BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Program magang difokuskan pada FullStack Development di bawah pengawasan Bapak Sugito, yang berperan sebagai supervisor dan pemimpin tim untuk analisis dan pemeliharaan sistem. Koordinasi proyek dilakukan di Github, yang berfungsi sebagai *platform* untuk transfer dan pengambilan proyek. Repositori Github memiliki dua cabang, yaitu pengembangan dan master. WebStorm dipilih sebagai lingkungan pengembangan lokal karena integrasinya dengan Git. Pengerjaan proyek dilakukan di dalam cabang pengembangan, dan untuk memasukkan pembaruan di WebStorm, operasi penarikan dari cabang pengembangan dilakukan. Kemajuan didokumentasikan, dikomitmenkan, dan kemudian didorong ke cabang pengembangan. Setelah komit, proyek Github menjalani proses pembangunan menggunakan Jenkins, yang juga melibatkan analisis kode melalui SonarQube.

Pemantauan dilakukan melalui komunikasi pribadi di Whatsapp atau Google Meet, atau dalam grup obrolan yang telah dibuat untuk komunikasi yang transparan. Interaksi dengan departemen lain difasilitasi melalui grup Whatsapp atau Google Meet untuk rapat antar-divisi. Dalam kasus-kasus di mana tugas dilakukan dalam pengaturan bekerja dari kantor (WFO), tidak ada keharusan untuk menggunakan Whatsapp atau Google Meet, karena pertemuan dapat dilakukan secara langsung.

3.2 Tugas yang Dilakukan

Tugas yang dilakukan selama kerja magang di PT Cranium Royal Aditama, yaitu:

1. Mengembangkan struktur database dan ER Description untuk menggambarkan hubungan antar database secara lebih rinci. Mengembangkan struktur database dan mendokumentasikan hubungan antar database secara lebih rinci, melibatkan beberapa entitas seperti customer type, user parameter, item withdraw reservation setting, forgot, confirm, dan user. Dalam perancangan database ini, diberlakukan beberapa aturan, termasuk penggunaan garis bawah (underscore) untuk memisahkan

kata pada nama kolom yang terdiri dari dua kata atau lebih. Panjang nama kolom tidak boleh melebihi 25 karakter, dan jika diperlukan, kolom dapat disingkat dengan kata umum. Referensi utama dalam perancangan database ini adalah website ERP yang sudah ada. Setelah perancangan selesai, supervisor melakukan pemeriksaan dan berdiskusi mengenai penambahan atau pengurangan kolom.

Keseluruhan perancangan tersebut didokumentasikan dalam sebuah file yang disebut ER-Description. ER-Description berfungsi sebagai dokumen penyimpanan database yang merinci setiap kolom secara lebih spesifik. Isi dari ER-Description mencakup tipe data, ukuran data, kunci primer, apakah data dapat berisi nilai null atau tidak, kunci asing, nilai default jika kolom tidak diisi, apakah data bersifat unik, penggunaan indeks antar database, dan keterangan tambahan yang relevan dengan setiap kolom. Akhirnya, migrasi database pada PostgreSQL dilaksanakan dengan merujuk pada ER-Description yang telah disusun sebelumnya.

2. Membuat API untuk metode CRUD dan paging untuk sub-modul Customer Type, User Parameter, dan Item Withdraw Reservation Setting. Pengembangan API dilaksanakan menggunakan IDE IntelliJ Idea. API yang dihasilkan disimpan dalam sebuah class yang dikenal sebagai controller. Fungsi utama dari controller adalah untuk menerima dan melakukan validasi terhadap input dari pengguna ketika suatu API diakses. Controller kemudian memanggil Service yang bertanggung jawab menjalankan berbagai manipulasi data dan logika bisnis yang ingin diterapkan pada data yang diakses melalui Repository. Repository, pada gilirannya, merupakan sebuah *class* yang berperan dalam berinteraksi secara langsung dengan database. *Class* Repository mengandung berbagai *query* yang digunakan dalam proses CRUD. Setiap operasi yang ingin mengakses atau memperoleh data melalui database harus diimplementasikan dengan memanggil *class* Repository.

3. Melakukan testing pada sistem *backend*. Setelah merancang API di *backend*, proses pengujian dilakukan menggunakan IntelliJ IDEA dan Postman. Dalam IntelliJ IDEA, file pengujian yang dikenal sebagai unit test dibuat untuk menguji langsung *class* Service yang telah dibuat. Pengujian unit mencakup situasi ketika API mengalami kegagalan dan ketika API berhasil dijalankan. Selain pengujian unit, terdapat juga file groovy yang berfungsi sebagai dokumentasi untuk *request* dan *response*, yang nantinya dapat

digunakan saat melakukan pengujian dengan Postman. Selanjutnya, *bug* yang ditemukan baik melalui pengujian maupun melalui pemeriksaan di SonarQube diperbaiki.

4. Langkah berikutnya adalah mengembangkan tampilan *frontend* menggunakan kerangka kerja React TypeScript, yang terhubung dengan modul yang sudah dibuat di *backend*. Terdapat beberapa pedoman, seperti setiap *file* dari submodul harus disimpan dalam direktori yang sesuai dengan modul dari submodul tersebut. Direktori setiap modul terdiri dari API, Type, dan Features. Folder API berisi jalur API *backend* yang diakses oleh *frontend*. Folder Type berisi entitas dalam bentuk objek yang diperoleh dari API. Folder Features digunakan untuk menyimpan berbagai komponen yang digunakan untuk membangun halaman situs *web*. Selain itu, ada folder layout yang berisi berbagai *template* tata letak situs *web* yang dapat digunakan untuk berbagai halaman. Terakhir, ada folder *page* yang digunakan untuk merender berbagai komponen pada folder features, sehingga menghasilkan halaman situs *web* yang lengkap.

5. Langkah terakhir melibatkan pengujian pada halaman *frontend* yang telah disusun. Dalam proses pengujian ini, beberapa folder menjadi penting, yaitu folder mocks dan tests. Folder tests berisi berbagai *file* pengujian, sementara folder mocks memuat API dan *database* imitasi yang diciptakan. Ini menciptakan ilusi bahwa *file* pengujian melakukan akses ke API dan mengambil data dari *database*, padahal sebenarnya data yang diterima dikembalikan melalui folder mocks.

3.3 Uraian Pelaksanaan Magang

Pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

Tabel 3.1. Tabel Daftar Kegiatan Kerja Magang Mingguan

Minggu Ke -	Pekerjaan yang setiap minggu
1	Latihan *frontend* ERP, melakukan pemasangan Webstorm, dan latihan terkait React Typescript dan MUI bersama supervisor.
Lanjut pada halaman berikutnya	

Tabel 3.1. Tabel Daftar Kegiatan Kerja Magang Mingguan (Lanjutan)

Minggu Ke -	Pekerjaan yang setiap minggu
2	Mulai melakukan cloning terhadap salah satu modul dari Master yaitu Customer Type. Cloning yang dilakukan terkait fitur *create*, *update*, *read*, *delete*, dan *paging* serta melakukan revisi pada backend agar menyesuaikan dengan pengembangan frontend
3	Mepelajari konsep *styling* dan UI dari figma yang diberikan supervisor dan membuat desain *frontend* untuk halaman landing page.
4	Membuat desain *frontend* untuk halaman forgot password dan mengikuti training terkait proses backend forgot password untuk proses pengembangan.
5	Membuat backend untuk proses forgot password agar dapat digunakan untuk *frontend*.
6	Mengintegrasikan backend forgot password ke *frontend* forgot password.
7	Membuat backend untuk proses confirm password agar dapat digunakan untuk *frontend*.
8	Mengintegrasikan backend confirm password ke *frontend* confirm password.
9	Memperbaiki *bug* yang ditemukan pada forgot password dan error pada confirm password.
10	Membuat *unit test* untuk melakukan *testing* proses confirm password.
11	Melanjutkan pembuatan *unit test* untuk melakukan *testing* proses confirm password hingga selesai dan memperbaiki *bug* yang ditemukan selama pengerjaan.
12	Mulai melakukan cloning terhadap salah satu modul dari Master yaitu Item Withdraw Reservation Settings. Cloning yang dilakukan terkait fitur *create*, *update*, *read*, *delete*, dan *paging* serta melakukan revisi pada backend agar menyesuaikan dengan pengembangan frontend
Lanjut pada halaman berikutnya	

Tabel 3.1. Tabel Daftar Kegiatan Kerja Magang Mingguan (Lanjutan)

Minggu Ke -	Pekerjaan yang setiap minggu
13	Mulai melakukan cloning terhadap salah satu modul dari Master yaitu User Parameter. Cloning yang dilakukan terkait fitur *create*, *update*, *read*, *delete*, dan *paging* serta melakukan revisi pada backend agar menyesuaikan dengan pengembangan frontend
14	Mengubah *backend* dan *frontend* User Parameter agar dapat melakukan *auto-complete* pada create dan update. *Auto-complete* diambil dari data modul lain yaitu Parameter dan User.
15	Memperbaiki bug pada *auto-complete* user pada modul user parameter agar dpaat berfungsi dengan baik.
16	Membuat *unit test* untuk modul Customer Type untuk melakukan *testing* pada CRUD dan *paging*.
17	Membuat *unit test* untuk modul Item Withdraw Reservation Setting untuk melakukan *testing* pada *method* CRUD *paging*.
18	Membuat *unit test* untuk modul User Parameter untuk melakukan *testing* pada *method create*, *read*, *update* , *delete*, dan *paging*.

3.3.1 *Software* dan *Hardware* yang digunakan

A. *Software*

Software yang digunakan selama mengembangkan sistem ERP ialah sebagai berikut.

1. Intellij IDEA

   IntelliJ IDEA adalah salah satu produk dari JetBrains berupa *Integrated Development Environment* (IDE). IDE ini dioptimalkan untuk pengembangan software menggunakan Java dan Kotlin [5]. Saat menjalani program magang, IntelliJ IDEA menjadi pilihan utama sebagai IDE untuk pengembangan backend. IDE ini direkomendasikan oleh supervisor untuk pengembangan sistem ERP.

2. WebStorm

WebStorm adalah suatu *Integrated Development Environment* (IDE) yang diproduksi oleh JetBrains. IDE ini dirancang khusus untuk mempermudah pengembangan situs web menggunakan berbagai bahasa pemrograman seperti Typescript, CSS, Javascript, dan bahasa pemrograman lainnya [6]. Dalam pengembangan *website* ERP dengan menggunakan *framework* React Typescript, WebStorm menjadi pilihan utama sebagai *Integrated Development Environment* (IDE) untuk mengelola sisi *frontend*.

3. pgAdmin 4

   pgAdmin 4 adalah sebuah *platform open-source administration and development* yang dikembangkan untuk PostgreSQL, sering kali digunakan dalam pengembangan sistem manajemen database relasional (RDBMS) [7]. *Software* ini dirancang dengan tujuan menyediakan antarmuka pengguna grafis (*graphical user interface* atau GUI) yang memungkinkan pengelolaan dan interaksi dengan *database* PostgreSQL. Dalam lingkup pelaksanaan magang, pgAdmin 4 dimanfaatkan untuk tugas seperti pembuatan, pengubahan, dan penghapusan tabel atau *database*.

4. Postman

   Postman adalah *software* pengembangan dan alat uji API yang bertujuan mempermudah proses interaksi dengan API [8]. Dalam konteks pembuatan sistem ERP, *tools* ini dimanfaatkan untuk melakukan uji terhadap API yang telah dikembangkan sebelumnya.

5. Jenkins

   Jenkins merupakan sebuah server otomatisasi sumber terbuka yang dirancang dengan dukungan untuk berbagai bahasa pemrograman [9]. Peran utamanya melibatkan pelaksanaan proses build, pengujian, dan deploying dalam proyek pengembangan perangkat lunak. Selama periode magang, Jenkins berfungsi sebagai simulasi tahap produksi, menjalankan pengujian unit dan mengevaluasi hasilnya di luar lingkungan server lokal, baik *frontend* maupun *backend*

6. SonarQube

   SonarQube adalah sebuah *platform open source* yang dirancang untuk memantau dan mengidentifikasi kualitas proyek perangkat lunak secara teratur yang menyediakan informasi statistik keamanan *code*, *bug*, dan *unit*

test coverage. SonarQube juga memberikan informasi terperinci terkait quality code yang dihasilkan oleh para pengembang [10].

7. Github

   Github adalah suatu platform berbasis *web* yang berfokus pada *version control* dan mempermudah pengembangan *software* secara bersama-sama [11]. Dalam pengembangan sistem ERP, Github digunakan untuk mempermudah kerja sama internal.

B. *Hardware*

Hardware yang dipakai selama tahap pengembangan sistem ERP adalah laptop ASUS TUF Gaming A15, yang memiliki spesifikasi sebagai berikut:

- Processor: AMD Ryzen 5 4600H
- Memori: 16 GB *Dual Channel*
- Kartu Grafis: NVIDIA GeForce GTX 1650
- Resolusi Layar: 1920x1080p
- Dimensi Layar: 15”
- Penyimpanan: 512 GB
- Sistem Operasi: Windows 11

3.3.2 *User Requirement*

Dengan merujuk pada desain sistem ERP yang telah disusun, berikut adalah kebutuhan pengguna (*user requirement*) yang telah ditetapkan.

1. Admin mampu melakukan proses CRUD untuk proses administrasi melalui menu Customer Type, Item Withdraw Reservation Settings, dan User Parameter.
2. User dapat melakukan *request* untuk reset password dari akun yang dimiliki melalui menu Forgot Password.

3. User dapat melakukan konfirmasi pergantian password dari akun yang dimiliki melalui tautan yang dikirimkan pada email user menuju ke menu Confirm Password

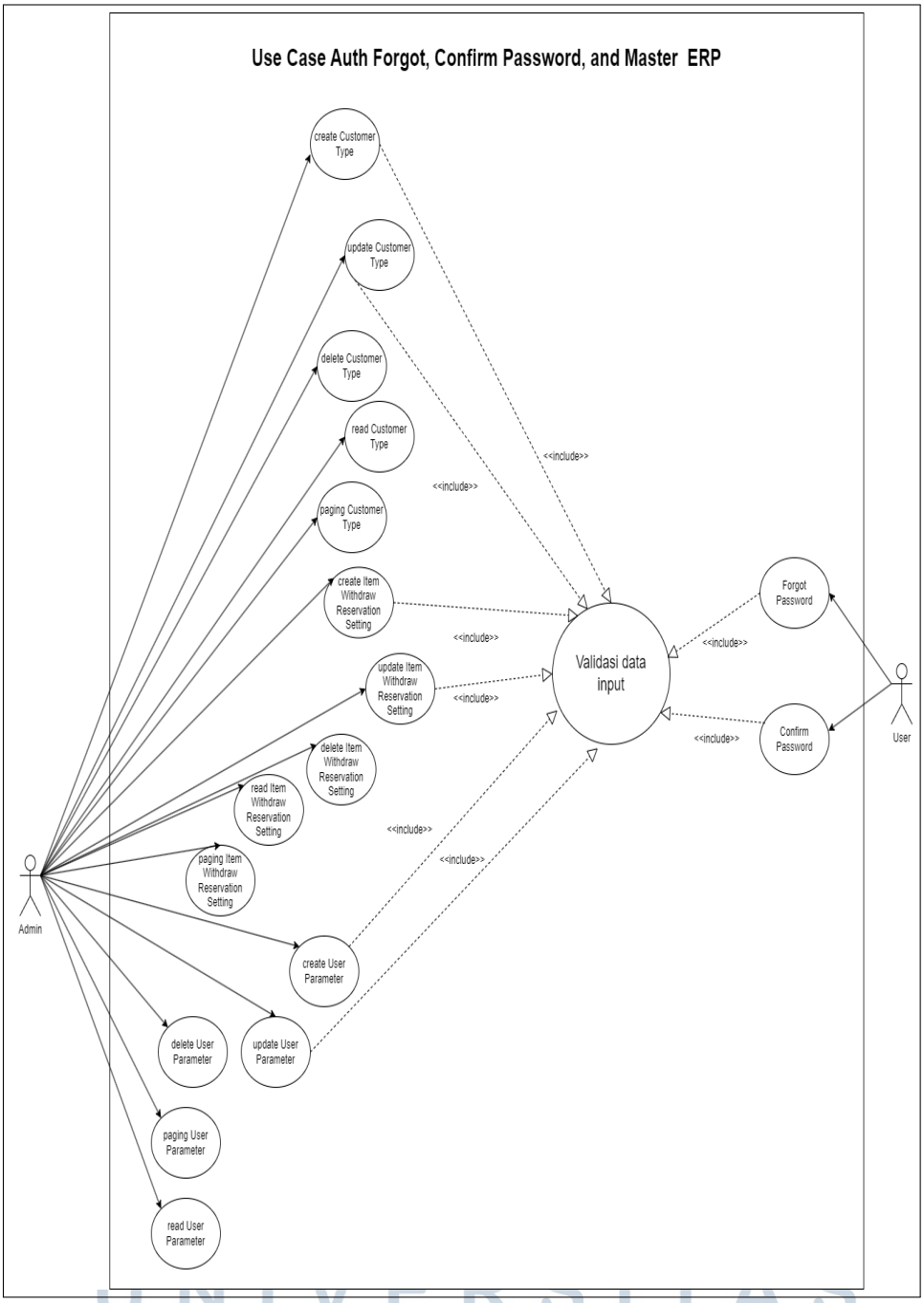
4. Sistem perlu memiliki kemampuan untuk mencegah adanya kesalahan pada *input* data yang dimasukkan oleh *user*.

5. Sistem perlu membatasi user yang tidak berwenang mengakses modul Master pada sub-modul Customer Type, Item Withdraw Reservation Settings, dan User Parameter.

3.4 Perancangan dan Hasil

3.4.1 Use Case Diagram

Diagram Kasus Pengguna (*Use Case Diagram*) adalah diagram berbasis objek yang digunakan sebagai metode analisis sistem. Diagram ini bertujuan untuk mengidentifikasi, menjelaskan, dan mengorganisir kebutuhan sistem yang sedang dikembangkan. Kasus pengguna) memperlihatkan hubungan antara entitas diluar sistem dengan sistem [12]. Untuk mengilustrasikan hubungan antara entitas admin saat berinteraksi dengan sistem ERP, terutama pada modul master sub-modul customer type, item withdraw reservation setting, dan user parameter serta *user* secara umum dalam berinteraksi dengan sistem untuk fitur forgot dan confirm password dapat dilihat pada Gambar 3.1.



Gambar 3.1. *Use Case Diagram*

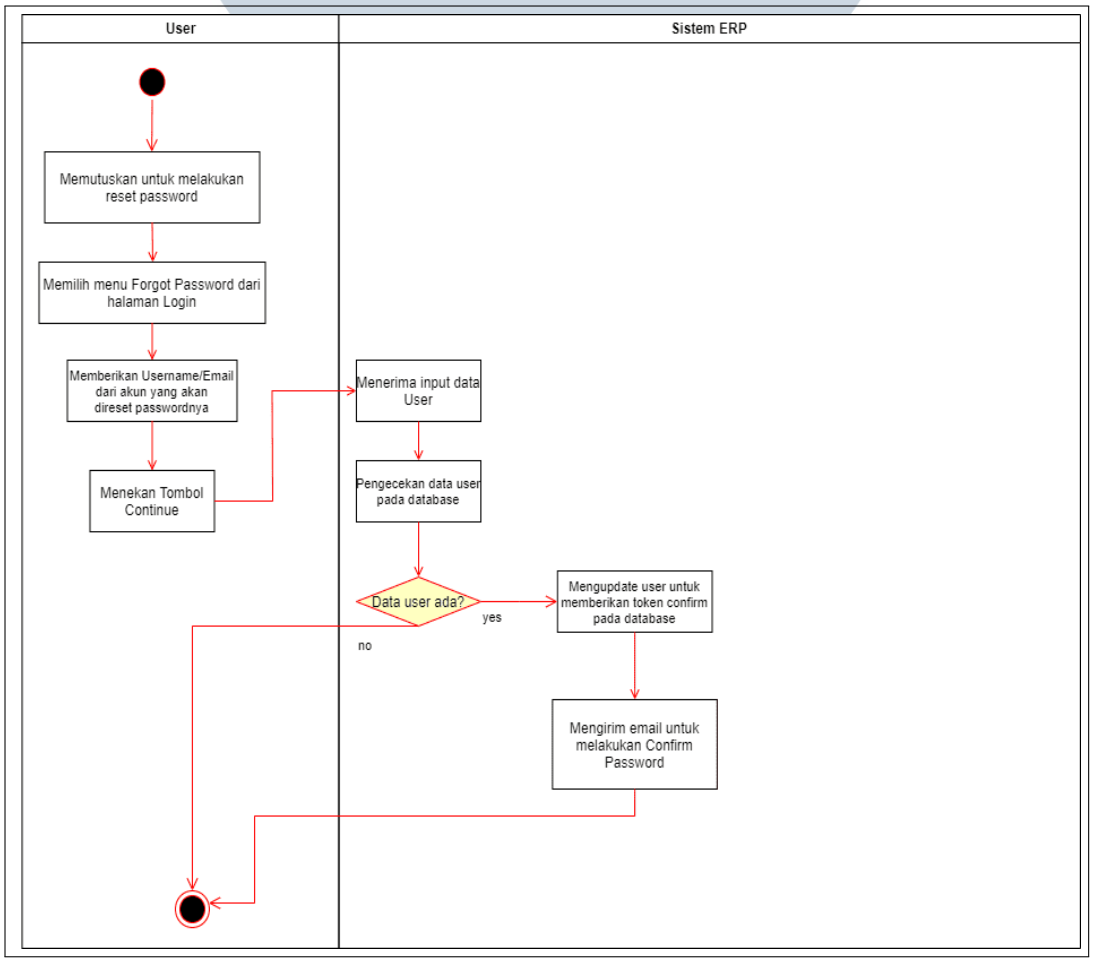
Admin memiliki akses untuk modul Master pada sub-modul Customer Type, Item Withdraw Reservation Setting, dan User Parameter, sedangkan User memiliki akses untuk fitur Forgot Password dan Confirm Password. Admin dapat langsung menggunakan fitur *create*, *read*, *update*, *paging*, dan *delete* dari sistem ERP sesuai dengan modul yang dikehendaki. Pada saat admin melakukan *create* dan *update*,

maka *input* data pada *form create* dan *update* dilakukan validasi data *input* pada prosesnya. Sedangkan user mendapatkan akses langsung untuk fitur forgot dan confirm password melalui home page.

