0] == righBot[1][0] and righTop[0][1] == righBot[1][1])
  return 0

if (leftTop[1][0] == leftBot[0][0] and leftTop[1][1] == leftBot[0][1])
  return 0
if (righTop[1][0] == righBot[0][0] and righTop[1][1] == righBot[0][1])
  return 0

# Jika tidak juga, hitung jarak menggunakan teknik Euclidian
dist = Math.sqrt(Math.pow((v2[1] + (v2[3]/2)) - (v1[1] + (v1[3]/2)), 2) +
    Math.pow((v2[2] + (v2[4]/2)) - (v1[2] + (v1[4]/2)), 2));
return dist
```

Kode IV.2: Perhitungan Jarak *Node*

Hasil dari pengelompokan ini berupa kelompok-kelompok tabel pada *spreadsheet*. Setiap tabel yang teridentifikasi selanjutnya akan dicari bagian *header* dan *label* menggunakan modul `framefinder`.

IV.2.6 Modul Framefinder

Modul `framefinder` melakukan pengidentifikasian terhadap tabel yang ada sehingga dapat diketahui baris yang merupakan *header* dan *data*. Implementasi modul ini dilakukan dengan mengikuti implementasi yang dilakukan pada penelitian yang dilakukan oleh Chen (Chen and Cafarella, 2013). Modul ini terdiri dari 5 kelas yang dapat dilihat pada Lampiran A.

1. Kelas LoadSheet

   Kelas ini melakukan pengambilan data pada *spreadsheet* dengan cara membaca file *spreadsheet* dan membentuk representasi kelas yang dibutuhkan. Kelas ini memiliki satu atribut utama yakni cmysheet yang merupakan kelas MySheet. Fungsi-fungsi yang terdapat pada kelas ini dapat dilihat pada Lampiran A.

2. Kelas MySheet

   Kelas MySheet merupakan kelas yang merepresentasikan *sheet* pada *spreadsheet* yang dipilih. Kelas ini memiliki 4 atribut yang dapat dilihat pada Lampiran A. Pada kelas ini terdapat 3 fungsi yang dapat dilihat juga pada Lampiran A.

3. Kelas MyCell

   Kelas MyCell merupakan kelas yang merepresentasikan sel pada suatu *sheet*. Kelas ini memiliki 17 atribut yang dapat dilihat pada Lampiran A.

IV.2.6.1 Kelas FeatureSheetRow

Kelas ini bertugas untuk melakukan ekstraksi fitur pada tiap baris sel yang telah dikumpulkan dari *spreadsheet*. Fitur-fitur yang diambil untuk setiap barisnya dapat dilihat pada Lampiran A.

Fitur-fitur di atas mengikuti fitur yang didefinisikan pada penelitian yang dilakukan oleh Chen (Chen and Cafarella, 2013). Fitur-fitur ini akan digunakan dalam perhitungan algoritma CRF.

IV.2.6.2 Kelas PredictSheetRows

Kelas PredictSheetRows memiliki tugas untuk melakukan konversi fitur-fitur yang telah didapatkan pada kelas FeatureSheetRow menjadi bentuk file teks yang dapat dibaca oleh

program CRFPP yang akan menjalankan algoritma CRF pada file tersebut. Contoh file yang dihasilkan oleh kelas ini dapat dilihat pada Kode IV.3.

```
DeadlineTA.xlsSheet11 1 0 1 0 0 1 0 1 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 Header
DeadlineTA.xlsSheet12 0 1 1 0 0 1 0 0 0 0 0 0 1 0 0 1 0 1 1 0 0 0 0 0 Data
DeadlineTA.xlsSheet13 0 1 1 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 Data
DeadlineTA.xlsSheet14 0 1 1 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 Data
DeadlineTA.xlsSheet15 0 1 1 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0 0 Data
```

Kode IV.3: File Berisikan Fitur Tiap Baris

File tersebut ditulis dalam format `namafile_namasheet_bariske [fitur-fitur pada baris] label`. File ini yang akan diolah oleh algoritma dan digunakan untuk memprediksi label dari setiap baris tersebut. Algoritma yang digunakan adalah Conditional Random Field dan menggunakan aplikasi CRFPP yang berjalan eksternal diluar kelas ini untuk membaca file, membaca model, serta memprediski label.

IV.2.7 Modul Table

Modul `table` memiliki tugas untuk mengelola tampilan dan isi *metadata table*, melakukan validasi data masukan, dan membuat representasi relasional yang dapat diterima oleh basis data. Modul `table` hanya memiliki satu kelas yakni kelas `table`. Kelas ini memiliki 8 atribut yang dapat dilihat pada Lampiran A. Pada kelas ini terdapat 6 fungsi yang dapat dilihat pada Lampiran A.

Modul ini berinteraksi langsung dengan modul `player` pada saat akan melakukan penyimpanan *state metadata table* dan menampilkan *metadata table* pada *front-end*. Selain itu, modul ini juga berhubungan dengan modul `mysql` pada saat melakukan penyimpanan.

IV.3 Pengujian

Pengujian dilakukan hanya kepada fitur yang di buat pada Tugas Akhir ini dan tidak kepada aplikasi EtherCalc secara keseluruhan. Pada subbab ini akan dibahas kasus-kasus yang diuji dan hasil dari pengujian tersebut.

IV.3.1 Tujuan Pengujian

Pada fitur yang dibangun pada Tugas Akhir ini, pengujian yang dilakukan mempunyai tujuan yaitu fitur yang diimplementasi dapat berjalan dengan baik. Pengujian yang dilakukan akan dikaitkan pada kebutuhan-kebutuhan kakas yang telah didefinisikan

sebelumnya. Dari pengujian akan dilihat kesesuaian hasil sesungguhnya dengan hasil yang diinginkan.

IV.3.2 Lingkungan Pengujian

Pengujian dilakukan pada komputer dengan spesifikasi pada Tabel IV.4.