3.4.2 *Activity Diagram*

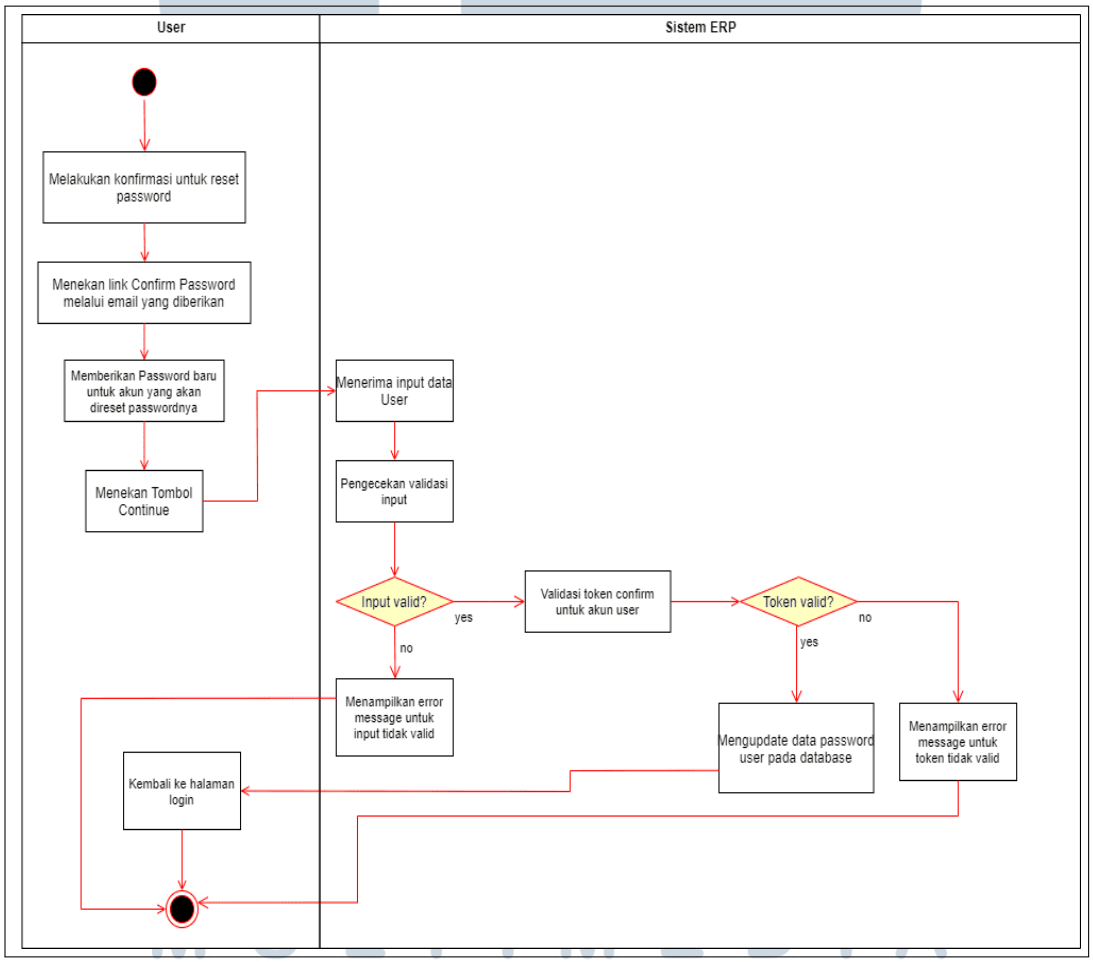
Activity Diagram digunakan untuk muntuk memberikan sebuah gambaran alur pengerjaan atau proses. Diagram aktivitas memiliki tujuan untuk memberikan penjelasan rinci tentang kasus penggunaan baik secara mandiri atau dengan cara yang lebih rumit. Diagram ini berkonsentrasi pada ilustrasi tindakan dalam eksekusi sistem dan kondisi yang dapat memengaruhi alur kerja sistem [13].

Berikut merupakan *activity diagram* dari modul forgot dan confirm password, submodul customer type, user parameter, dan item withdraw reservation setting.



Gambar 3.2. *Activity Diagram* Forgot Password

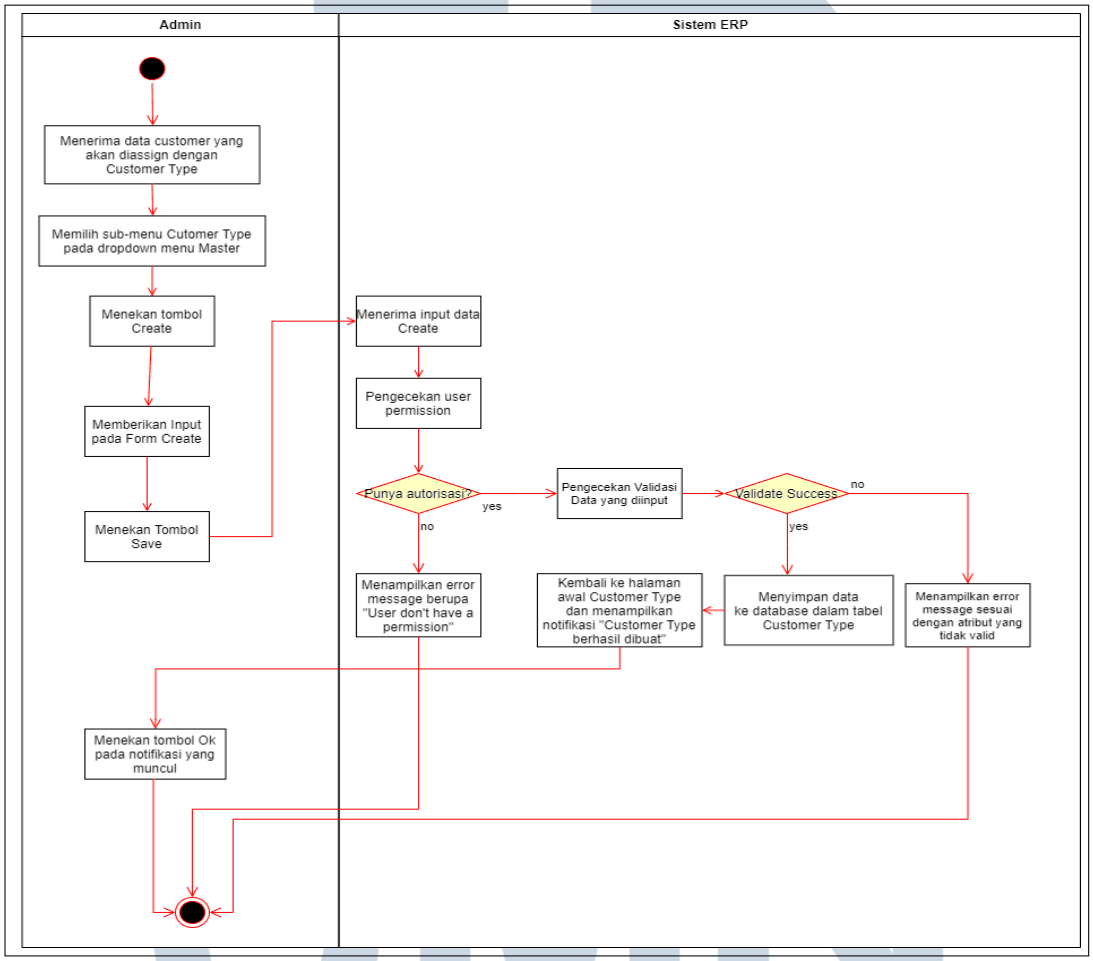
Activity diagram untuk fitur forgot password dapat dilihat pada Gambar 3.2. Proses forgot password dimulai saat aktor, dalam hal ini merupakan user memutuskan untuk melakukan reset password pada akun. User menekan tombol Forgot Password dari halaman login untuk menuju halaman forgot password. Pada halaman forgot password, user diminta untuk memasukkan *email* / *username* dari akun yang direset serta menekan tombol continue. Saat tombol continue ditekan, sistem ERP menerima input user dan melakukan pengecekan untuk akun user yang direset passwordnya pada database. Jika data user ada pada database, sistem melakukan update untuk memberikan token confirm pada user dan mengirim *email* kepada user untuk melakukan confirm password.



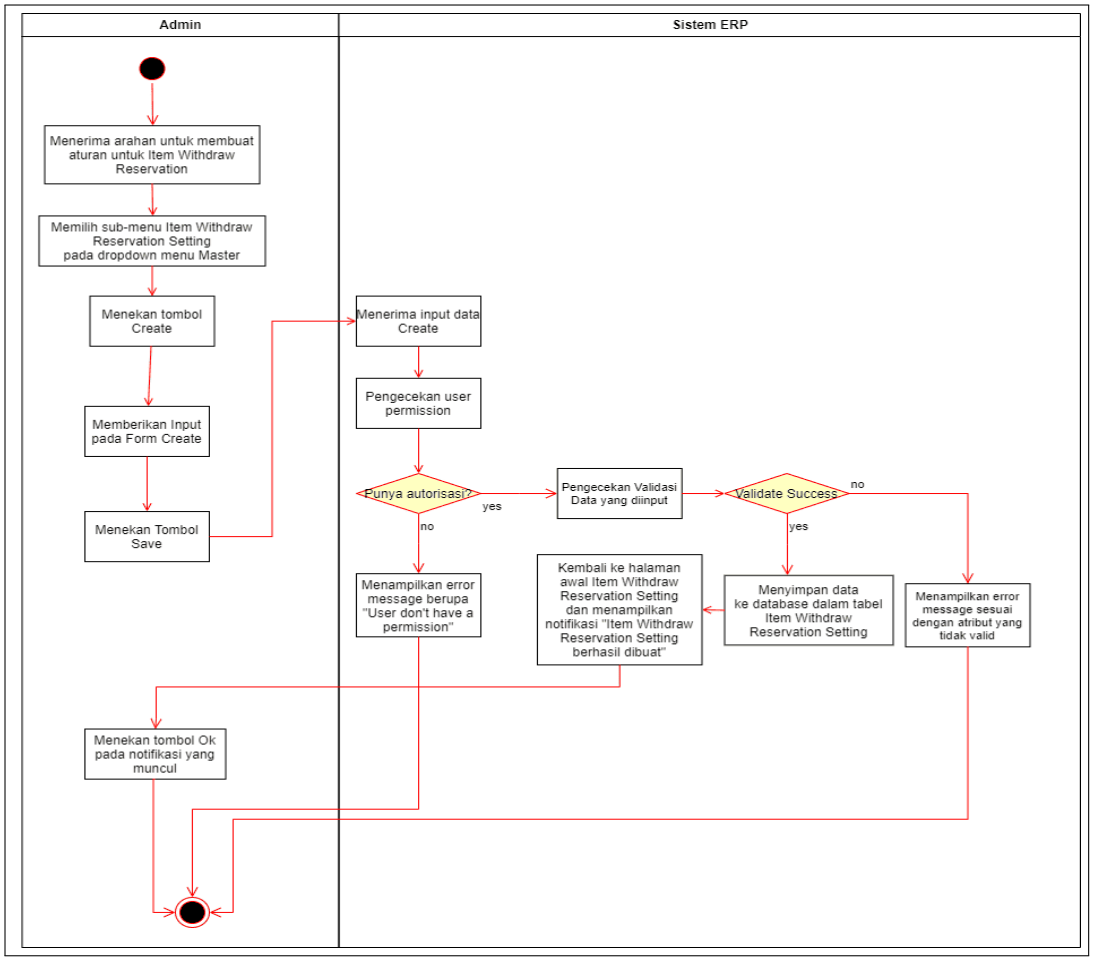
Gambar 3.3. *Activity Diagram* Confirm Password

Activity diagram untuk fitur confirm password dapat dilihat pada Gambar 3.3.User mengakses confirm password melalui email yang dikirimkan dari forgot password. User mengisi password baru untuk akunnya pada halaman confirm

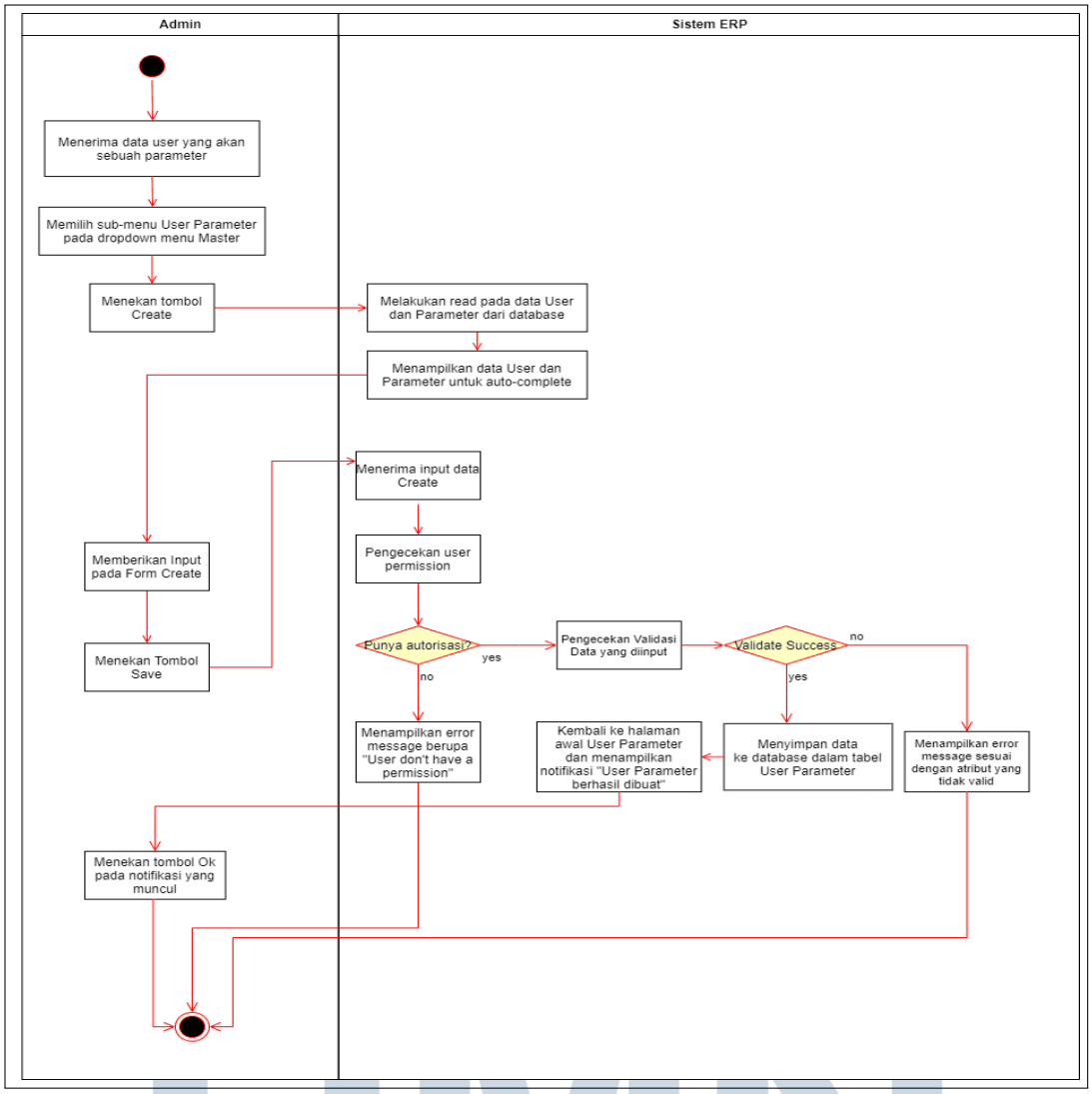
password. Saat ditekan tombol continue, sistem menerima input password dan mengecek validasi password tersebut. Jika password valid maka dilakukan cek apakah token confirm yang diberikan valid dan jika valid maka data password baru diupdate untuk akun user tersebut.



Gambar 3.4. *Activity Diagram Create* Customer Type



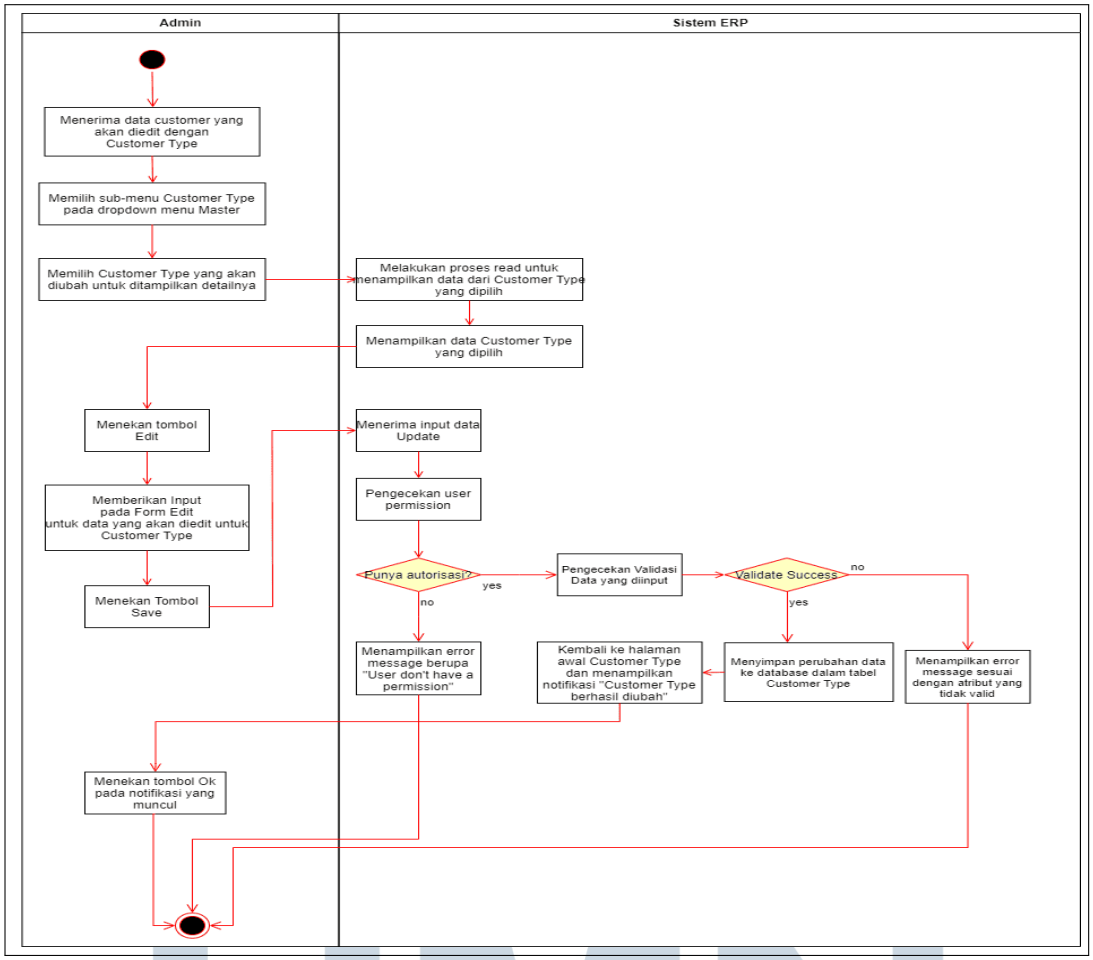
Gambar 3.5. *Activity Diagram Create* Item Withdraw Reservation Setting



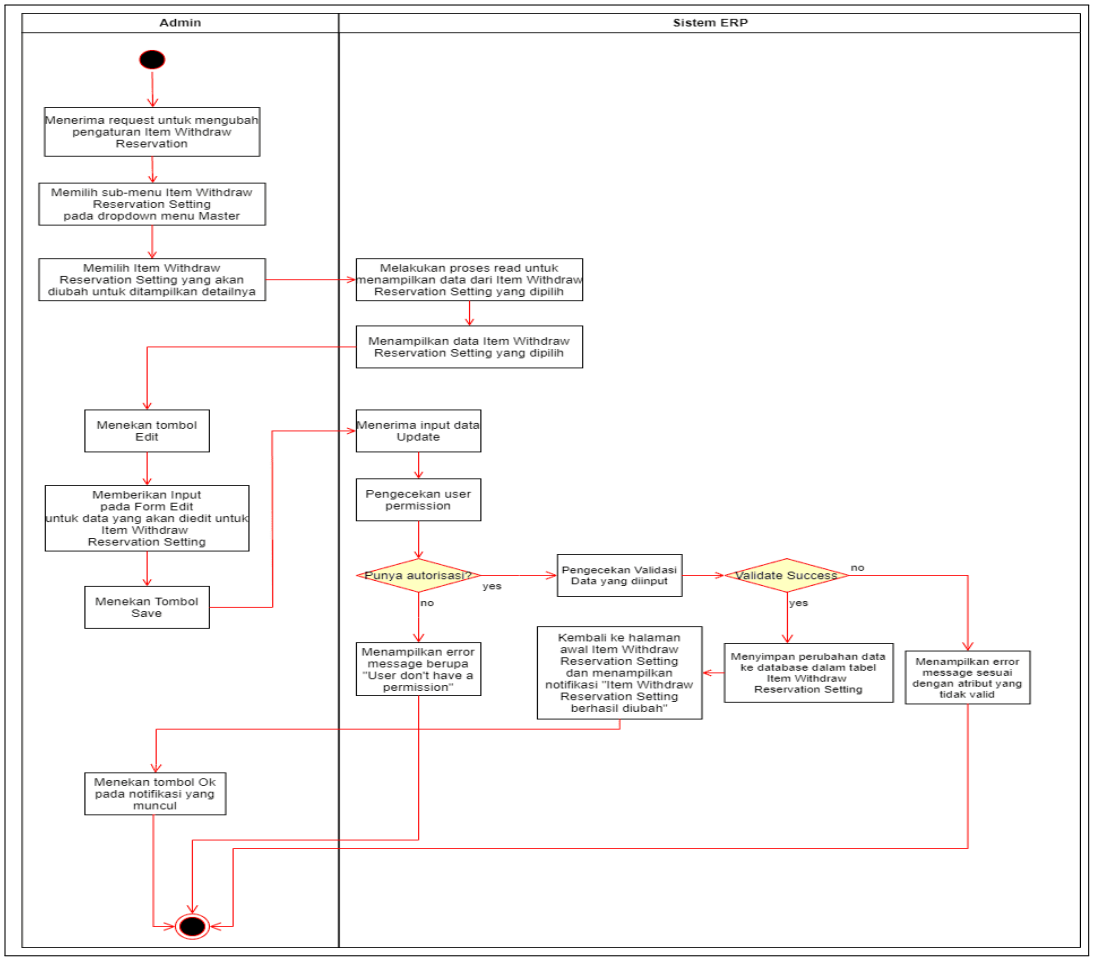
Gambar 3.6. *Activity Diagram Create* User Parameter

Activity diagram untuk fitur *create* submodul customer type, item withdraw reservation setting, dan user parameter dapat dilihat pada Gambar 3.4, 3.5, dan 3.6. Dalam proses fitur *create* pada sistem ERP, admin memasuki sistem melalui submenu yang diinginkan dari menu Master. Admin menginisiasi pembuatan data dengan menekan opsi *create* memberikan input melalui formulir yang disediakan, dan menyimpan informasi dengan menekan tombol *save* Setelah menerima input, sistem melakukan pemeriksaan terhadap izin pengguna dan melakukan validasi terhadap data yang dimasukkan. Setelah menyelesaikan pemeriksaan izin dan validasi data, sistem menjalankan proses untuk metode penciptaan guna menyimpan data ke dalam *database*. Untuk proses *create* user parameter, saat ditekan tombol *create* dibaca terlebih dahulu data user dan parameter untuk mengisi auto-complete

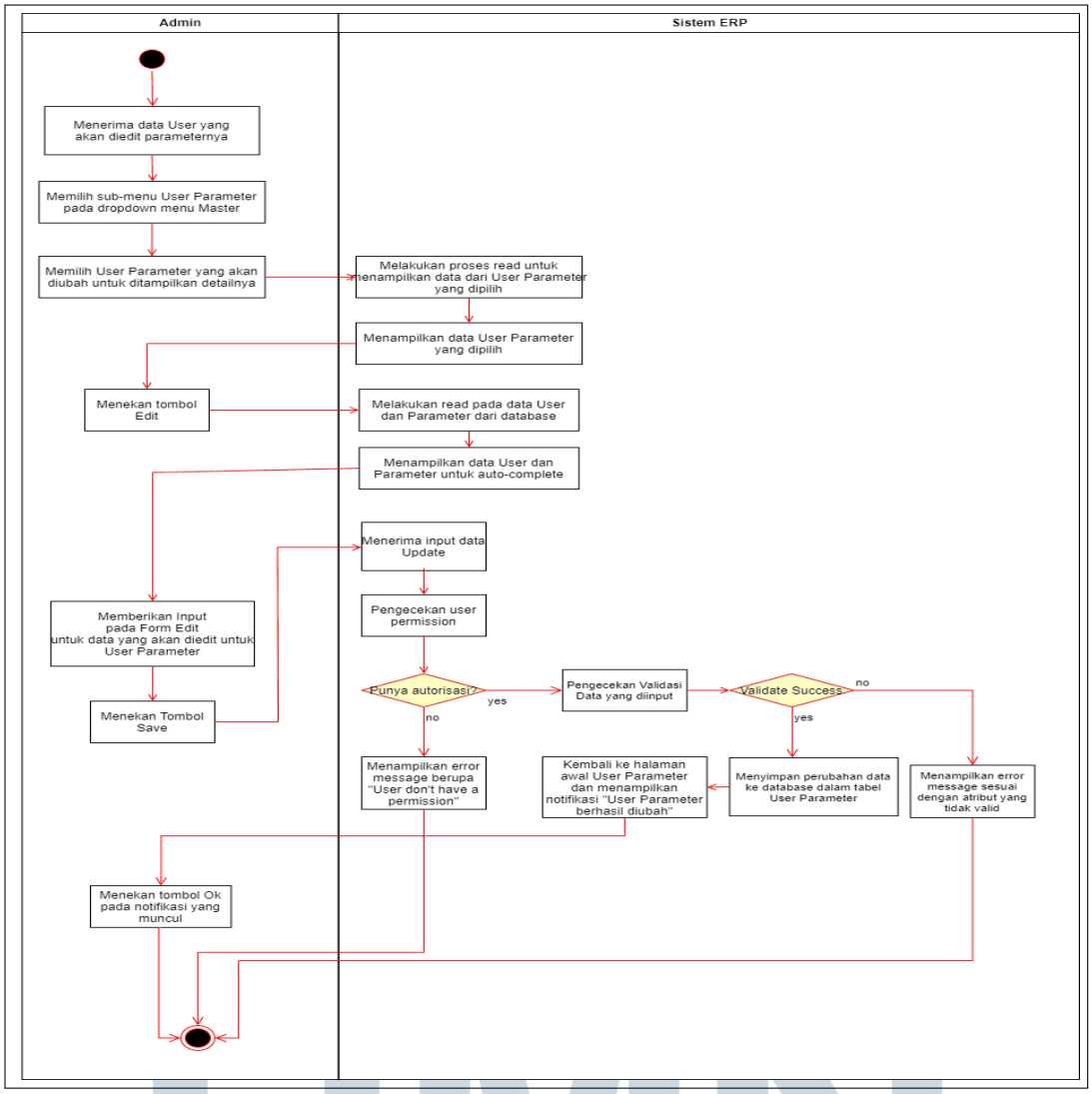
pada form.



Gambar 3.7. *Activity Diagram Update* Customer Type



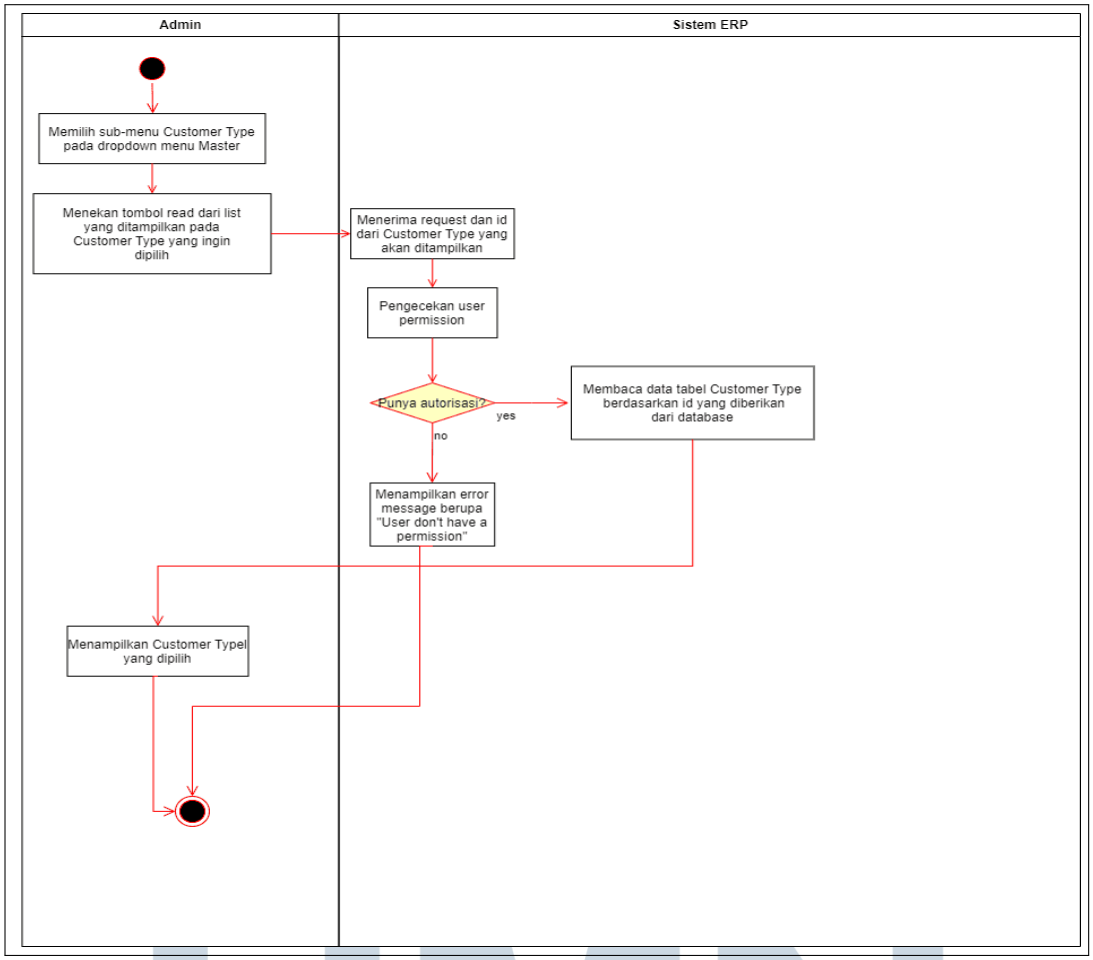
Gambar 3.8. *Activity Diagram Update* Item Withdraw Reservation Setting



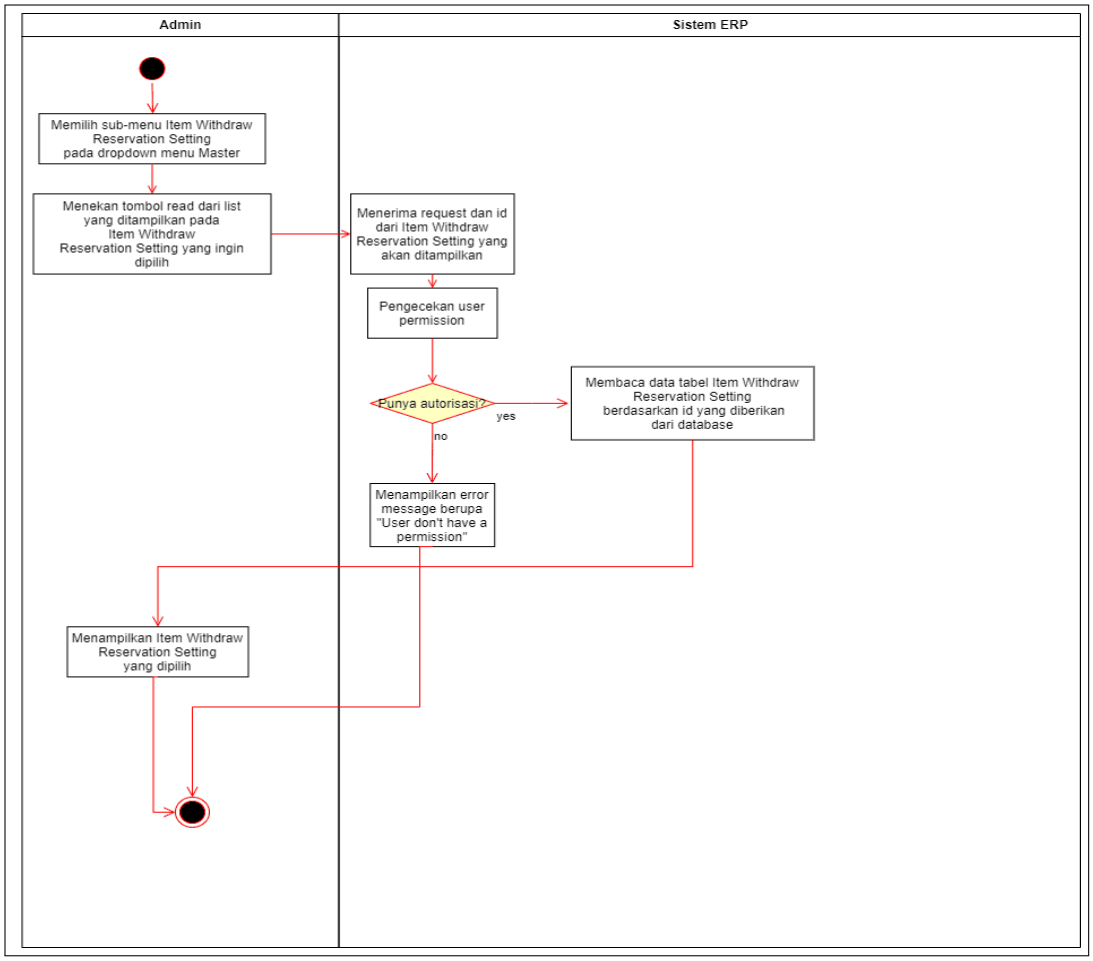
Gambar 3.9. *Activity Diagram Update* User Parameter

Activity diagram untuk fitur *update* submodul customer type, item withdraw reservation setting, dan user parameter dapat dilihat pada Gambar 3.7, 3.8, dan 3.9. Untuk memodifikasi data, administrator perlu menavigasi ke submenu di dalam modul Master. Administrator memilih data yang diinginkan dari daftar item yang tersedia. Sistem kemudian menampilkan detail dari item yang dipilih dan menampilkan formulir untuk mengedit data yang sesuai. Administrator melengkapi formulir dan menyimpan perubahan dengan mengklik tombol Simpan. Setelah menerima input data yang telah diubah, sistem melakukan verifikasi izin dan memastikan keabsahan data sebelum menyimpan perubahan tersebut di dalam tabel database yang relevan. Mengenai proses pembaruan parameter pengguna, ketika tombol *create* diaktifkan, sistem awalnya membaca data pengguna dan parameter

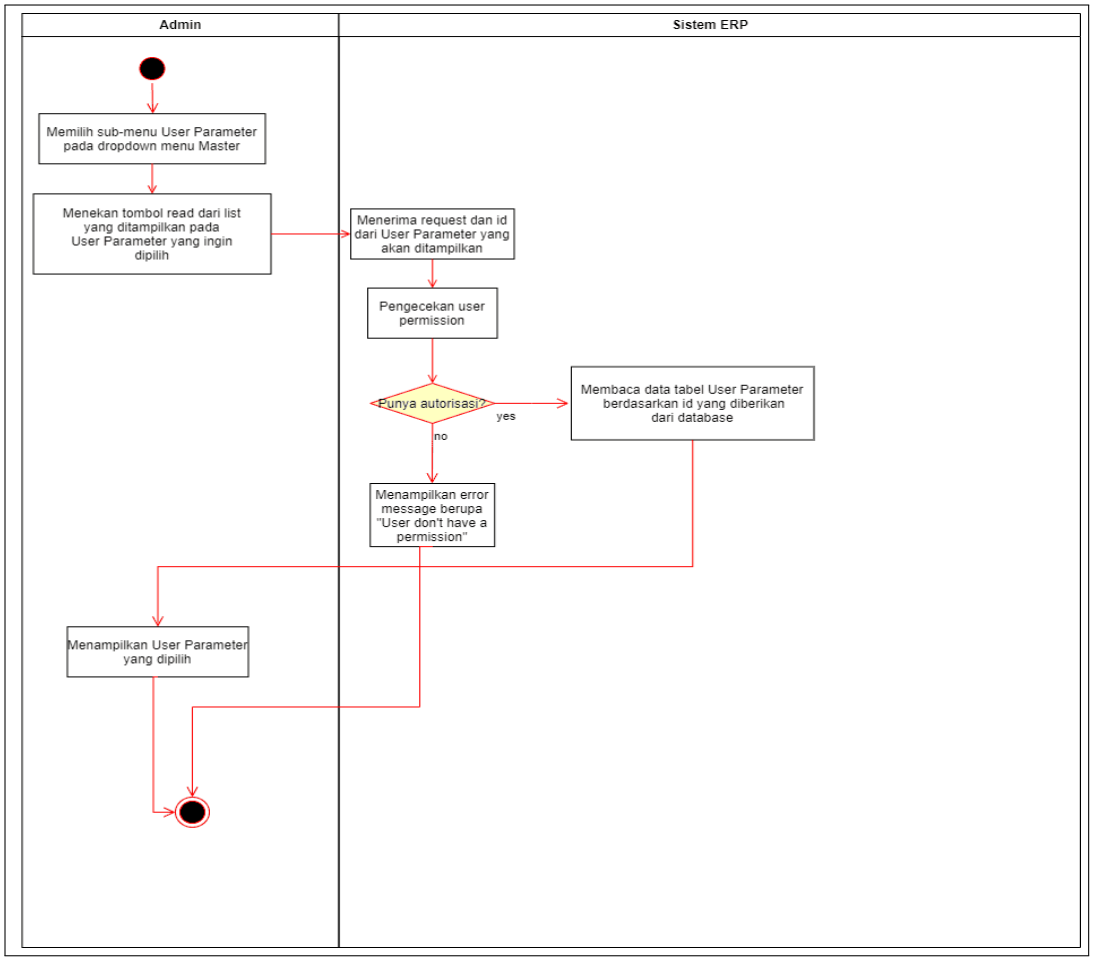
untuk mengisi formulir secara otomatis.