Tabel IV.4: Spesifikasi Lingkungan Pengujian

Komponen	Deskripsi
Prosesor	AMD A8-4500M APU with Radeon(tm) HD Graphics 1.90GHz
Memori Fisik	12 GB
Storage	50 GB
Sistem Operasi	Arch Linux 64-bit
Docker version	1.13.1
Web browser	Mozilla Firefox 51.0.1

IV.3.3 Eksekusi Pengujian

Kasus pengujian disesuaikan dengan kebutuhan fungsional dan non-fungsional yang sudah dijelaskan pada Subbab IV.1. Setiap kasus uji diberi kode TC-XX dengan XX adalah nomor kasus uji. Daftar kasus uji dapat dilihat pada Tabel IV.5.

Tabel IV.5: Kasus Uji Fitur Kakas

ID	Tujuan	ID kebutuhan terkait
TC-01	Terkoneksi ke basis data MySQL yang dipilih pengguna dan dapat menggunakan basis data tersebut	FR-01, FR-02
TC-02	Mendeteksi label dan data secara otomatis pada *spreadsheet*	FR-04, FR-05
TC-03	Menambahkan *metadata table* secara manual sesuai dengan masukan pengguna	FR-04
TC-04	Pengguna dapat mengubah atribut yang ada pada *metadata table*	FR-06
TC-05	Data dapat dimasukkan ke basis data sesuai dengan *metadata table*	FR-03, FR-09, NF-01
TC-06	Data yang dimasukkan ke basis data berhasil divalidasi sesuai aturan pengguna	FR-07, FR-08, FR-09, NF-01

IV.3.4 Hasil Pengujian

Hasil pengujian dikaitkan dengan ID skenario pengujian yang berkaitan langsung dengan kasus uji yang telah didefinisikan pada Bab IV.3.3. Tiap hasil pengujian mempunyai ID skenario terkait, ekspektasi, hasil uji yang dilakukan, dan diterima atau tidaknya hasil pengujian. Hasil pengujian merupakan hasil pelaksanaan skenario yang sudah dibuat dan dapat dilihat pada Lampiran B.

BAB V

KESIMPULAN DAN SARAN

Bab ini berisi hal-hal yang dapat disimpulkan dari pelaksanaan Tugas Akhir ini. Bab ini juga mencakup saran untuk pengembangan Tugas Akhir ini di masa mendatang.

V.1 Kesimpulan

Berdasarkan hasil pengembangan kakas pengumpulan data menggunakan *spreadsheet* yang telah dilakukan. Berikut adalah kesimpulan yang diperoleh.

1. Telah berhasil dilakukan penambahan fitur pada aplikasi EtherCalc yang dapat melakukan pengumpulan data ke dalam bentuk basis data.
2. Data yang akan dimasukkan ke basis data penyimpanan berhasil divalidasi menggunakan fitur yang dibuat dengan tiga tipe validasi yakni tipe data, domain data, dan relasi data.
3. Identifikasi tabel pada suatu *sheet* dapat dilakukan dengan menggunakan algoritma kNN dengan mencari kedekatan antar sel. Identifikasi label suatu baris pada tabel dapat dilakukan dengan teknik *frame finder* dengan membagi label menjadi empat jenis yakni *title*, *data*, *header*, dan *footer*. Jika teknik *frame finder* tidak berhasil menemukan label dan data, maka pengguna dapat memasukkan *metadata table* secara manual dan mengubahnya sesuai dengan keinginan pengguna.
4. Penggabungan data antar *spreadsheet* dapat dilakukan dengan fitur yang dibuat dan dapat digabungkan secara horizontal, vertikal, maupun gabungan keduanya. Data pada *spreadsheet* berhasil dimasukkan ke dalam basis data yang ditentukan sesuai dengan *metadata table* yang telah dibuat pengguna maupun hasil pencarian otomatis dari algoritma *frame finder*.
5. Alur kerja pengumpulan data berubah sehingga dapat diusulkan alur kerja baru dimana *versioning* dilakukan oleh aplikasi EtherCalc karena seluruh data berada pada satu tempat menggunakan mekanisme penyimpanan oleh EtherCalc. Pada saat

pengumpulan data dari berbagai *spreadsheet* dapat dilakukan menggunakan aplikasi yang sama yakni EtherCalc, sehingga pada alur kerja yang diusulkan, pengumpulan data tidak memerlukan bantuan aplikasi lain ataupun manual. Hasil akhir dari pengumpulan data merupakan data pada basis data sehingga data mudah diolah, ditampilkan, maupun diubah menggunakan banyak aplikasi yang tersedia.

V.2 Saran

Saran yang dapat diberikan untuk pengembangan di masa mendatang adalah sebagai berikut:

1. Pada pembangunan selanjutnya dapat ditambahkan penanganan kasus penggunaan *spreadsheet* selain *data frame* dan relasi.
2. Penambahan data pembelajaran untuk identifikasi label baris dapat dilakukan sehingga akan memperbaiki hasil identifikasi otomatis. Pada pengembangan selanjutnya dapat ditambahkan *feedback* dari pengguna sebagai data pembelajaran.
3. Menambahkan fungsionalitas yakni memperbolehkan kolom *key* yang tidak hanya satu pada *metadata table*.
4. Pengembangan fitur validasi, contohnya adalah menambahkan jenis validasi contohnya validasi masukan berbentuk formula. Di samping itu, dapat ditambahkan jenis validasi pada validasi tipe seperti tipe tanggal. Dapat juga penambahan fitur pada validasi domain seperti atribut yang dapat menerima tidak hanya satu aturan validasi domain.
5. Penanganan jenis tabel dengan *header* yang berada di kiri dan kanan data mungkin dapat dikembangkan menggunakan teknik *transpose*.

Daftar Pustaka

Bansal, S. K. (2014). Towards a Semantic Extract-Transform-Load (ETL) framework for big data integration. *Proceedings - 2014 IEEE International Congress on Big Data, BigData Congress 2014*, pages 522–529.

Bishop, B. and McDaid, K. (2008). An empirical study of end-user behaviour in spreadsheet error detection & correction. *arXiv preprint arXiv:0802.3479*.

Bradbard, D. A., Alvis, C., and Morris, R. (2014). Spreadsheet usage by management accountants: An exploratory study. *Journal of Accounting Education*, 32(4):24–30.