Gambar 3.10. *Activity Diagram Read* Customer Type

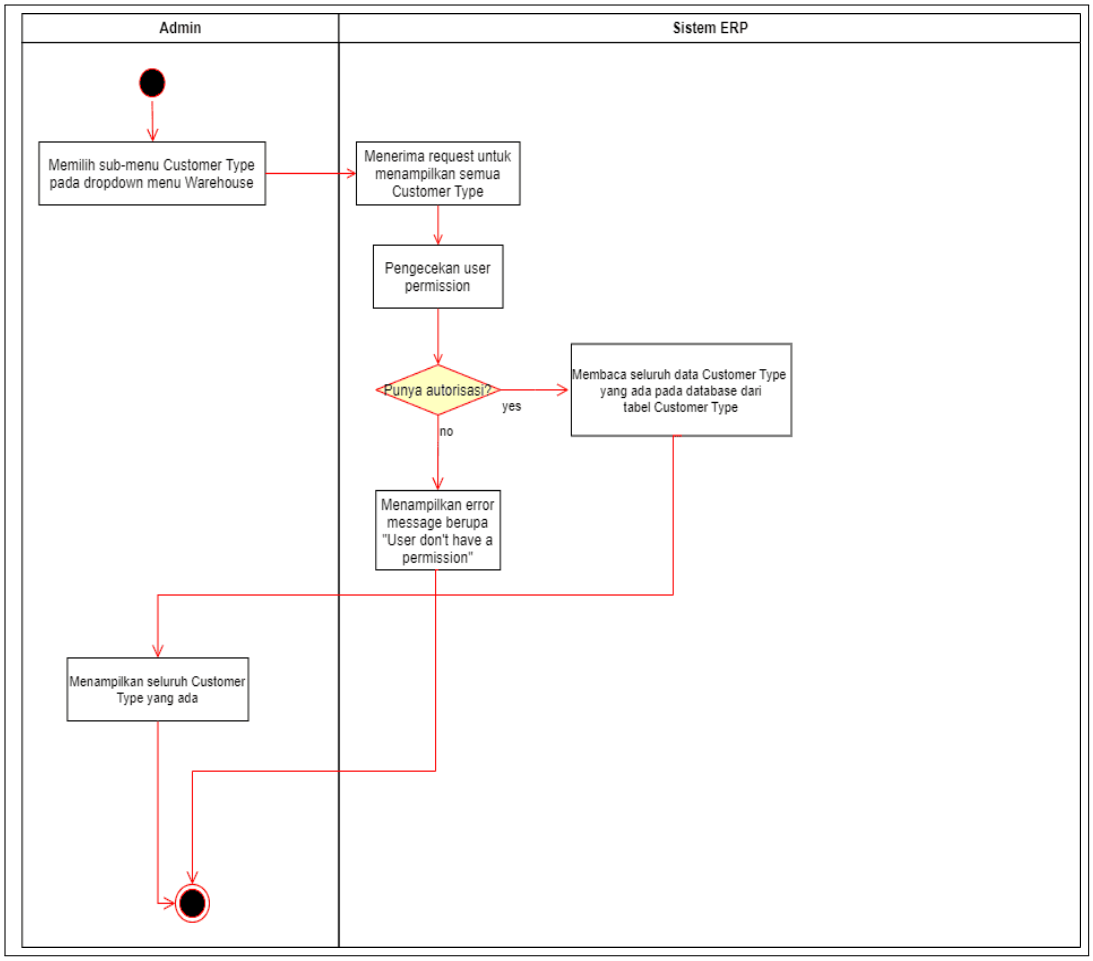


Gambar 3.11. *Activity Diagram Read* Item Withdraw Reservation Setting

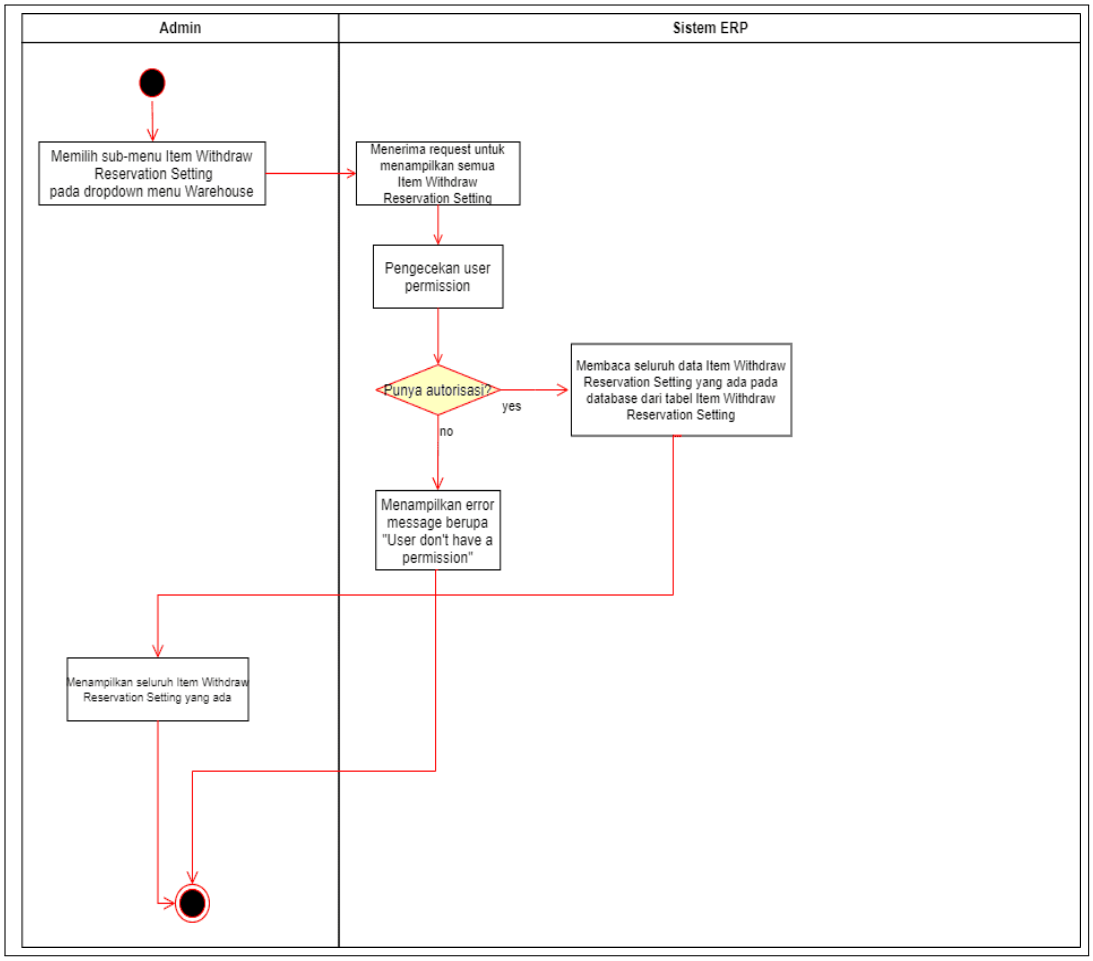


Gambar 3.12. *Activity Diagram Read* User Parameter

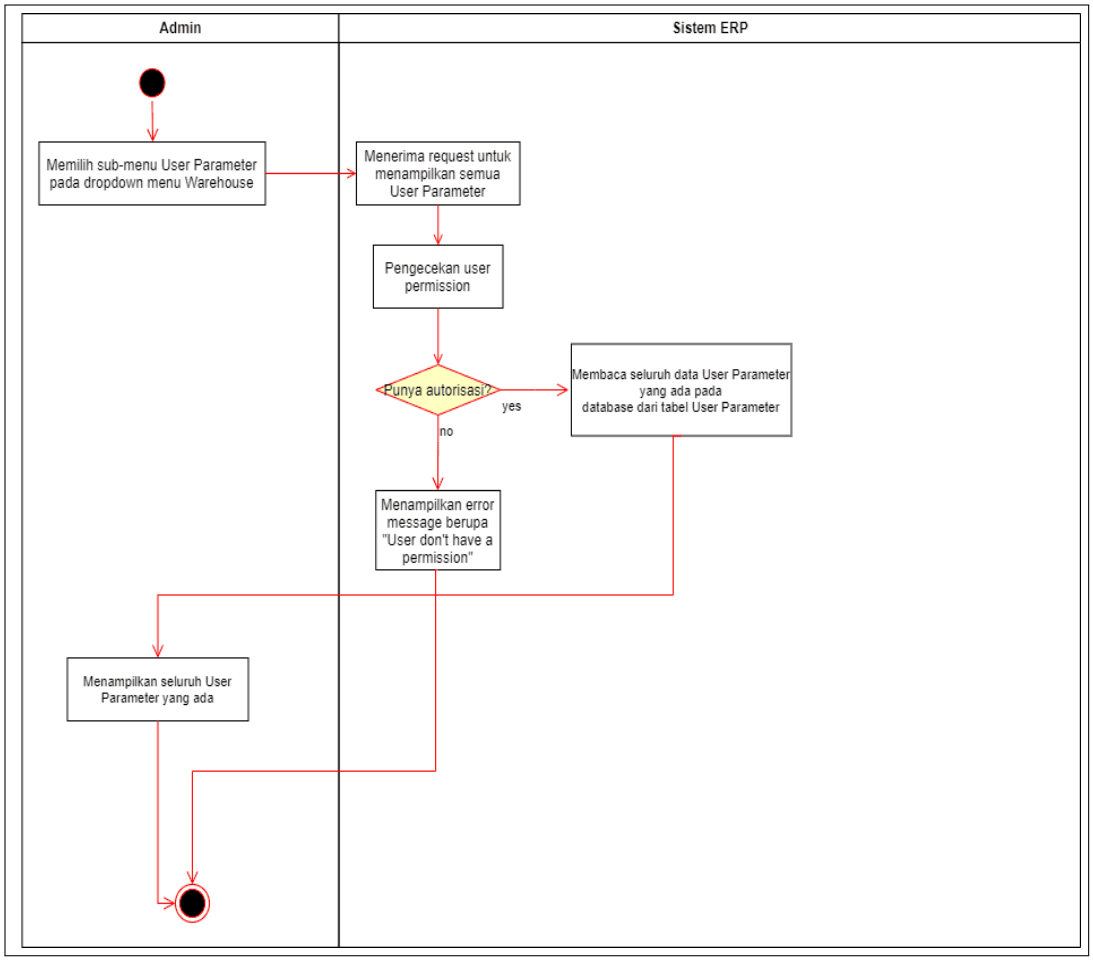
Activity diagram untuk fitur *read* submodul customer type, item withdraw reservation setting, dan user parameter dapat dilihat pada Gambar 3.10, 3.11, dan 3.12. Untuk membaca data pada *item* yang diinginkan, pengguna perlu masuk ke sub-menu di dalam modul Master. Setelah mengakses sub-menu, admin melihat daftar data dari sub-menu yang diinginkan, dan untuk membuka salah satu *item*, admin harus mengklik tombol ikon ”read” pada daftar *item* yang ditampilkan. Sistem memeriksa izin pengguna, dan jika izin ditemukan, informasi mengenai item yang dipilih ditampilkan.



Gambar 3.13. *Activity Diagram Paging* Customer Type

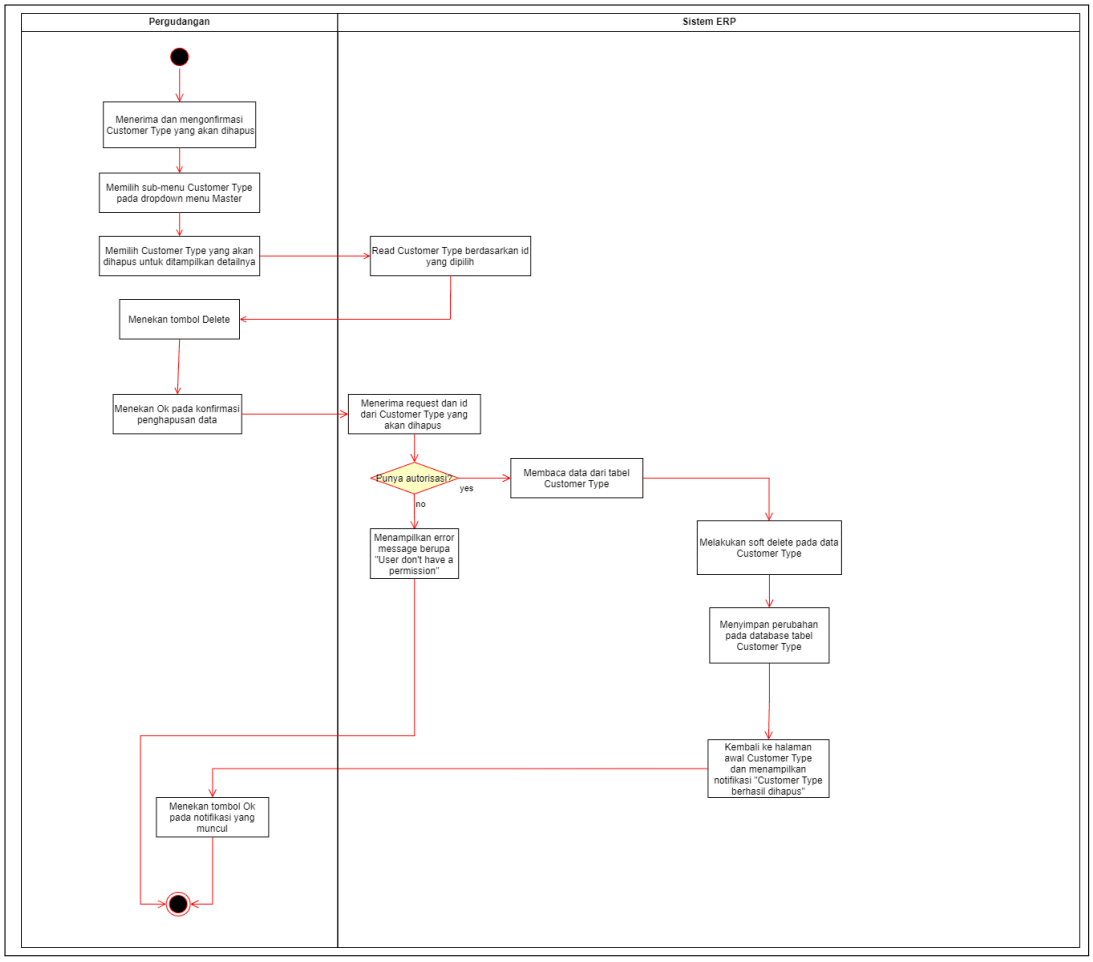


Gambar 3.14. *Activity Diagram Paging* Item Withdraw Reservation Setting

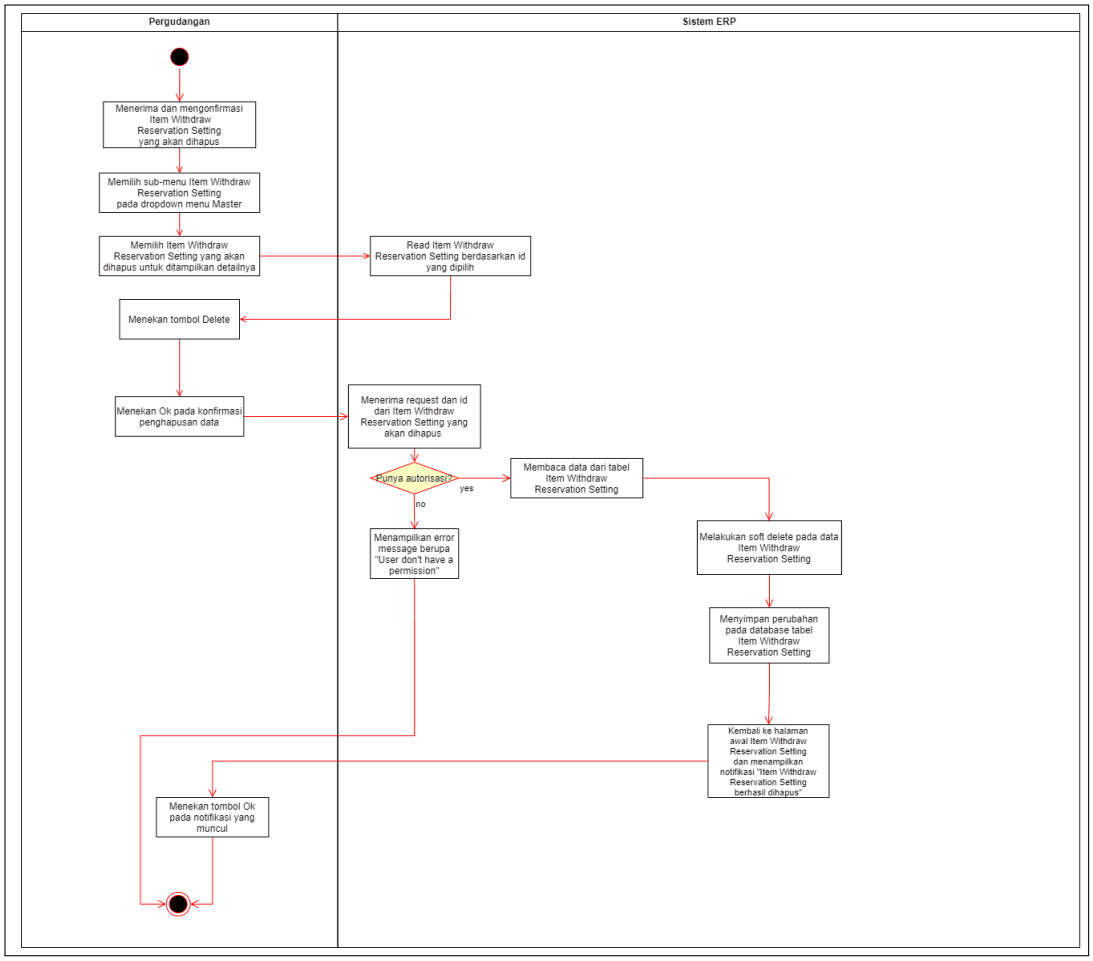


Gambar 3.15. *Activity Diagram Paging* User Parameter

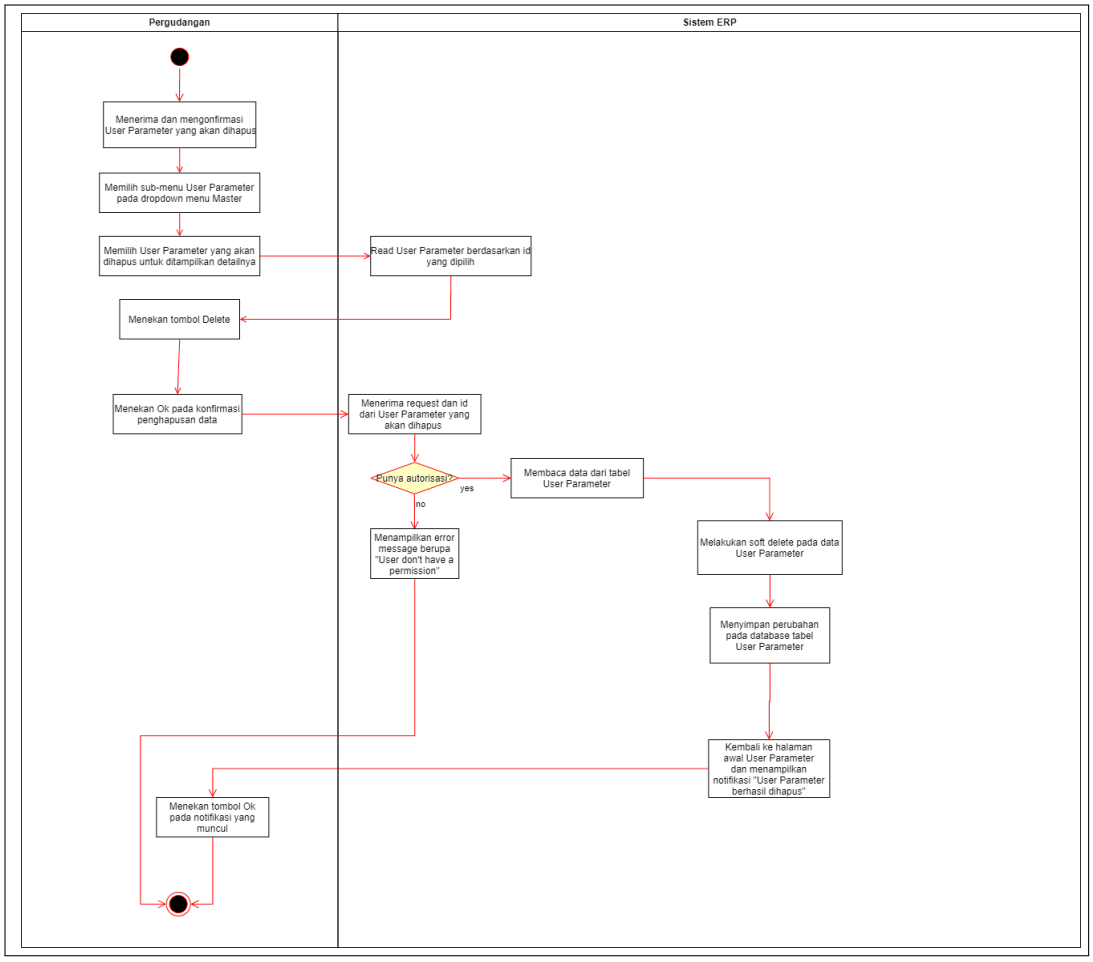
Activity diagram untuk fitur *paging* submodul customer type, item withdraw reservation setting, dan user parameter dapat dilihat pada Gambar 3.13, 3.14, dan 3.15. Mekanisme *paging* secara otomatis aktif ketika administrator masuk ke salah satu sub-menu di dalam modul Master. Setelah mengakses sub-menu, sistem menerima permintaan untuk menampilkan semua data dari sub-menu tersebut dalam format tabel *paging*. Tabel ini ditampilkan jika administrator memiliki izin yang diperlukan untuk mengakses sub-menu yang dipilih.



Gambar 3.16. *Activity Diagram Delete* Customer Type



Gambar 3.17. *Activity Diagram Delete* Item Withdraw Reservation Setting



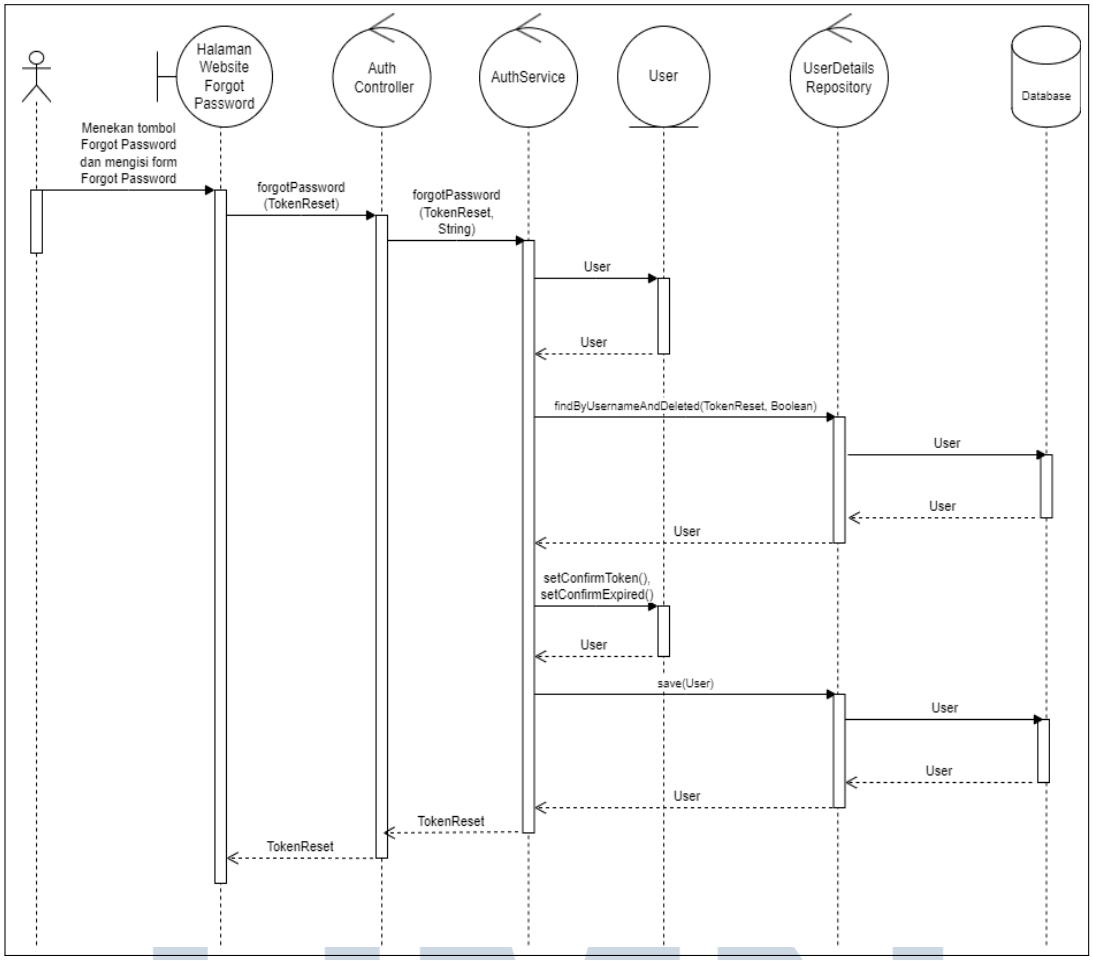
Gambar 3.18. *Activity Diagram Delete* User Parameter

Activity diagram untuk fitur *delete* submodul customer type, item withdraw reservation setting, dan user parameter dapat dilihat pada Gambar 3.16, 3.17, dan 3.18. Ketika menerima permintaan untuk melakukan penghapusan, sistem melakukan verifikasi terhadap keberadaan data pada *item* yang dihapus dan otorisasi yang dimiliki oleh pengguna. Apabila pengguna memiliki otorisasi yang sesuai, maka data dihapus secara lunak (*soft delete*), dan perubahan tersebut disimpan ke dalam *database*.

3.4.3 Sequence Diagram

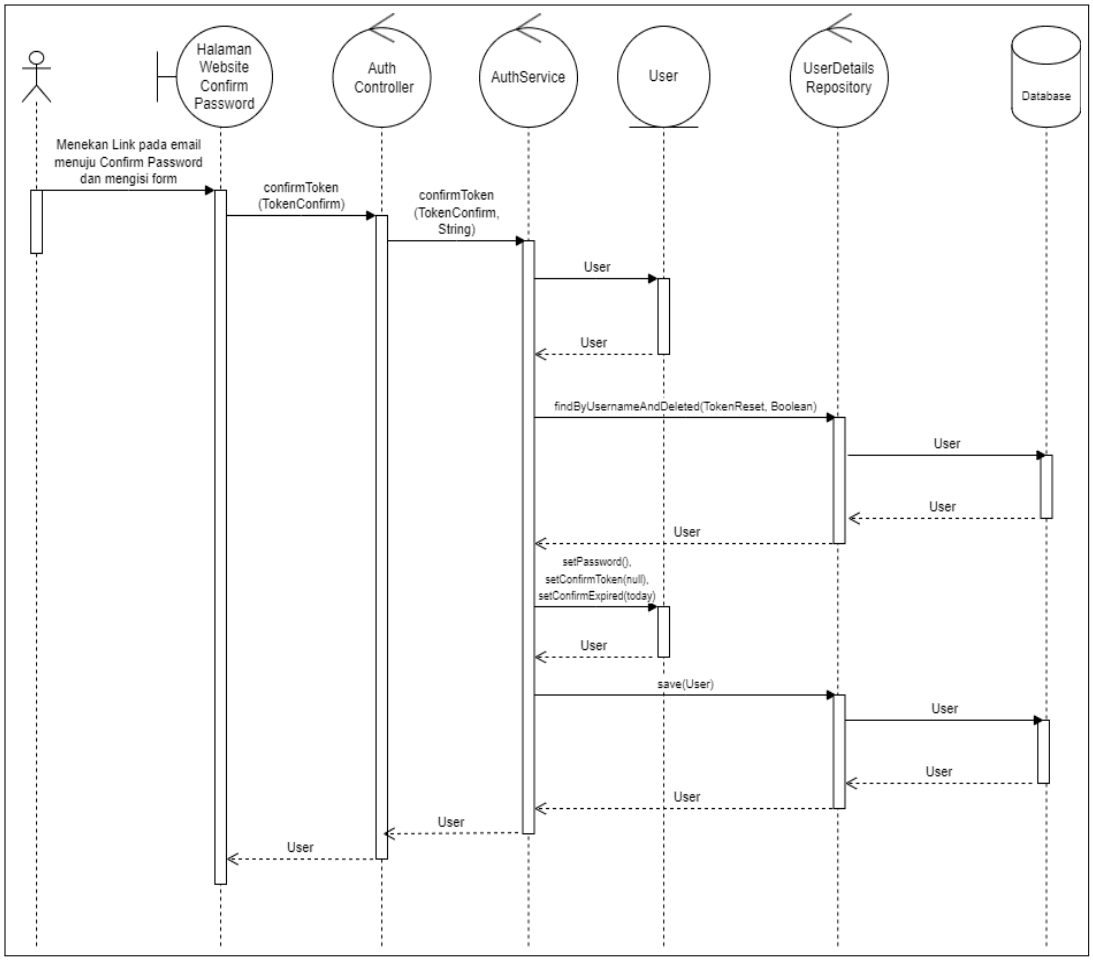
Diagram Urutan dibuat untuk menampilkan komunikasi antar objek dan urutan jalannya informasi dalam suatu sistem. Diagram ini dapat memberikan elemen yang menggambarkan struktur *control flow* dalam *code* pemrograman sistem yang telah dibuat [14]. Penggunaan diagram urutan membantu menyajikan

dengan lebih jelas alur sistem saat menjalankan suatu proses.



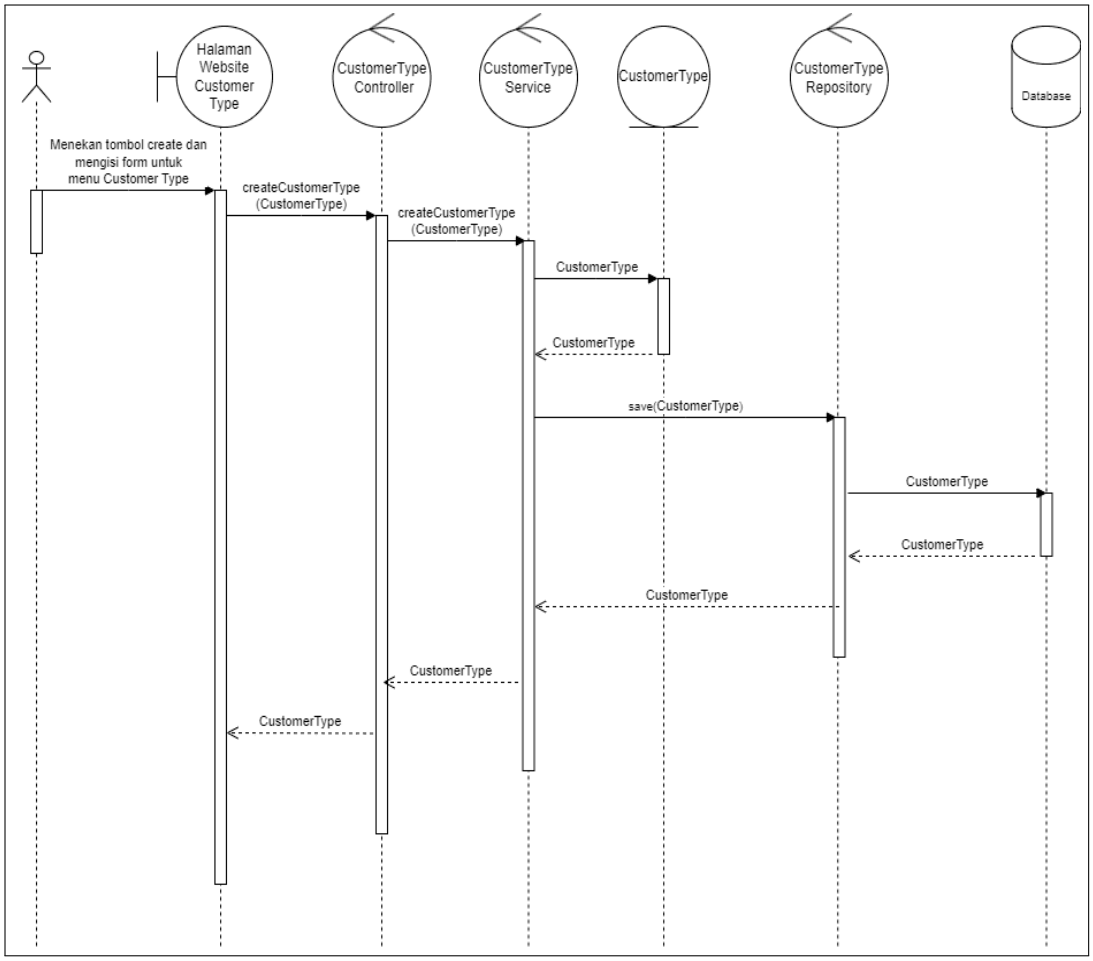
Gambar 3.19. *Sequence Diagram* Forgot Password

Gambar 3.19 merupakan *sequence diagram* untuk forgot password. *Request* untuk forgot password diterima oleh AuthController dengan parameter objek TokenReset yang berisi informasi akun user yang direset beserta token untuk konfirmasi yang nantinya memanggil *method forgotPassword* dari *service*. *Method forgotPassword* pada *service* mendeklarasikan objek User untuk diset datanya. *Service* melakukan *request* ke *repository* untuk melakukan *findByUsernameAndDeleted* dengan parameter TokenReset untuk menemukan data user dari *database*. Setelah itu *service* melakukan *set* untuk ConfirmToken yang digunakan pada confirm password beserta expirednya. Setelah melakukan set data, objek User disimpan ke dalam *database* melalui *repository*.

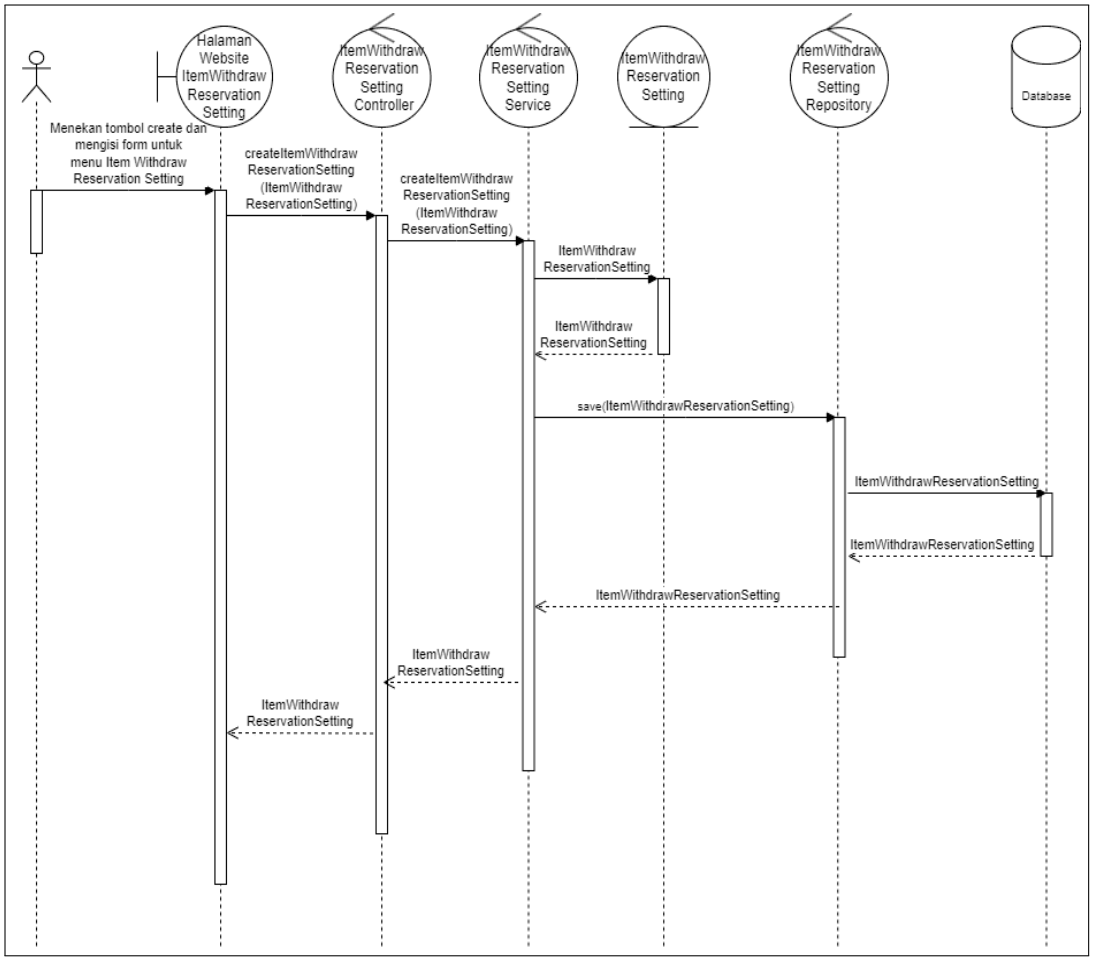


Gambar 3.20. *Sequence Diagram* Confirm Password

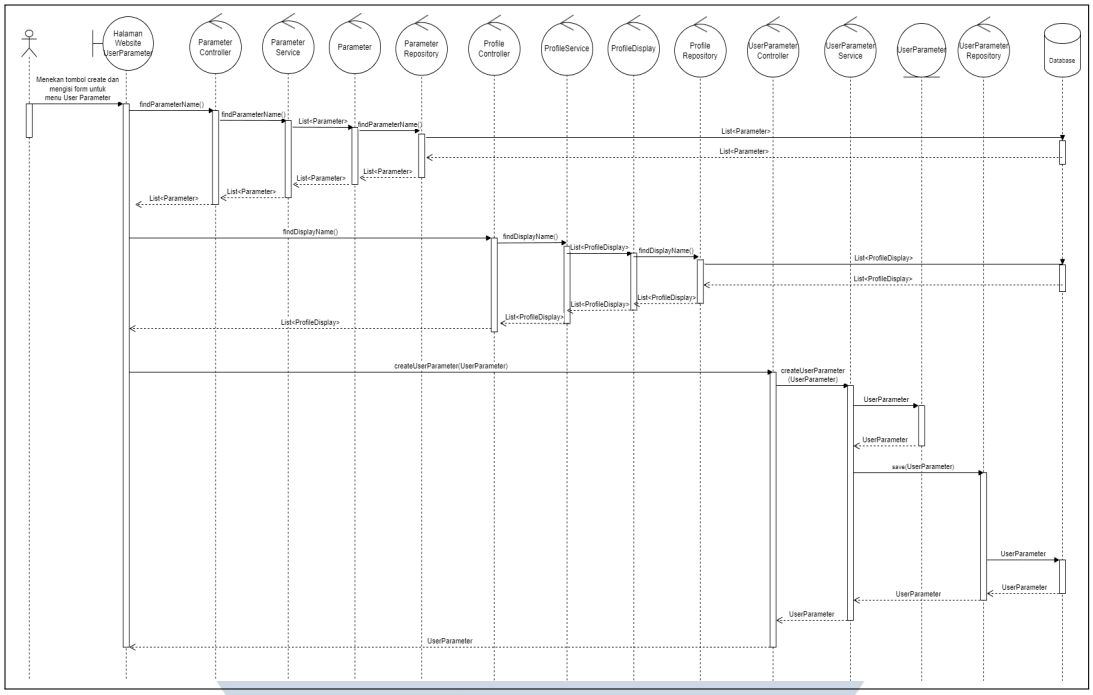
Gambar 3.20 merupakan *sequence diagram* untuk confirm password. *Request* untuk confirm password diterima oleh AuthController dengan parameter objek TokenConfirm yang berisi informasi akun user yang direset beserta token konfirmasi dari forgot password yang nantinya memanggil *method confirmPassword* dari *service*. *Method confirmPassword* pada *service* mendeklarasikan objek User untuk diset datanya. *Service* melakukan *request* ke *repository* untuk melakukan *findByUsernameAndDeleted* dengan parameter TokenConfirm yang dikirimkan untuk menemukan data user dari *database* dengan token confirm yang sesuai. Setelah itu *service* melakukan *set* untuk Password, ConfirmToken menjadi *null* serta expirednya. Setelah melakukan set data, objek User yang sudah diubah passwordnya disimpan ke dalam *database* melalui *repository*.