Chan, Y. E. and Storey, V. C. (1996). The use of spreadsheets in organizations: Determinants and consequences. *Information and Management*, 31(3):119–134.

Chen, Z. (2015). Spreadsheet framefinder. `https://github.com/chenzheruc/spreadsheetframefinder`.

Chen, Z. and Cafarella, M. (2013). Automatic web spreadsheet data extraction. *Proceedings of the 3rd International Workshop on Semantic Search Over the Web - SS@ '13*, pages 1–8.

Chen, Z., Cafarella, M., Chen, J., Prevo, D., and Zhuang, J. (2013). Senbazuru: a prototype spreadsheet database management system. *Vldb*, 6(12):1202–1205.

dan Pengembangan Bahasa, P. P., dan Pengembangan Bahasa (Indonesia), P. P., dan Kebudayaan, I. D. P., and Balai Pustaka, P. (1991). *Kamus Besar Bahasa Indonesia*. BP (Series). Balai Pustaka.

Dictionary, M.-W. L. (2016). *OnlyOffice*. Dipetik November 7, 2016, dari http://www.merriam-webster.com/dictionary/spreadsheet.

EuSpRIG, E. S. R. I. G. (2010a). *EuSpRIG Horror Stories*. Dipetik Oktober 23, 2016, dari http://www.eusprig.org/horror-stories.htm.

EuSpRIG, E. S. R. I. G. (2010b). *EuSpRIG Why Are We Here?* Dipetik Oktober 23, 2016, dari http://www.eusprig.org/about.htm.

Foundation, T. D. (2016). *LibreOffice*. Dipetik November 26, 2016, dari https://www.libreoffice.org.

Freeman, D. (1996). How to make spreadsheets error-proof. *Journal of Accountancy*, 181(5):75–77.

Google (2016). *Google Sheet*. Dipetik November 27, 2016, dari https://developers.google.com/apps-script/overview.

Howe, H. and Simkin, M. G. (2006). Factors Affecting the Ability to Detect Spreadsheet Errors. *NA*, 4(1):101–122.

Khatri, V. and Brown, C. V. (2010). Designing data governance. *Communications of the ACM*, 53(1):148.

Microsoft (2006). *Introducing the Office (2007) Open XML File Formats*. Dipetik November 17, 2016, dari https://msdn.microsoft.com/en-us/library/aa338205(v=office.12).aspx.

Microsoft (2015a). *Microsoft Office help and training - Office Support*. Dipetik November 17, 2016, dari https://support.office.com.

Microsoft (2015b). *Spreadsheet Software Program — Excel Free Trial*. Dipetik November 17, 2016, dari https://products.office.com/en/excel.

OASIS (2016). *OASIS Open Document Format for Office Applications (OpenDocument) TC*. Dipetik November 26, 2016, dari https://www.oasis-open.org/.

Panko, R. R. (1998). What We Know About Spreadsheet Errors. In *Journal of End User Computing's*, pages 15–21.

Powell, S. G., Baker, K. R., and Lawson, B. (2009). Impact of errors in operational spreadsheets. *Decision Support Systems*, 47(2):126–132.

Power, D. J. (2004). A brief history of spreadsheets. *DSSResources. com*.

Rajalingham, K., Chadwick, D. R., and Knight, B. (2001). Classification of Spreadsheet Errors. *Proc. European Spreadsheet Risks Int. Grp. (EuSpRIG)*, page 9.

Ronen, B., Palley, M. a., and Lucas, H. C. (1989). Spreadsheet analysis and design. *Communications of the ACM*, 32(1):84–93.

SIA, A. S. (2016). *OnlyOffice*. Dipetik November 27, 2016, dari http://www.onlyoffice.com/.

Tang, A. (2014). *EtherCalc*. Dipetik November 27, 2016, dari https://ethercalc.net/.

Tyszkiewicz, J. (2010). Spreadsheet as a relational database engine. *Proceedings of the 2010 international conference on Management of data - SIGMOD '10*, page 195.

Lampiran A. Detail Implementasi

Tabel A.1: Fungsi pada Kelas `player`

Fungsi	Deskripsi
refreshView	Melakukan pembaharuan tampilan sehingga menunjukkan tabel terbaru.
getDBSetting	Mengambil pengaturan basis data yang telah diberikan pengguna.
saveState	Menyimpan pengaturan yang telah dilakukan ke basis data.
loadState	Mengambil pengaturan yang pernah disimpan pada basis data.
saveConfig	Melakukan penyimpanan konfigurasi tabel yang dilakukan oleh pengguna.
deleteTable	Menghapus *metadata table* yang dipilih.
addManual	Menambahkan *metadata table* baru sesuai dengan masukan pengguna.
connect	Melakukan koneksi ke basis data yang dipilih.
save	Melakukan pemanggilan terhadap modul `table` dan melakukan penyimpanan ke basis data.
scan	Melakukan identifikasi tabel melalui pemanggilan modul `framefinder` yang selanjutnya akan menampilkan hasil identifikasi dan kolom perubahan konfigurasi yang dapat diisi pengguna.

Tabel A.2: Fungsi pada Kelas `mysql`

Fungsi	Deskripsi
createTable	Fungsi yang digunakan untuk membuat Table tempat pengisian data.
isTableExists	Melakukan pengecekan ada atau tidaknya tabel tersebut pada basis data.
getColumns	Mendapatkan kolom-kolom yang ada pada suatu tabel.
selectData	Mendapatkan data sesuai dengan syarat yang diminta.
insertData	Memasukkan data ke dalam tabel yang dipilih.
deleteData	Menghapus data dari tabel.
updateData	Melakukan pembaharuan data dari tabel.
dropTable	Menghapus tabel yang dipilih.

Tabel A.3: Kelas pada Modul `FrameFinder`

Nama Kelas	Deskripsi
LoadSheet	Kelas ini berfungsi sebagai kelas yang melakukan pengambilan data dan konversi sel dan *sheet* pada *spreadsheet* ke dalam bentuk kelas-kelas yang ada pada modul ini.
MySheet	Merupakan kelas bentukan yang merepresentasikan *sheet* pada *spreadsheet* yang dipilih.