Gambar 3.21. *Sequence Diagram method Create* Customer Type

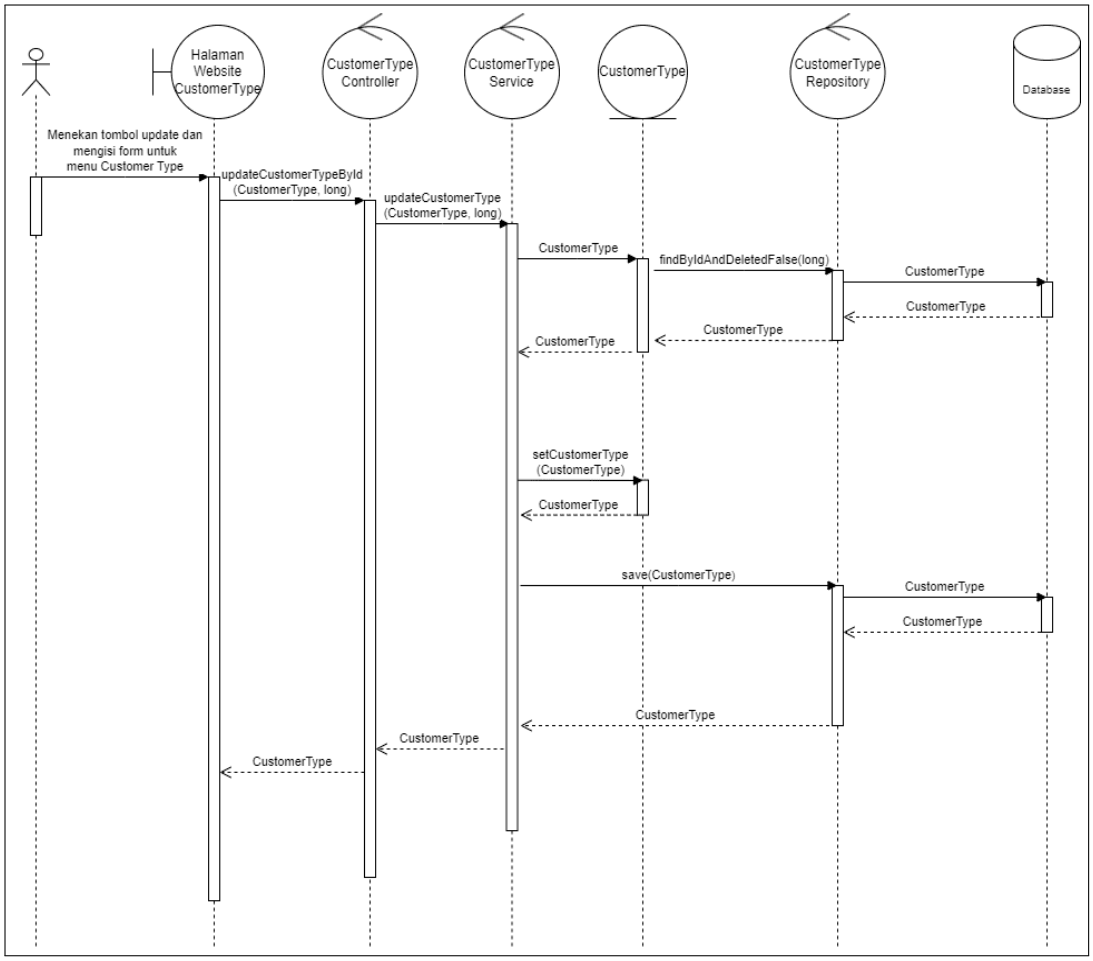


Gambar 3.22. *Sequence Diagram method Create* Item Withdraw Reservation Setting

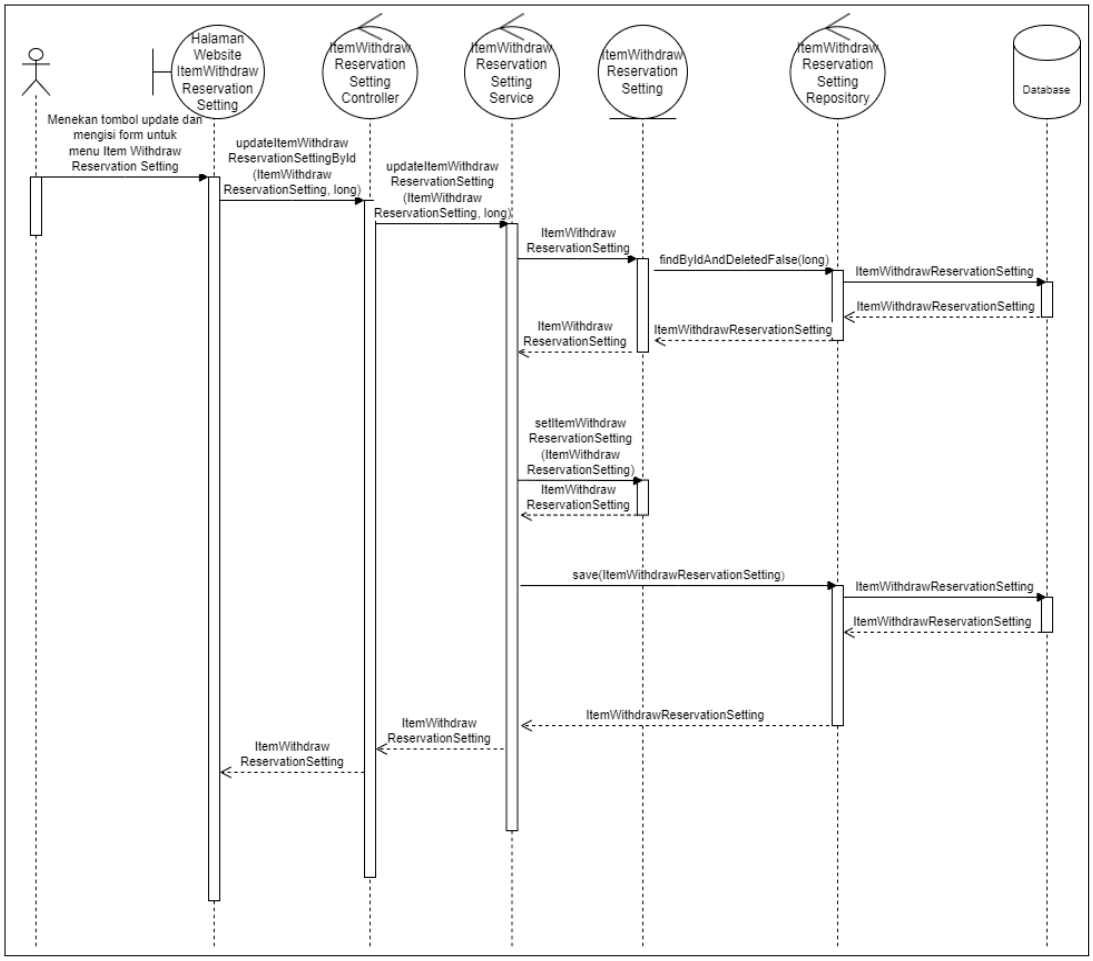


Gambar 3.23. *Sequence Diagram method Create* User Parameter

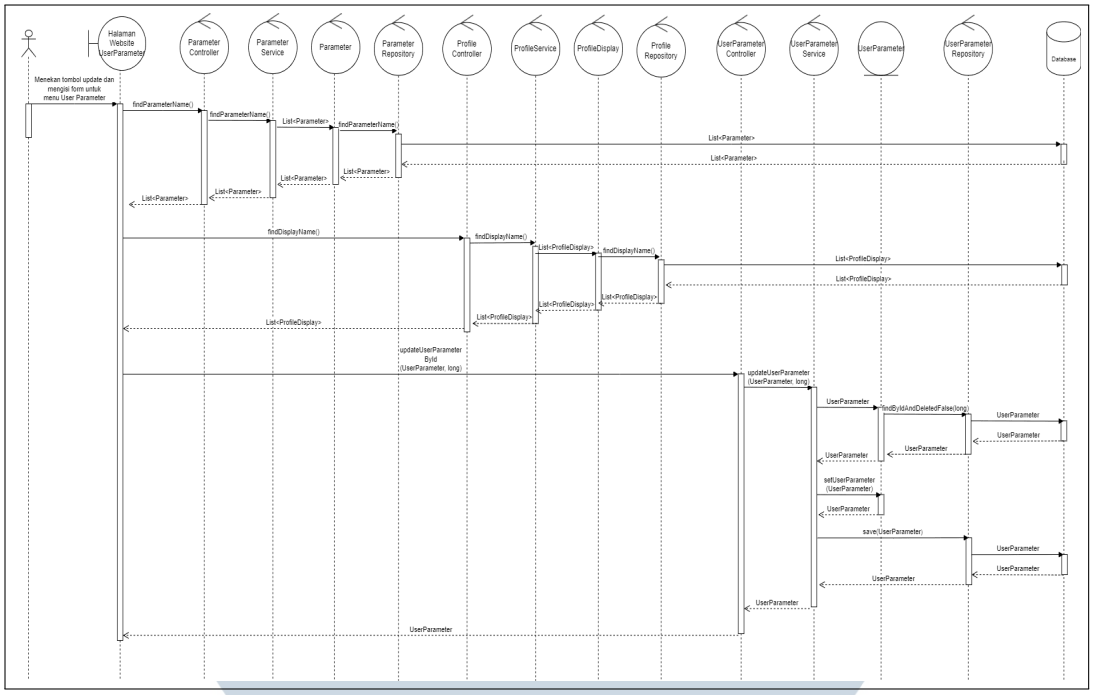
Gambar 3.21, 3.22, 3.23 merupakan *sequence diagram* untuk *method create* pada customer type, item withdraw reservation setting, dan user parameter. Permintaan (*request*) untuk melaksanakan operasi *create* diterima oleh *controller* dengan menggunakan parameter objek data *input*. Setelah itu, *controller* memanggil *method create* dari *service*. *Service* membuat deklarasi objek yang diisi dengan data yang relevan. Untuk sub-modul user parameter melakukan pengecekan terlebih dahulu terhadap data pada sub-modul Parameter dan User melalui profile untuk mendapatkan data sebagai *auto-complete*. Setelah menyelesaikan proses pengaturan data, objek tersebut disimpan ke dalam *database* dengan menggunakan *repository*.



Gambar 3.24. *Sequence Diagram method Update* Customer Type

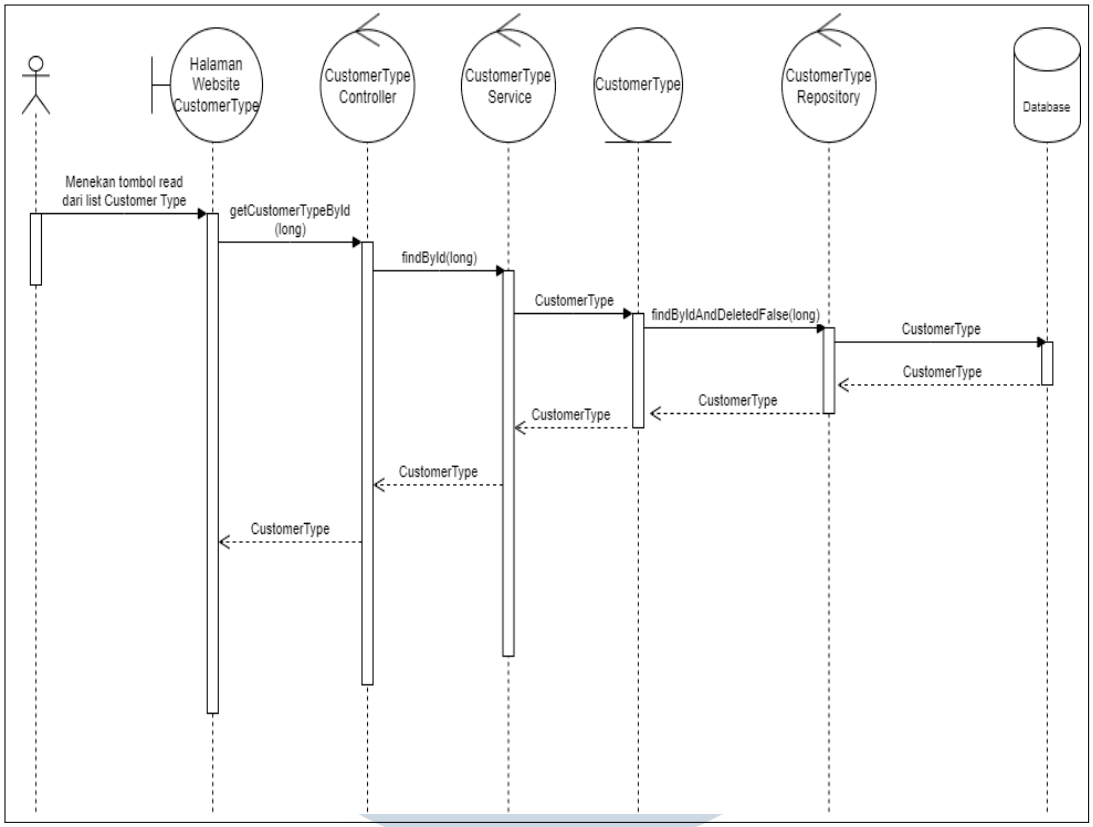


Gambar 3.25. *Sequence Diagram method Update* Item Withdraw Reservation Setting

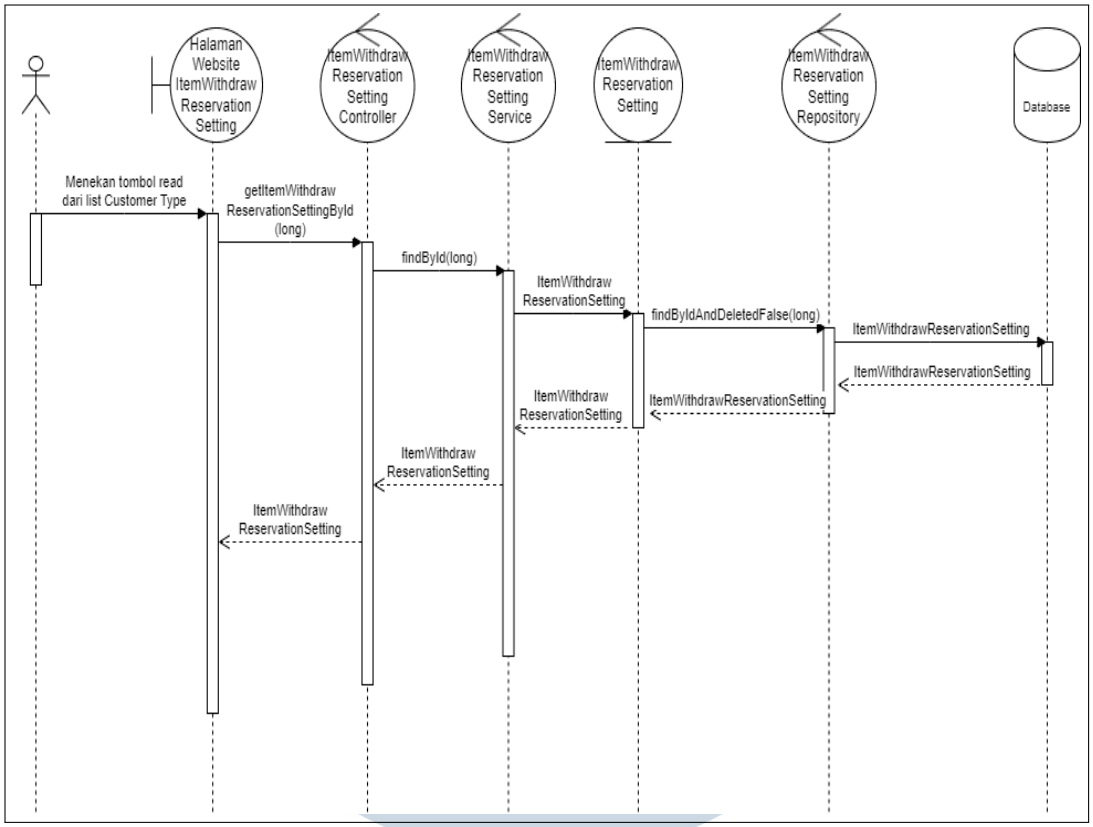


Gambar 3.26. *Sequence Diagram method Update* User Parameter

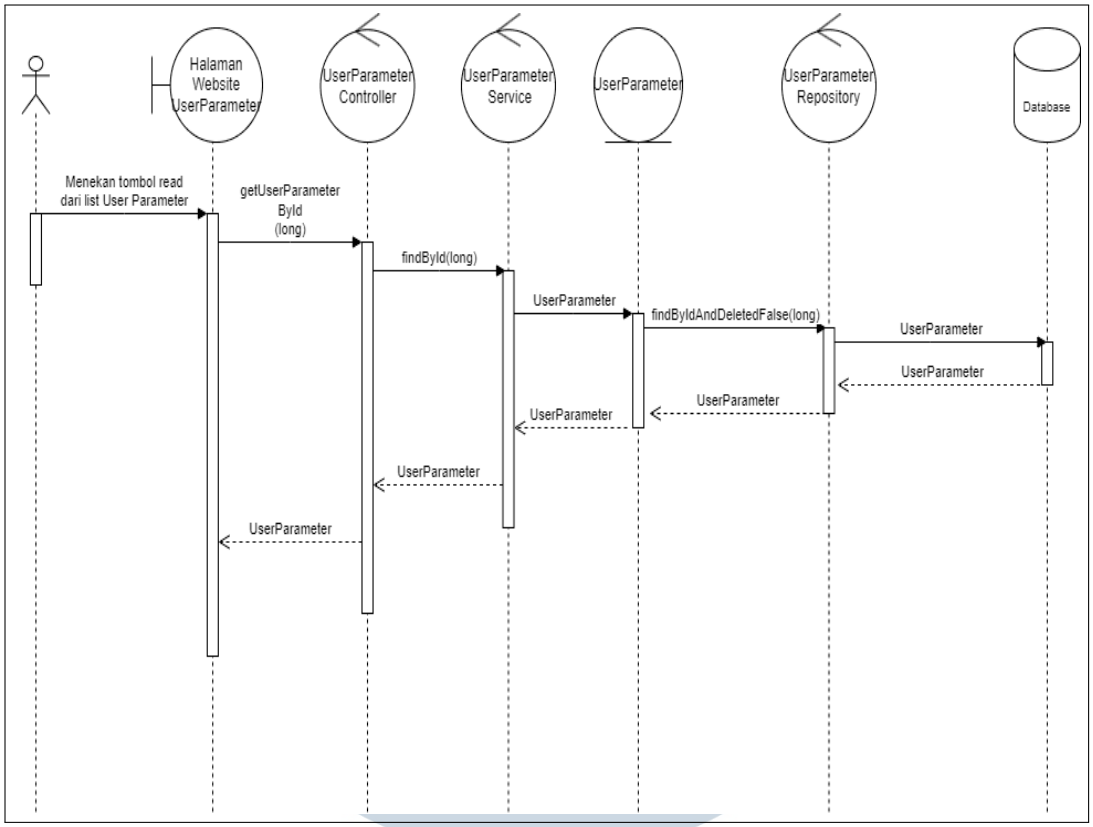
Gambar 3.24, 3.25, 3.26 merupakan *sequence diagram* untuk *method update* pada customer type, item withdraw reservation setting, dan user parameter. *Controller* menerima proses *update*, termasuk parameter objek dan ID data yang dimodifikasi. Selanjutnya, *controller* memanggil *service* untuk menghasilkan deklarasi objek yang dirancang untuk menyimpan data yang diubah dalam database. Sebelum proses penyimpanan yang sebenarnya, *service* melakukan pemeriksaan pada *database* untuk mengonfirmasi ketersediaan data yang dimodifikasi. Data input yang diperuntukkan untuk perubahan kemudian dimasukkan ke dalam objek yang telah disiapkan. Setelah itu, data yang telah dimodifikasi disimpan di dalam basis data dengan menggunakan repositori.



Gambar 3.27. *Sequence Diagram method Read* Customer Type

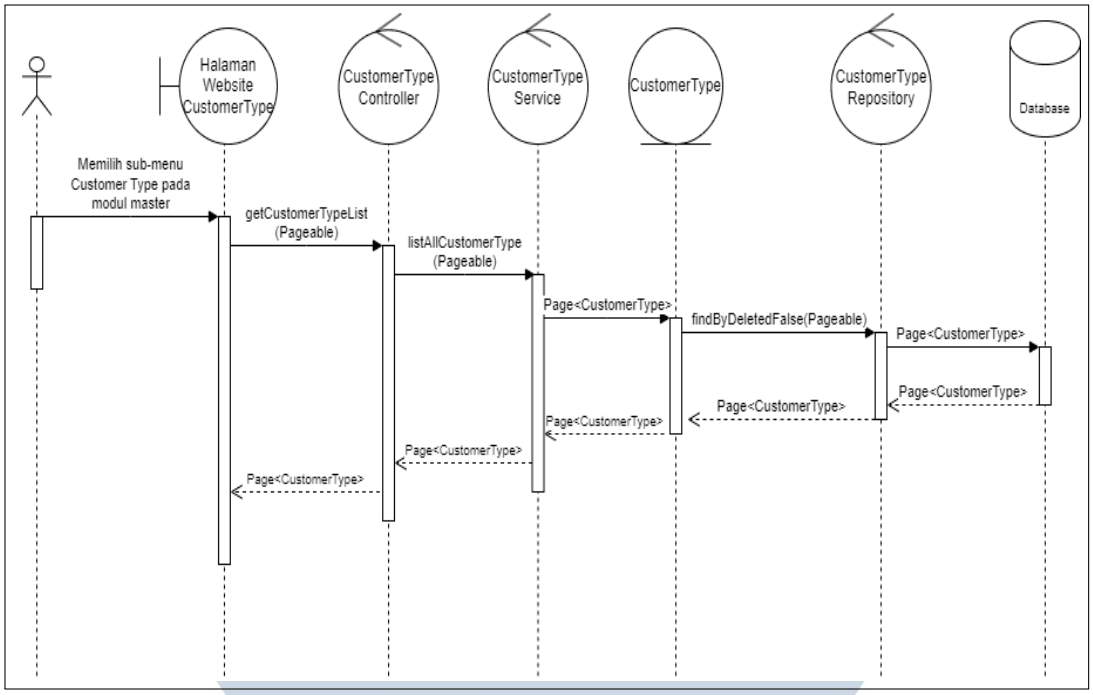


Gambar 3.28. *Sequence Diagram method Read* Item Withdraw Reservation Setting

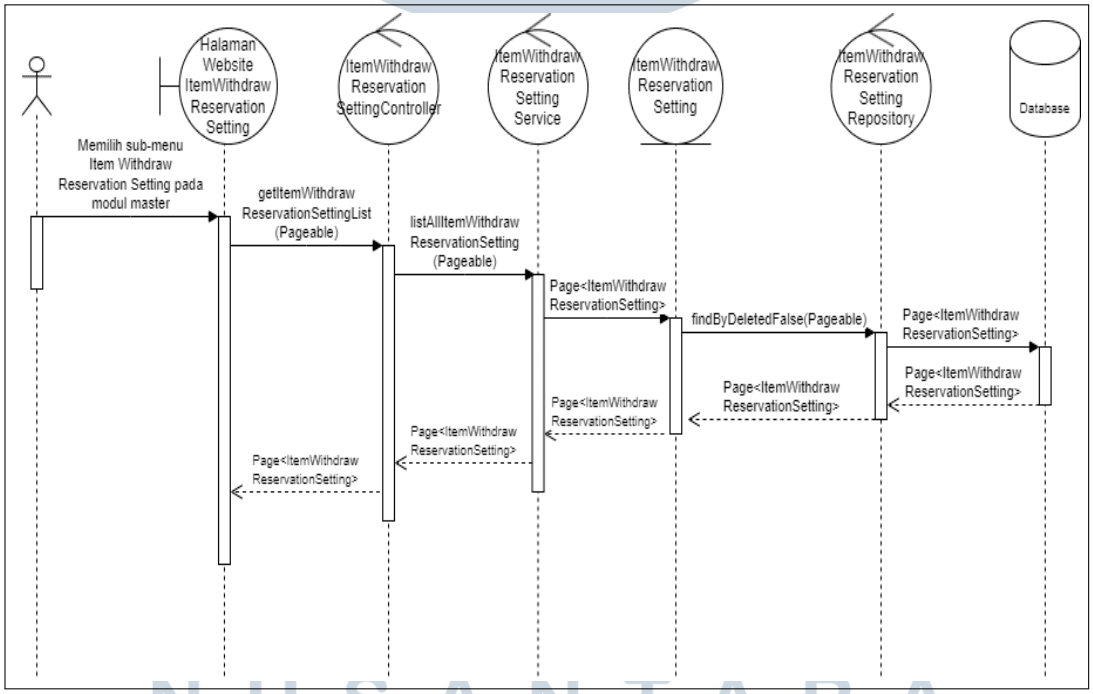


Gambar 3.29. *Sequence Diagram method Read* User Parameter

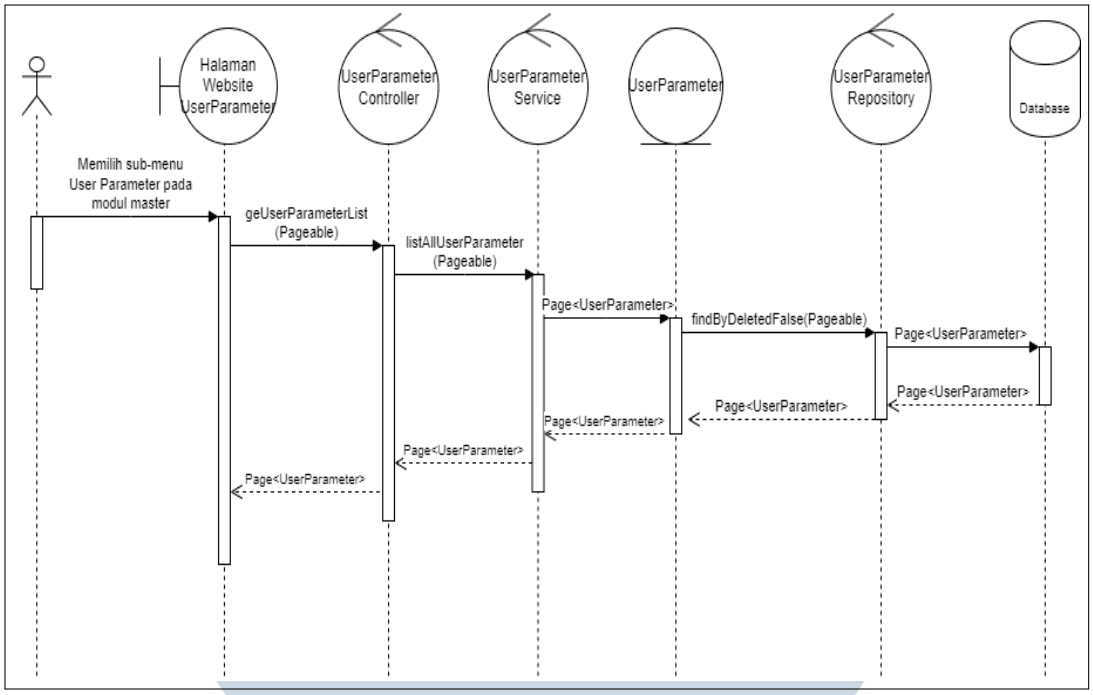
Gambar 3.27, 3.28, 3.29 merupakan *sequence diagram* untuk *method read* pada customer type, item withdraw reservation setting, dan user parameter. *Controller* menerima permintaan untuk operasi baca, termasuk parameter ID data yang diambil. Selanjutnya, *controller* berkomunikasi dengan service untuk menjalankan metode baca. *Service*, pada gilirannya, membuat sebuah objek untuk menampung data yang diambil dari database melalui repositori. Data yang diperoleh dari database disimpan di dalam objek dan kemudian dikirim kembali ke controller untuk ditampilkan kepada pengguna.