MyCell	Merupakan kelas untuk merepresentasikan *properties* yang ada pada sel pada *sheet* yang dipilih.
FeatureSheetRow	Melakukan ekstraksi fitur-fitur yang terdapat pada suatu *sheet* pada *spreadsheet*.
PredictSheetRows	Kelas ini digunakan untuk menghasilkan file dalam format yang dapat dibaca oleh algoritma Conditional Random Field (CRF) dari fitur-fitur yang telah diekstraksi pada *sheet*.

Tabel A.4: Fungsi pada Kelas LoadSheet

Nama Fungsi	Deskripsi
loadSheetDict	Fungsi ini merupakan fungsi utama yang bertugas untuk membuat representasi *spreadsheet* yang diterima ke dalam kelas MySheet.
getValueType	Untuk mendapatkan tipe representasi data yang diberikan oleh *spreadsheet*. Contoh: tanggal, nominal uang, desimal, dan lain-lain.
getDataType	Untuk mendapatkan tipe data primitif pada suatu sel.
featureIndentation	Digunakan untuk mengecek keberadaan *property indentation* pada sel.
featureAlignStyle	Digunakan untuk mengecek keberadaan *property align* pada sel
featureFontBold	Digunakan untuk mengecek keberadaan *property bold* pada sel
featureFontHeight	Digunakan untuk mengecek keberadaan *property height* pada sel
featureFontUnderline	Digunakan untuk mengecek keberadaan *property underline* pada sel
featureFontItalic	Digunakan untuk mengecek keberadaan *property italic* pada sel
featureFontBgcolor	Digunakan untuk mengecek keberadaan *property background color* pada sel

Nama Fungsi	Deskripsi
featureBorderStyle	Digunakan untuk mengecek keberadaan *property border* pada sel.

Tabel A.5: Atribut pada Kelas MySheet

Nama Atribut	Deskripsi
sheetdict	Merupakan representasi kumpulan sel-sel pada suatu *sheet*. Tiap sel direpresentasikan dalam bentuk kelas MyCell.
mergerowdict	Merupakan kumpulan sel-sel yang digabungkan.
maxcolnum	Nilai kolom terbesar pada sel.
maxrownum	Nilai baris terbesar pada sel.

Tabel A.6: Fungsi pada Kelas MySheet

Nama Fungsi	Deskripsi
getCellsArray	Digunakan untuk mendapatkan seluruh refresentasi sel pada kelas ini dalam bentuk *array*.
addMergeCell	Digunakan pada saat terdapat sel yang digabungkan. Sel tersebut akan dimasukkan ke dalam daftar *merged cells*.
insertCell	Menambahkan sel ke dalam kelas ini. Sel yang ditambahkan akan direpresentasikan dalam bentuk kelas MyCell.

Tabel A.7: Atribut pada Kelas MyCell

Nama Atribut	Deskripsi
x	Merupakan letak sel pada koordinat X.
y	Merupakan letak sel pada koordinat Y.
w	Nilai lebar sel.
h	Nilai tinggi sel.
cstr	Isi sel dalam bentuk *string*.
mtype	Tipe konten yang ada di dalam sel.
indents	Nilai indentasi jika terdapat indentasi pada konten.
centeralign	Bernilai *true* atau *false* bergantung pada *align* sel merupakan rata tengah atau tidak.

Nama Atribut	Deskripsi
leftalign	Bernilai *true* atau *false* bergantung pada *align* sel merupakan rata kiri atau tidak.
rightalign	Bernilai *true* atau *false* bergantung pada *align* sel merupakan rata kanan atau tidak.
bottomborder	Bernilai *true* atau *false* bergantung pada *property bottom border* ada atau tidak.
upperborder	Bernilai *true* atau *false* bergantung pada *property upper border* ada atau tidak.
leftborder	Bernilai *true* atau *false* bergantung pada *property left border* ada atau tidak.
rightborder	Bernilai *true* atau *false* bergantung pada *property right border* ada atau tidak.
bold	Bernilai *true* atau *false* bergantung pada *property bold* ada atau tidak.
italic	Bernilai *true* atau *false* bergantung pada *property italic* ada atau tidak.
underline	Bernilai *true* atau *false* bergantung pada *property underline* ada atau tidak.

Tabel A.8: Fitur yang Diambil dari Sel

Fitur
Baris memiliki sel yang digabung
Sel pada baris mencapai kolom paling kanan
Sel pada baris mencapai kolom paling kiri
Baris hanya memiliki 1 kolom
Baris memiliki sel rata tengah
Baris memiliki sel rata kiri
Baris memiliki sel yang ditebalkan (*bold*)
Baris memiliki sel berindentasi
Baris memiliki sel berisi kata ‘Table‘
Baris memiliki sel berisi kata berawalan tanda baca
Baris memiliki sel dengan presentase angka tinggi
Baris memiliki sel berisi huruf besar seluruhnya
Baris memiliki sel berisi kata dengan awal huruf besar

Fitur
Baris memiliki sel berisi kata dengan awal huruf kecil
Baris memiliki persentase sel memiliki isi tinggi
Baris memiliki persentase sel memiliki isi berupa kata tinggi
Baris memiliki sel berisi karakter spesial
Baris memiliki sel berisi karakter titik koma
Baris memiliki jumlah sel berisi tahun tinggi
Baris memiliki persentase sel berisi tahun tinggi
Baris memiliki jumlah sel berisi kata dengan huruf yang banyak tinggi

Tabel A.9: Atribut pada Kelas Table

Nama Atribut	Deskripsi
sheet	Berisikan data untuk masing-masing sel pada suatu *sheet*.
rows	Representasi tabel yang berisikan; nama *header*, kolom data, dan aturan-aturan validasi.
range	*Range* sel data pada tabel.
title	Kumpulan baris yang diidentifikasi sebagai *title* oleh pengenalan otomatis.
footnote	Kumpulan baris yang diidentifikasi sebagai *footnote* oleh pengenalan otomatis.
header	Kumpulan baris yang diidentifikasi sebagai *header* oleh pengenalan otomatis.
data	Kumpulan baris yang diidentifikasi sebagai *data* oleh pengenalan otomatis.
name	Nama tabel yang akan dijadikan nama tabel pada basis data.