Gambar 3.30. *Sequence Diagram method Paging* Customer Type

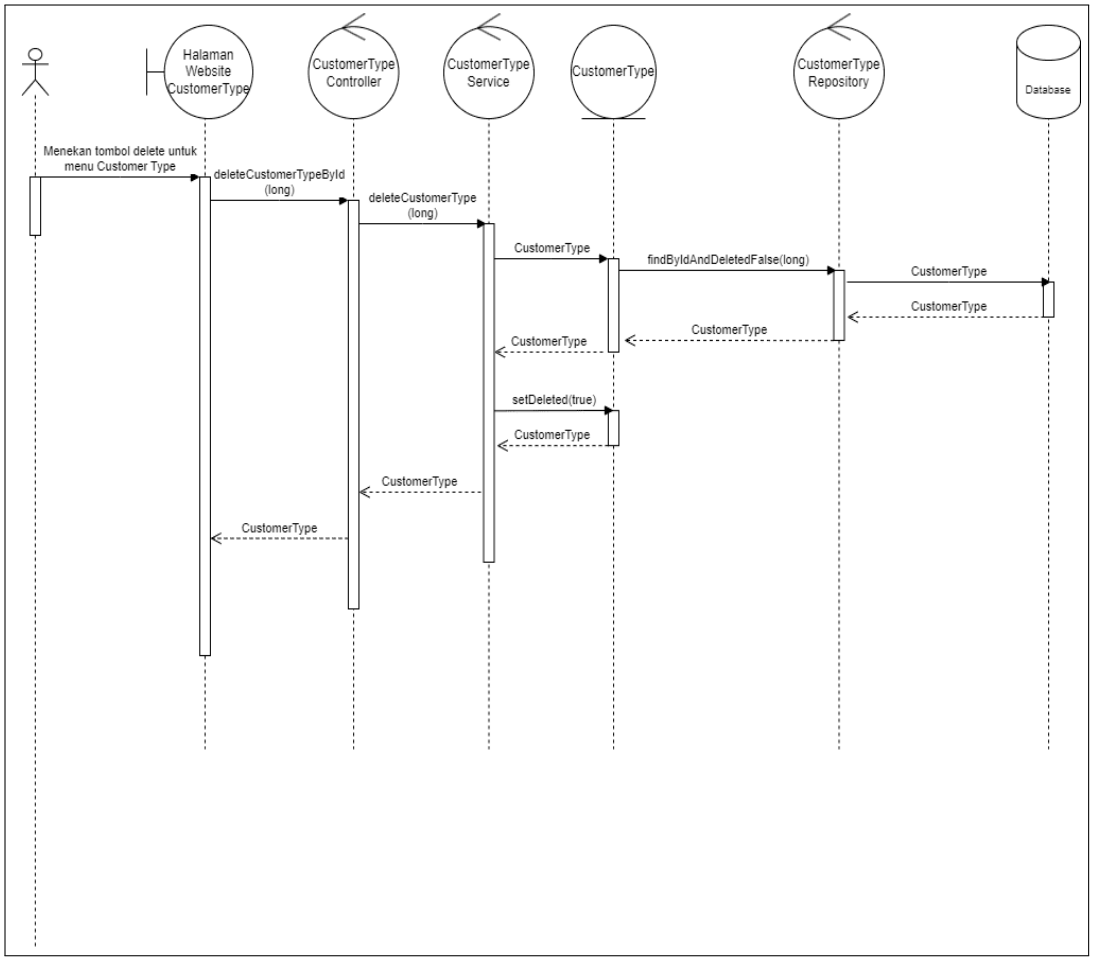


Gambar 3.31. *Sequence Diagram method Paging* Item Withdraw Reservation Setting

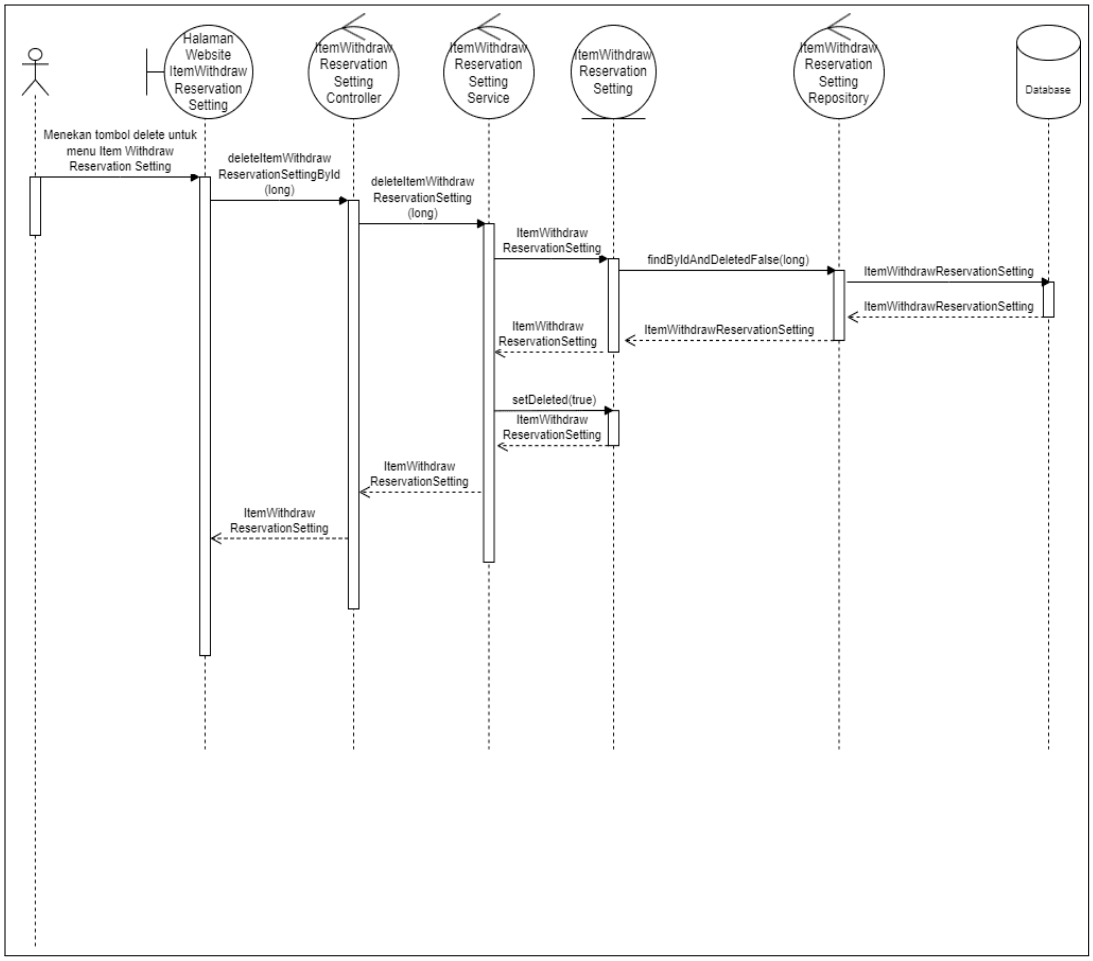


Gambar 3.32. *Sequence Diagram method Paging* User Parameter

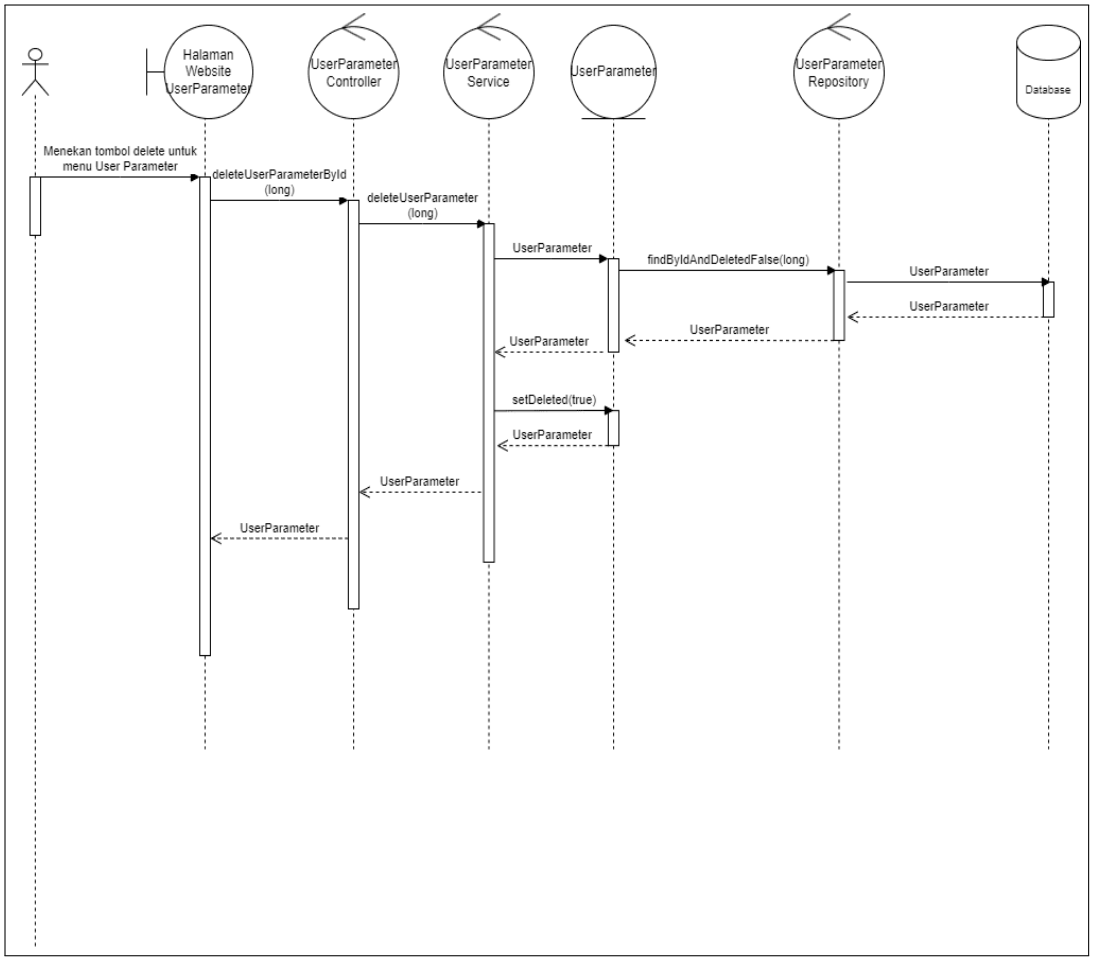
Gambar 3.30, 3.31, 3.32 merupakan *sequence diagram* untuk *method paging* pada customer type, item withdraw reservation setting, dan user parameter. *Controller* menerima permintaan untuk melakukan operasi *paging*, termasuk parameter objek yang dapat di*paging*. *Controller* kemudian menghubungi *service* untuk memulai metode *paging*. *Service*, sebagai tanggapan, menghasilkan deklarasi objek halaman untuk merangkum daftar lengkap data yang diperoleh dari database melalui repositori. *Service* mengembalikan objek yang berisi data yang dapat dipagari kepada pengguna melalui *controller*.



Gambar 3.33. *Sequence Diagram method Delete* Customer Type



Gambar 3.34. *Sequence Diagram method Delete* Item Withdraw Reservation Setting



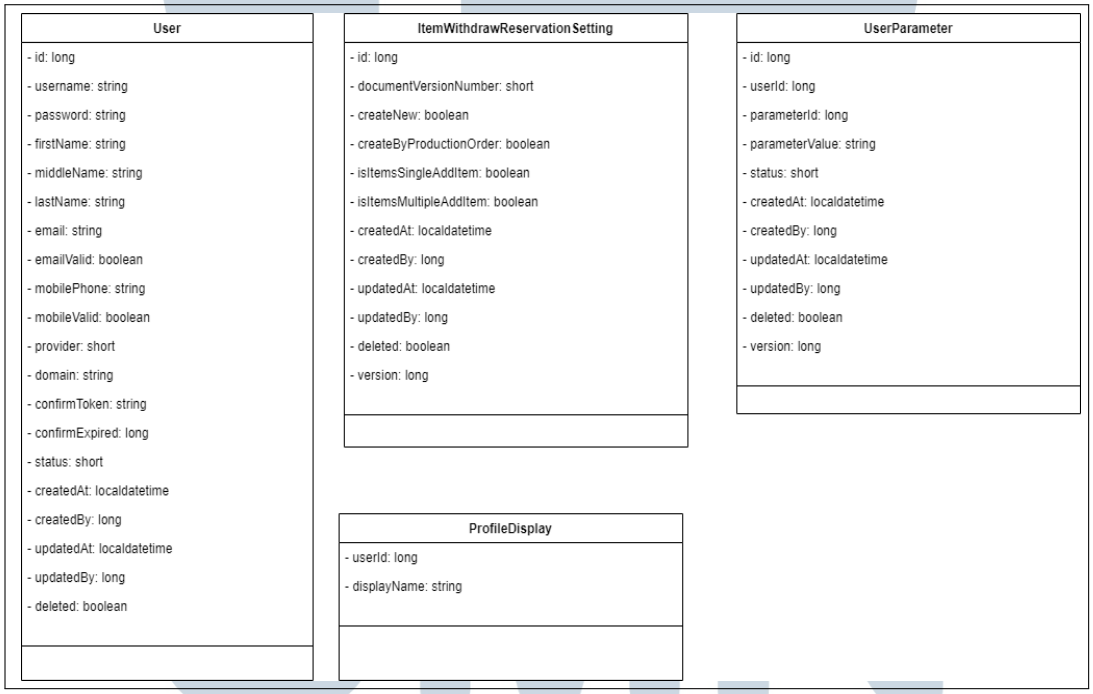
Gambar 3.35. *Sequence Diagram method Delete* User Parameter

Gambar 3.33, 3.34, 3.32 merupakan *sequence diagram* untuk *method delete* pada customer type, item withdraw reservation setting, dan user parameter. *Controller* menerima permintaan penghapusan, termasuk parameter ID data yang dijadwalkan untuk dihapus. Selanjutnya, *controller* memanggil layanan untuk menjalankan metode penghapusan. *Service*, pada gilirannya, membuat deklarasi objek untuk menyimpan data yang dihapus. Sebelum mengeksekusi penghapusan, *service* melakukan pemeriksaan basis data melalui repositori untuk memverifikasi keberadaan data yang ditargetkan untuk dihapus. Proses penghapusan melibatkan pengaturan atribut *deleted* menjadi *true* dan menyimpannya dalam *database* sebagai indikasi bahwa data telah dihapus.

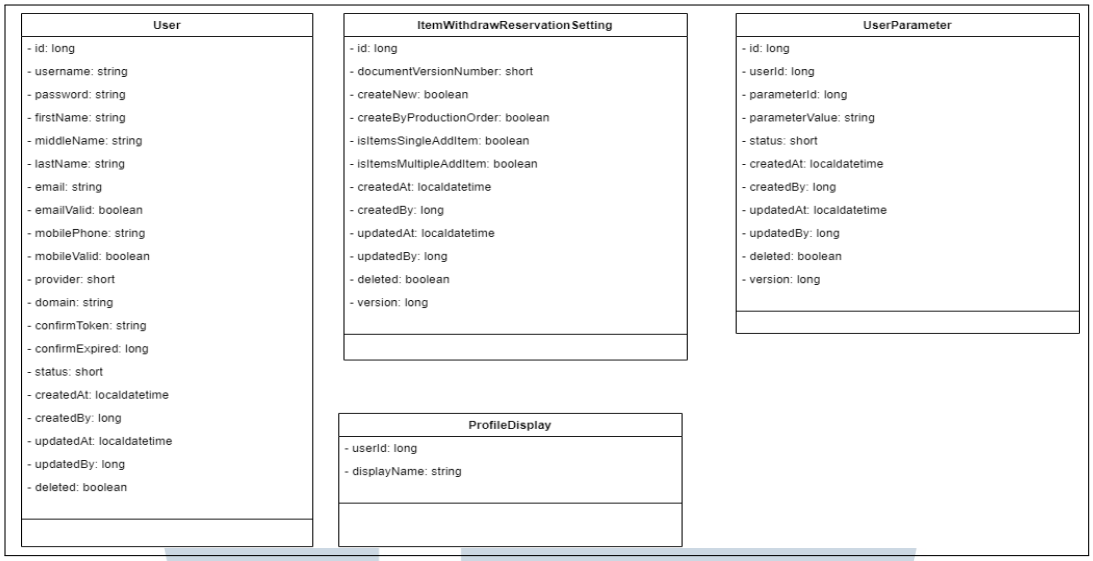
3.4.4 Class Diagram

Diagram Kelas menggambarkan kerangka rancangan sistem yang telah dibuat. Diagram kelas menguraikan entitas-entitas dalam sistem dengan rincian struktur dan hubungan yang terbentuk antar entitas. Fungsionalitas diagram kelas dapat berperan sebagai dokumentasi serta representasi visual yang lebih canggih terkait dengan interaksi antar entitas dalam sistem [15].

Entity sistem ERP memiliki berbagai atribut dengan tipe data yang berbeda-beda. *Class diagram* untuk *Entity* yang digunakan oleh sub-modul customer type, item withdraw reservation setting, user parameter, beserta modul forgot dan confirm password dapat dilihat pada Gambar 3.36, 3.37.



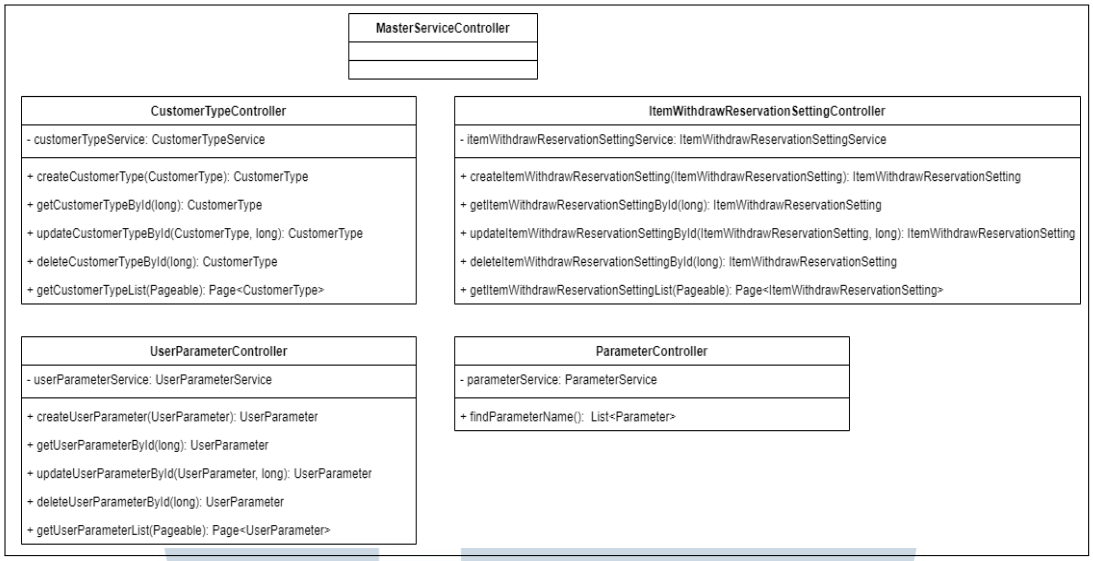
Gambar 3.36. *Class Diagram Entity* User, ItemWithdrawReservationSetting, UserParameter, dan ProfileDisplay



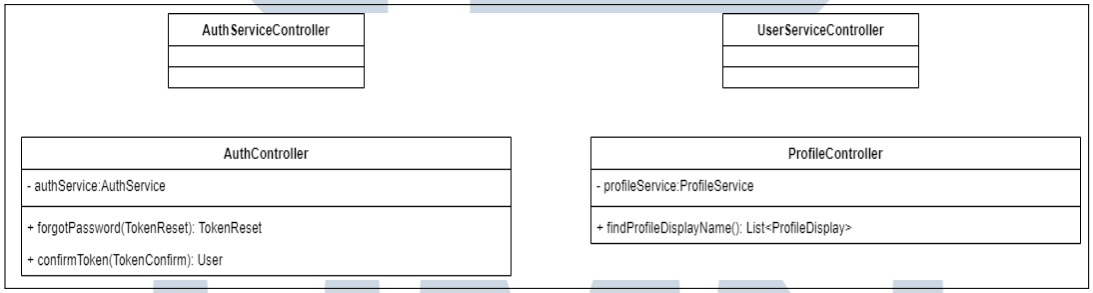
Gambar 3.37. *Class Diagram Entity* Parameter, CustomerType, TokenReset, TokenConfirm

Sistem ERP yang sedang dikembangkan menggunakan *controller* sebagai perantara untuk berinteraksi dengan API, yang mana *request* dikirim melalui *path* yang telah dideklarasikan dengan memanggil *method* dari *service*. Pada tingkat implementasi, *controller* memiliki variabel *service* yang diinisialisasi untuk memungkinkan pemanggilan *class service*. *Controller* dalam sistem ERP menyediakan *method* untuk *create*, *read*, *update*, *delete*, *paging*, dan *authentication* proses forgot dan confirm password.

Class diagram untuk *Controller* pada sub-modul customer type, item withdraw reservation setting, user parameter, beserta modul forgot dan confirm password dapat dilihat pada Gambar 3.38, 3.39.