Tabel A.10: Fungsi pada Kelas LoadSheet

Nama Fungsi	Deskripsi
ParseData	Melakukan *parse* terhadap data dari *spreadsheet* menjadi bentuk objek tabel.
TupleSerializeWithChecker	Digunakan untuk mengubah struktur tabel menjadi tabel relasional dan melakukan validasi data masukan sesuai dengan aturan yang diminta pengguna.

Nama Fungsi	Deskripsi
Serialize	Mengubah objek tabel menjadi JSON.
Deserialize	Mengubah JSON menjadi objek tabel.
GetHTMLForm	Menghasilkan bentuk HTML dari objek tabel agar dapat ditampilkan pada antarmuka.

Lampiran B. Detail Pengujian

B.1 Skenario Pengujian

Skenario pengujian dilakukan berdasarkan kasus uji yang sudah dijabarkan pada Bab 4. Untuk setiap kasus uji, dapat mempunyai satu atau lebih skenario. Tiap skenario diberi kode TC-XX-YY dengan TC-XX adalah ID kasus uji dan YY adalah nomor skenario terkait ID kasus uji. Skenario pengujian dapat dilihat pada Tabel B.1.

Tabel B.1: Skenario Pengujian

ID Skenario	Deskripsi	Prosedur
TC-01-01	Validasi koneksi kepada pengaturan basis data yang diberikan dapat dilakukan saat konfigurasi benar	1. Masukkan konfigurasi basis data pada kolom yang tersedia. Pada pengujian, basis data yang dicoba berada di Docker dengan konfigurasi Host: 172.17.0.3 atau 172.17.0.2 Port: 3306 Username: root Password: root Database: TA 2. Tekan tombol 'Connect'
TC-01-02	Memberikan pesan kesalahan saat konfigurasi basis data tidak benar	1. Masukkan konfigurasi kosong pada kolom yang disediakan 2. Tekan tombol 'Connect'"

TC-01-03	Menyimpan dan mengambil data *metadata table* yang telah dibuat ke basis data	1. Menambahkan konfigurasi tabel baru secara manual dengan masukkan;
		{
		"header":[4],
		"data":"5:13",
		"range":[1,2,3,4,5,6,7]
		}
		2. Tambahkan manual
		3. Tutup browser
		4. Koneksikan kakas ke basis data yang sebelumnya
		5. Buka kembali browser ke spreadsheet tadi

ID Skenario	Deskripsi	Prosedur
TC-02-01	Menampilkan hasil pendeteksian label dan data secara otomatis	1. Menekan tombol ‘Detect Spreadsheet Table‘ pada antarmuka
TC-02-02	Pendeteksian header dengan hirarki dua atau lebih	1. Buat tabel yang dapat dideteksi dan memiliki header dengan hirarki dua atau lebih 2. Menekan tombol ‘Detect Spreadsheet Table‘ pada antarmuka

TC-03-01	Membuat *metadata table* baru sesuai masukan manual pengguna dimana masukan sesuai dengan aturan	1. Menambahkan konfigurasi tabel baru secara manual dengan masukkan;
		{
		”header”:[4],
		”data”:”5:13”,
		”range”:[1,2,3,4,5,6,7]
		}
		2. Tekan tombol ‘Add New Table‘

TC-03-02	Membuat *metadata table* baru sesuai masukan manual pengguna dimana masukan tidak sesuai dengan aturan	1. Menambahkan konfigurasi tabel baru secara manual dengan masukkan;
		{
		”header”:[5,6],
		”range”:”A9:F11”
		}
		2. Tekan tombol ‘Add New Table‘

TC-04-01	Mengubah atribut nama label pada baris menggunakan *metadata table*	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Ubah kolom label pada *metadata table* 3. Tekan tombol ‘Save‘ pada *metadata table* 4. Tutup browser 5. Buka kembali browser ke spreadsheet tadi
TC-04-02	Mengubah atribut ’kolom data’ pada *metadata table*	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Ubah kolom data pada *metadata table* 3. Tekan tombol ‘Save‘ pada *metadata table* 4. Tutup browser 5. Buka kembali browser ke spreadsheet tadi

ID Skenario	Deskripsi	Prosedur
TC-04-03	Mengubah tipe data pada suatu kolom menggunakan *metadata table*	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Ubah kolom tipe data pada *metadata table* 3. Tekan tombol 'Save' pada *metadata table* 4. Tutup browser 5. Buka kembali browser ke spreadsheet tadi
TC-04-04	Mengubah nilai yang diizinkan pada suatu kolom menggunakan *metadata table*	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Ubah kolom permitted values pada *metadata table* 3. Tekan tombol 'Save' pada *metadata table* 4. Tutup browser 5. Buka kembali browser ke spreadsheet tadi
TC-04-05	Mengubah relasi yang diinginkan pada suatu kolom menggunakan *metadata table*	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Ubah kolom relation pada *metadata table* menjadi range cell atau suatu tabel dan kolom unik pada basis data 3. Tekan tombol 'Save' pada *metadata table* 4. Tutup browser 5. Buka kembali browser ke spreadsheet tadi
TC-04-06	Menghapus *metadata table*	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Hapus *metadata table* yang sudah ditambahkan 3. Tekan tombol 'Save' pada *metadata table* 4. Tutup browser 5. Buka kembali browser ke spreadsheet tadi
TC-04-07	Menentukan kolom key	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Centang pilihan tipe key pada suatu kolom 3. Tekan tombol 'Save' pada *metadata table* 4. Tekan tombol 'Save to Database'

ID Skenario	Deskripsi	Prosedur
TC-05-01	Data dimasukkan ke basis data pertama kali dimana belum ada data sebelumnya	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Tekan tombol 'Save to Database'
TC-05-02	Nama pada dua *metadata table* pada spreadsheet yang sama, saling ditukar satu dengan yang lain. Contohnya, tabel 1 bernama A dan tabel 2 bernama B menjadi tabel 1 bernama B dan tabel 2 bernama A.	1. Tambahkan *metadata table* baik secara manual atau otomatis sehingga minimal terdapat dua tabel 2. Tekan tombol 'Save to Database' 3. Tukar nama kedua tabel tersebut 4. Tekan tombol 'Save to Database'
TC-05-03	Spreadsheet lain menggunakan nama tabel yang sama dengan konfigurasi kolom yang sama, tanpa kolom key	1. Tambahkan *metadata table* baik secara manual atau otomatis, tanpa kolom key 2. Tekan tombol 'Save to Database' 3. Pindah ke spreadsheet lain dan tambahkan *metadata table* baru dengan nama dan jumlah kolom yang sama 4. Tekan tombol 'Save to Database'
TC-05-04	Spreadsheet lain menggunakan nama tabel yang sama dengan konfigurasi kolom yang sama, dengan nama kolom key yang sama	1. Tambahkan *metadata table* baik secara manual atau otomatis, dengan satu kolom key 2. Tekan tombol 'Save to Database' 3. Pindah ke spreadsheet lain dan tambahkan *metadata table* baru dengan nama dan jumlah kolom yang sama dan sebuah kolom key 4. Tekan tombol 'Save to Database'

ID Skenario	Deskripsi	Prosedur
TC-05-05	Spreadsheet lain menggunakan nama tabel yang sama dengan konfigurasi kolom yang sama, dengan nama kolom key yang berbeda	1. Tambahkan *metadata table* baik secara manual atau otomatis, dengan kolom key 2. Tekan tombol 'Save to Database' 3. Pindah ke spreadsheet lain dan tambahkan *metadata table* baru dengan nama dan jumlah kolom yang sama dan kolom unik yang berbeda 4. Tekan tombol 'Save to Database'
TC-05-06	Spreadsheet lain menggunakan nama tabel yang sama dengan konfigurasi kolom yang berbeda dimana sudah ada spreadsheet lain yang menggunakan, tanpa kolom key	1. Tambahkan *metadata table* baik secara manual atau otomatis, tanpa kolom key 2. Tekan tombol 'Save to Database' 3. Pindah ke spreadsheet lain dan tambahkan *metadata table* baru dengan nama dan jumlah kolom yang berbeda seluruhnya tanpa kolom key 4. Tekan tombol 'Save to Database'
TC-05-07	Spreadsheet lain menggunakan nama tabel yang sama dengan konfigurasi kolom yang berbeda dimana sudah ada spreadsheet lain yang menggunakan, dengan nama kolom key yang sama	1. Tambahkan *metadata table* baik secara manual atau otomatis, dengan sebuah kolom key 2. Tekan tombol 'Save to Database' 3. Pindah ke spreadsheet lain dan tambahkan *metadata table* baru dengan nama dan jumlah kolom yang berbeda, dengan satu kolom key yang sama 4. Tekan tombol 'Save to Database'
TC-05-08	Spreadsheet lain menggunakan nama tabel yang sama dengan konfigurasi kolom yang berbeda dimana sudah ada spreadsheet lain yang menggunakan, dengan nama kolom unik yang berbeda	1. Tambahkan *metadata table* baik secara manual atau otomatis, dengan suatu kolom unik 2. Tekan tombol 'Save to Database' 3. Pindah ke spreadsheet lain dan tambahkan *metadata table* baru dengan nama dan jumlah kolom yang berbeda, dan dengan kolom unik yang berbeda juga 4. Tekan tombol 'Save to Database'

ID Skenario	Deskripsi	Prosedur
TC-05-09	Menggunakan nama tabel yang sama dengan konfigurasi kolom yang berbeda dimana belum ada spreadsheet lain yang menggunakan	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Tekan tombol 'Save to Database' 3. Ganti nama *metadata table* sebelumnya menjadi nama lain lalu gunakan *metadata table* lain dengan kolom yang berbeda menjadi nama tabel lama. 4. Tekan tombol 'Save to Database'
TC-05-10	Penghapusan suatu konfigurasi tabel	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Tekan tombol 'Save to Database' 3. Hapus *metadata table* 4. Tekan tombol 'Save to Database'
TC-05-11	Penggantian nama tabel pada *metadata table*	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Tekan tombol 'Save to Database' 3. Ganti nama *metadata table* sebelumnya menjadi nama lain. 4. Tekan tombol 'Save to Database'

ID Skenario	Deskripsi	Prosedur
TC-05-12	Memasukkan data dengan fragmen gabungan pada dua atau lebih spreadsheet secara tidak beraturan, namun data yang ada tidak saling menimpa	Terdapat data yang akan dimasukkan berupa NIM, Nilai 1, Nilai 2, dan Nilai 3. Terdapat tiga kelompok NIM. 1. Dibuat spreadsheet A yang akan mengisi data pada NIM kelompok 1 untuk Nilai 1 dan 3, dan NIM kelompok 3 untuk Nilai 1. 2. Dibuat spreadsheet B yang akan mengisi data pada NIM kelompok 1 untuk Nilai 2, dan NIM kelompok 2 untuk Nilai 3. 3. Dibuat spreadsheet C yang akan mengisi data pada NIM kelompok 2 untuk Nilai 1 dan 2, dan NIM kelompok 3 untuk Nilai 2. 4. Setiap spreadsheet dimasukkan datanya ke dalam basis data dengan urutan yang berbeda-beda
TC-06-01	Memvalidasi tipe data Integer	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah tipe data kolom menjadi Integer 3. Tekan tombol ‘Save to Database‘
TC-06-02	Memvalidasi tipe data Double	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah tipe data kolom menjadi Double 3. Tekan tombol ‘Save to Database‘
TC-06-03	Memvalidasi tipe data String	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah tipe data kolom menjadi String 3. Tekan tombol ‘Save to Database‘

ID Skenario	Deskripsi	Prosedur
TC-06-04	Memvalidasi tipe data Boolean	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah tipe data kolom menjadi Boolean 3. Tekan tombol ‘Save to Database‘
TC-06-05	Memvalidasi masukan yang diizinkan berupa nilai diskrit	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah kolom permitted value menjadi [”Alabama”, ”Seattle”] 3. Tekan tombol ‘Save to Database‘
TC-06-06	Memvalidasi masukan yang diizinkan berupa nilai range	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah kolom permitted value menjadi 10-1000 3. Tekan tombol ‘Save to Database‘
TC-06-07	Memvalidasi masukan yang diizinkan dengan syarat kurang dari	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah kolom permitted value menjadi < 1000 3. Tekan tombol ‘Save to Database‘
TC-06-08	Memvalidasi masukan yang diizinkan dengan syarat kurang dari sama dengan	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah kolom permitted value menjadi ≤ 1000 3. Tekan tombol ‘Save to Database‘
TC-06-09	Memvalidasi masukan yang diizinkan dengan syarat lebih dari	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah kolom permitted value menjadi > 1000 3. Tekan tombol ‘Save to Database‘

ID Skenario	Deskripsi	Prosedur
TC-06-10	Memvalidasi masukan yang diizinkan dengan syarat lebih dari sama dengan	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah kolom permitted value menjadi ≥ 1000 3. Tekan tombol ‘Save to Database‘
TC-06-11	Memvalidasi relasi antar sel	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah kolom relasi menjadi suatu range sel. Contohnya A8:B9 3. Tekan tombol ‘Save to Database‘
TC-06-12	Memvalidasi relasi dengan tabel	1. Tambahkan *metadata table* baik secara manual atau otomatis 2. Pada *metadata table* yang telah ada, ubah kolom relasi menjadi target tabel dan kolom. Contohnya [”nama_tabel”, ”nama_kolom”] 3. Tekan tombol ‘Save to Database‘
TC-06-13	Memvalidasi konflik yang terjadi jika terjadi penyimpanan	1. Buat suatu spreadsheet dan masukkan data menggunakan *metadata table* dan memiliki kolom unik 2. Buat spreadsheet lain dengan nama kolom yang sama dan kolom unik yang juga sama. 3. Tekan tombol ‘Check Conflict‘ untuk memastikan adanya konflik 4. Ubah nama kolom yang bukan merupakan kolom unik pada spreadsheet kedua 5. Tekan kembali tombol ‘Check Conflict‘

B.2 Hasil Pengujian

Hasil pengujian berkaitan dengan ID skenario pengujian yang telah dijabarkan sebelumnya. Tiap hasil pengujian mempunyai ID skenario terkait, ekspektasi, hasil uji yang dilakukan, dan diterima atau tidaknya hasil pengujian. Hasil pengujian dapat dilihat pada Tabel B.2.

Tabel B.2: Hasil Pengujian

ID Skenario	Ekspektasi	Hasil	Diterima
TC-01-01	Kakas dapat terkoneksi ke basis data dan dapat melakukan perintah-perintah SQL kepada basis data yang dipilih	Berhasil melakukan koneksi ke basis data dan dapat melakukan perintah-perintah SQL yang dibutuhkan, serta dapat menginisiasi tabel kosong dengan konfigurasi yang dibutuhkan.	Ya
TC-01-02	Muncul pesan kesalahan yang menjelaskan bahwa koneksi belum dapat dilakukan	Muncul pesan 'Cannot Established Connection to Database' pada layar dan informasi untuk mencoba memperbaiki pengaturan	Ya
TC-01-03	*metadata table* yang pernah dibuat, muncul kembali walaupun aplikasi ditutup	Konfigurasi terakhir, muncul kembali setelah terkoneksi ke basis data yang sama.	Ya
TC-02-01	Muncul *metadata table* sesuai dengan label dan data yang terdeteksi	Pada kasus yang dapat ditangani oleh model, muncul tabel-*metadata table* sesuai dengan jumlah tabel, letak header, dan data yang di deteksi. Jika tidak, muncul pesan 'No Table Detected'	Ya
TC-02-02	Header yang berhirarki akan bernama header1_header2 dan seterusnya	Header berhirarki berhasil dideteksi, sesuai dengan tingkat hirarkinya. Tingkat hirarki yang dicoba adalah satu, dua, dan tiga tingkat	Ya

ID Skenario	Ekspektasi	Hasil	Diterima
TC-03-01	*metadata table* baru ditambahkan ke layar dan sesuai dengan masukkan pengguna	*metadata table* berhasil ditambahkan dan ditampilkan kepada pengguna	Ya
TC-03-02	Muncul pesan kesalahan dan tidak muncul tambahan *metadata table*	Muncul pesan kesalahan berupa kesalahan yang terjadi karena tidak adanya bagian data dan *metadata table* tidak ditambahkan	Ya
TC-04-01	Label pada *metadata table* berubah dan tersimpan pada basis data	Label yang disimpan, muncul kembali walaupun browser telah ditutup dan konfigurasinya tersimpan dibasis data. Pada saat penyimpanan, tabel dibentuk dengan nama kolom yang sama dengan nama label	Ya
TC-04-02	Kolom data pada *metadata table* berubah dan tersimpan pada basis data	Perubahan kolom data yang disimpan, muncul kembali walaupun browser telah ditutup dan konfigurasinya tersimpan dibasis data. Pada saat penyimpanan, data yang masuk sesuai dengan kolom data yang dipilih	Ya
TC-04-03	Tipe data pada *metadata table* berubah dan tersimpan pada basis data	Perubahan tipe data yang disimpan, muncul kembali walaupun browser telah ditutup dan konfigurasinya tersimpan dibasis data	Ya

ID Skenario	Ekspektasi	Hasil	Diterima
TC-04-04	Nilai yang diizinkan pada *metadata table* berubah dan tersimpan pada basis data	Perubahan nilai yang diizinkan (permitted value) berhasil disimpan dan muncul kembali walaupun browser telah ditutup	Ya
TC-04-05	Relasi pada *metadata table* berubah dan tersimpan pada basis data	Perubahan rujukan relasi berhasil disimpan dan muncul kembali walaupun browser telah ditutup	Ya
TC-04-06	*metadata table* yang dihapus tidak muncul lagi dan terhapus dari basis data. Data yang terhapus dari basis data hanya data yang berhubungan dengan *metadata table.*	*metadata table* berhasil dihapus dari basis data dan tidak muncul kembali pada saat dibuka kembali walaupun browser telah ditutup.	Ya
TC-04-07	Jika kolom tidak unik, maka akan menghasilkan error. Jika sudah unik, akan dimasukkan ke basis data.	Key hanya bisa dapat dipilih untuk maksimal satu kolom. Pada saat kolonm tersebut memiliki nilai yang tidak unik, akan muncul pesan kesalahan. Jika unik, maka kolom dapat dijadikan key. Pengaturan key juga akan muncul kembali walaupun broswer telah ditutup	Ya
TC-05-01	Tabel baru dibuat dan data masuk ke dalam basis data	Tabel berhasil dibuat dan data berhasil masuk ke dalam tabel yang sesuai.	Ya

ID Skenario	Ekspektasi	Hasil	Diterima
TC-05-02	Data yang ada pada tabel tersebut berubah menjadi data baru	Data pada kedua tabel berhasil ditukar dengan baik jika belum ada spreadsheet lain yang memasukkan datanya pada tabel yang ingin ditukar. Jika sudah, maka tabel gagal ditukar.	Ya
TC-05-03	Terdapat data dari dua spreadsheet pada satu tabel. Digabungkan secara horizontal	Data dari kedua spreadsheet atau lebih berhasil digabungkan secara horizontal pada suatu tabel.	Ya
TC-05-04	Terjadi penimpaan data.	Data disimpan duluan akan tertimpa oleh data yang disimpan setelahnya sesuai dengan key data tersebut.	Ya
TC-05-05	Tidak dapat digabungkan	Tabel tidak dapat digabungkan dan muncul pesan kesalahan.	Ya
TC-05-06	Tidak dapat digabungkan	Tabel tidak dapat digabungkan dan muncul pesan kesalahan.	Ya
TC-05-07	Terdapat data dari dua spreadsheet pada satu tabel. Digabungkan secara horizontal. Jika kolom key sama, maka digabungkan vertikal.	Data dari kedua spreadsheet atau lebih berhasil digabungkan secara vertikal dimana data yang berbeda, digabungkan menurut key yang dipilih. Jika tidak ada key yang bersesuaian, maka digabungkan secara horizontal	Ya

ID Skenario	Ekspektasi	Hasil	Diterima
TC-05-08	Tidak dapat digabungkan	Tabel tidak dapat digabungkan dan muncul pesan kesalahan.	Ya
TC-05-09	Data lama dihapus, lalu jumlah dan nama kolom disesuaikan dengan konfigurasi baru.	Tabel berhasil dihapus, dan datanya diganti dengan data baru dengan kolom yang baru.	Ya
TC-05-10	Data yang tadinya berhubungan dengan *metadata table* yang dihapus, ikut terhapus	Data yang berhubungan dengan *metadata table* tersebut berhasil dihapus dari baris pada tabel yang bersangkutan dan jika tabel kosong, maka akan dilakuakan penghapusan tabel pada basis data.	Ya
TC-05-11	Data yang berhubungan dengan *metadata table* pada tabel dengan nama awal, dipindahkan ke tabel dengan nama baru. Data yang berhubungan dengan *metadata table* pada tabel dengan nama awal dihapus.	Data yang berhubungan dengan *metadata table* awal berhasil dihapus, serta dipindahkan ke tabel dengan nama yang baru.	Ya
TC-05-12	Data dapat masuk seluruhnya tanpa harus berurutan dan data yang masuk bersesuaian dengan kolom yang diinginkan	Data berhasil dimasukkan tanpa harus memperhitungkan urutan masukan dengan syarat data telah *disjoint* satu dengan yang lainnya. Jika tidak, maka urutan masukkan data harus diperhatikan.	Ya

ID Skenario	Ekspektasi	Hasil	Diterima
TC-06-01	Jika data Integer, maka data akan masuk. Jika tidak, maka akan muncul pesan kesalahan.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya
TC-06-02	Jika data Double, maka data akan masuk. Jika tidak, maka akan muncul pesan kesalahan.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya
TC-06-03	Jika data String, maka data akan masuk. Jika tidak, maka akan muncul pesan kesalahan.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya
TC-06-04	Jika data Boolean, maka data akan masuk. Jika tidak, maka akan muncul pesan kesalahan.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya

ID Skenario	Ekspektasi	Hasil	Diterima
TC-06-05	Jika data berada di daftar nilai diskrit, maka data akan masuk ke basis data. Jika tidak, maka akan muncul pesan kesalahan.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya
TC-06-06	Jika data berada pada rentang yang ditentukan, maka data akan masuk ke basis data. Jika tidak, maka akan muncul pesan kesalahan.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya
TC-06-07	Jika data berada pada rentang yang ditentukan, maka data akan masuk ke basis data. Jika tidak, maka akan muncul pesan kesalahan.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya
TC-06-08	Jika data berada pada rentang yang ditentukan, maka data akan masuk ke basis data. Jika tidak, maka akan muncul pesan kesalahan.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya

ID Skenario	Ekspektasi	Hasil	Diterima
TC-06-09	Jika data berada pada rentang yang ditentukan, maka data akan masuk ke basis data. Jika tidak, maka akan muncul pesan kesalahan.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya
TC-06-10	Jika data berada pada rentang yang ditentukan, maka data akan masuk ke basis data. Jika tidak, maka akan muncul pesan kesalahan.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya
TC-06-11	Jika data ada yang tidak berada pada rentang sel relasi yang ditentukan, maka akan muncul pesan kesalahan dan seluruh data tidak akan masuk ke basis data. Selain itu, data akan masuk ke basis data.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya
TC-06-12	Jika data ada yang tidak berada pada tabel di kolom yang ditentukan, maka akan muncul pesan kesalahan dan seluruh data tidak akan masuk ke basis data. Selain itu, data akan masuk ke basis data.	Data berhasil divalidasi. Jika terdapat kesalahan, muncul pesan error yang menunjukkan sel yang salah. Jika sudah benar, maka data akan masuk ke basis data.	Ya

ID Skenario	Ekspektasi	Hasil	Diterima
TC-06-13	Pada saat ada konflik data, maka akan keluar error. Pada saat tidak ada konflik, akan diberikan info bahwa tidak ada konflik.	Jika terdapat kemungkinan bahwa pada saat penyimpanan terjadi penimpaan terhadap data yang ada, maka akan muncul pesan error. Jika tidak, maka akan muncul pesan bahwa tidak terjadi konflik.	Ya