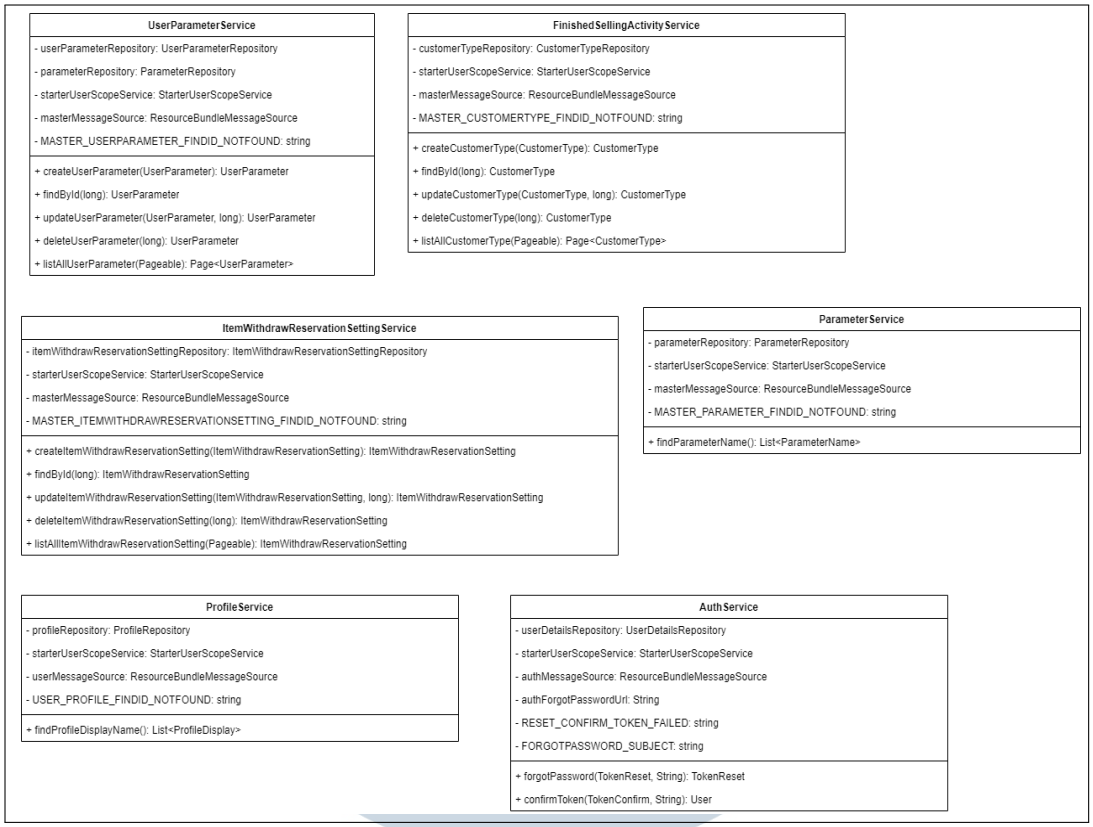
Gambar 3.38. *Class Diagram Controller* CustomerType, ItemWithdrawReservationSetting, UserParameter, dan Parameter



Gambar 3.39. *Class Diagram Controller* Auth dan Profile

Service memiliki peran penting dalam menjalankan *method* yang dipanggil dan mengembalikan hasilnya melalui *controller* kepada pengguna. Dalam *service*, diperlukan deklarasi *repository* untuk melakukan *database query* guna mengakses *database*, *scope* untuk melakukan pengecekan otorisasi pengguna, dan deklarasi *message* untuk menampilkan pesan *error* jika terjadi kegagalan dalam menjalankan *method*. *Service* menyediakan *method* untuk operasi *create*, *read*, *update*, *delete*, *paging*, beserta pengecekan data untuk kepentingan *auto-complete*.

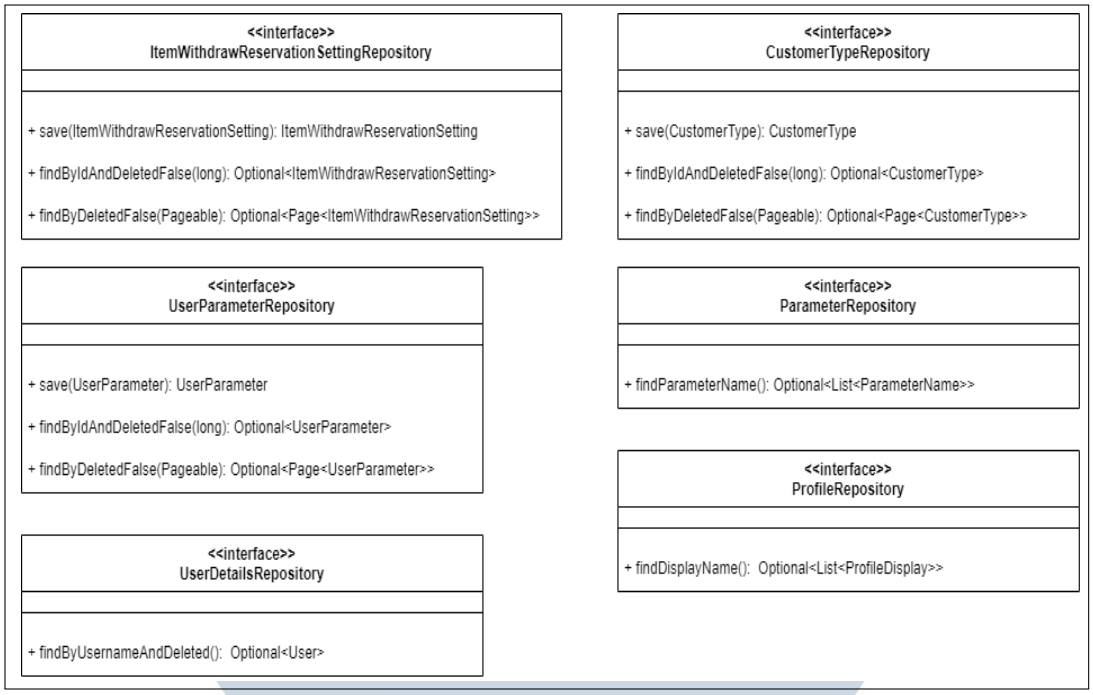
Class diagram untuk *Service* pada sub-modul customer type, item withdraw reservation setting, user parameter, beserta modul forgot dan confirm password dapat dilihat pada Gambar 3.40.



Gambar 3.40. *Class Diagram Service*

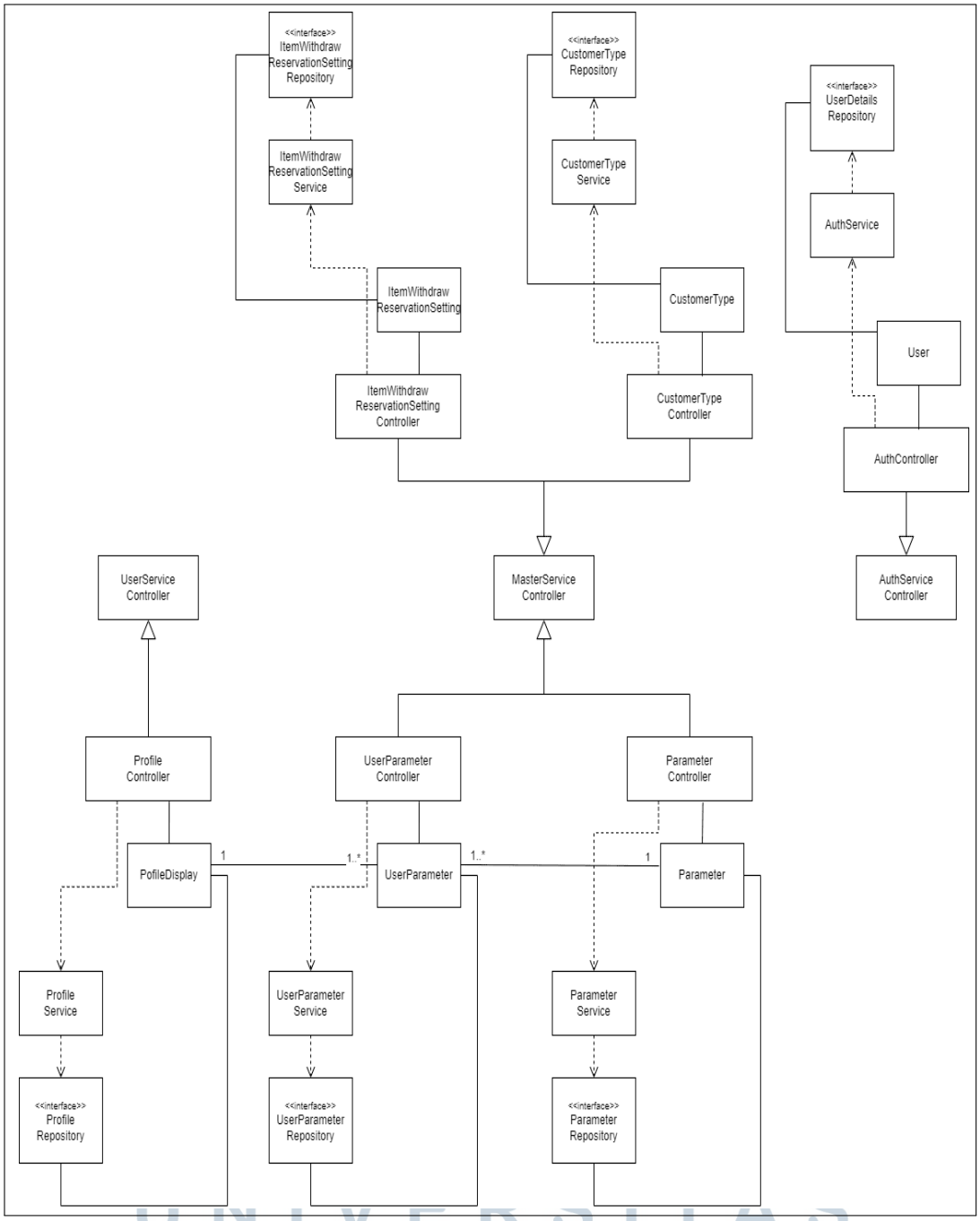
Repository pada sistem ERP digunakan untuk melakukan *database query* pada *database* PostgreSQL. Fungsi utama *repository* adalah untuk menjembatani komunikasi antara sistem dan *database*, yang kemudian dipanggil melalui *service*. *Method* yang terdapat pada *repository* melibatkan *method save* untuk menyimpan objek ke dalam *database*, dan *method find* untuk mencari data dari *database*. *Method find* juga dapat diperluas dengan penerapan aturan tertentu pada *query*, memungkinkan pencarian data dengan menggunakan filter yang telah ditentukan.

Class diagram untuk *Repository* pada sub-modul customer type, item withdraw reservation setting, user parameter, beserta modul forgot dan confirm password dapat dilihat pada Gambar 3.41.



Gambar 3.41. *Class Diagram Repository*

Dari *class diagram* yang telah diuraikan, setiap *class diagram* memiliki keterkaitan (*relationship*) satu sama lain. Hubungan antar *class diagram* dapat terlihat pada Gambar3.42.



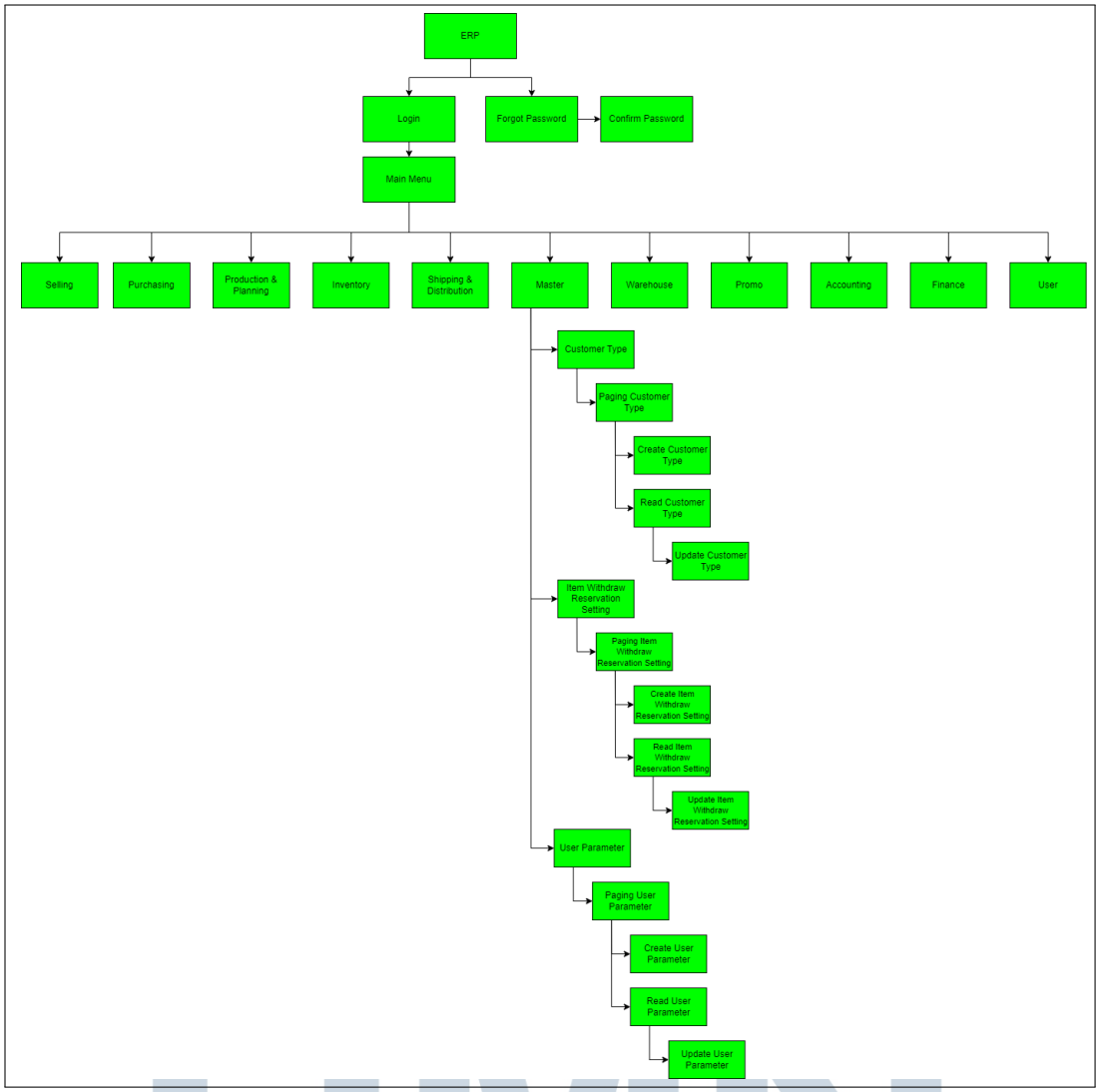
Gambar 3.42. *Relationship Class Diagram*

Pada visualisasi *relationship class diagram* di atas ItemWithdrawReservationSettingController, CustomerTypeController, ParameterController, dan UserParameterController memiliki sifat *inheritance* karena 4 (empat) *controller* tersebut merupakan *subclass* yang melakukan *extend* terhadap MasterServiceController, begitu pula dengan ProfileController

yang melakukan extend terhadap UserServiceController dan AuthController dengan AuthServiceController. *Repository* dengan *Entity* memiliki hubungan berupa *simple association* sedangkan untuk ProfileDisplay dan Parameter dengan UserParameter memiliki hubungan *cardinality one to many*. *Controller* menunjukkan *dependency* terhadap *service*, karena dalam melaksanakan fungsi-fungsi *method*, *controller* memerlukan panggilan *service*. Hal serupa juga terjadi antara *service* dan *repository*.

3.4.5 Sitemap

Sitemap merupakan sebuah diagram yang digunakan untuk menunjukkan struktur sebuah situs. *Sitemap* berperan sebagai peta yang digunakan untuk merefleksikan pemahaman tentang struktur informasi dari situs yang sedang dibangun [16]. Oleh karena itu, untuk menggambarkan keseluruhan navigasi situs sistem ERP yang sudah dibuat sampai saat ini dibuat *sitemap* ERP pada Gambar 3.43.

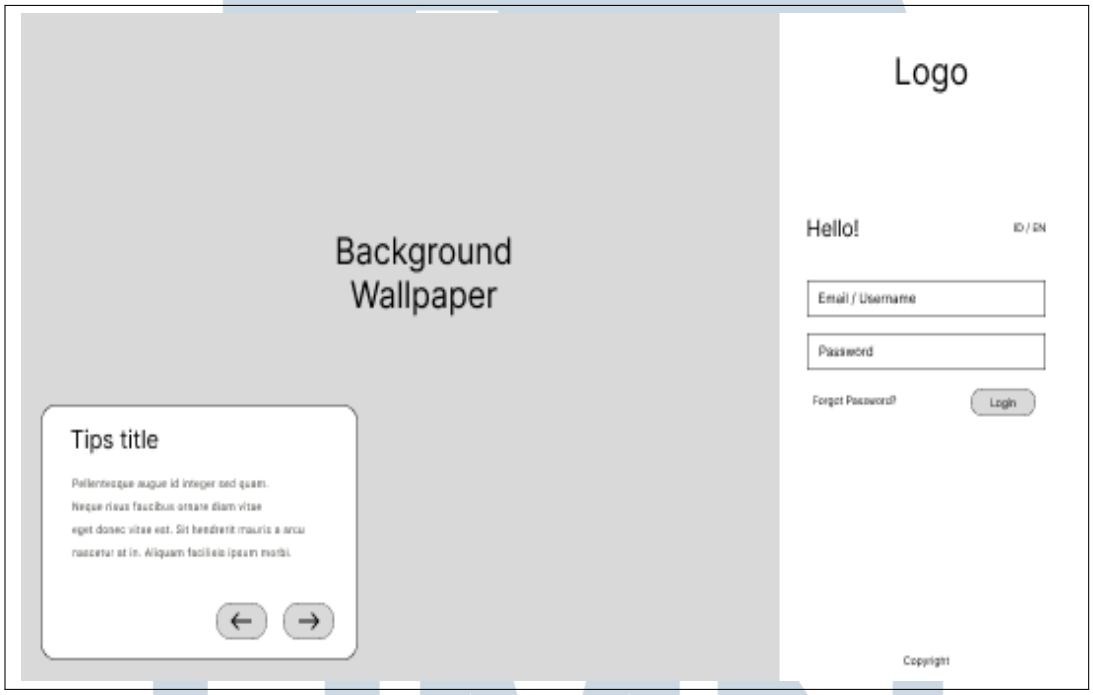


Gambar 3.43. *Sitemap* ERP

Berdasarkan gambar di atas, pengguna dapat mengakses sistem ERP melalui menu login terlebih dahulu, atau pengguna dapat menuju ke forgot password jika ingin mengganti password dari akun yang dimiliki. Setelah pengguna melakukan login, pengguna masuk ke main menu dengan berbagai pilihan modul menu yang dapat dipilih sesuai kebutuhan. Untuk akses modul master, disediakan sub-modul customer type, item withdraw reservation settings, dan user parameter dengan page *create*, *read*, *paging*, dan *update*.

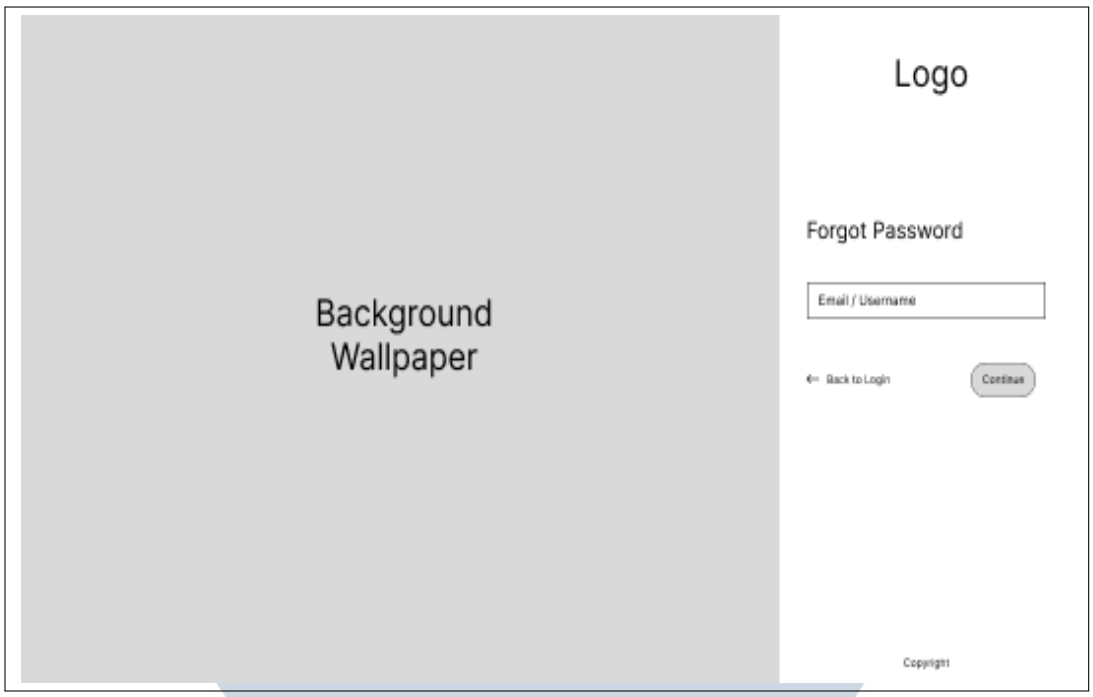
3.4.6 Wireframe

Wireframe salah satu diagram untuk melakukan desain situs *web* pada level struktural yang berupa tata letak halaman *web* yang menunjukkan elemen antarmuka apa yang ada pada halaman utama [17]. Berikut merupakan wireframe untuk website ERP sub-modul Customer Type, Item Withdraw Reservation Setting, User Parameter, Forgot Password, dan Confirm Password.



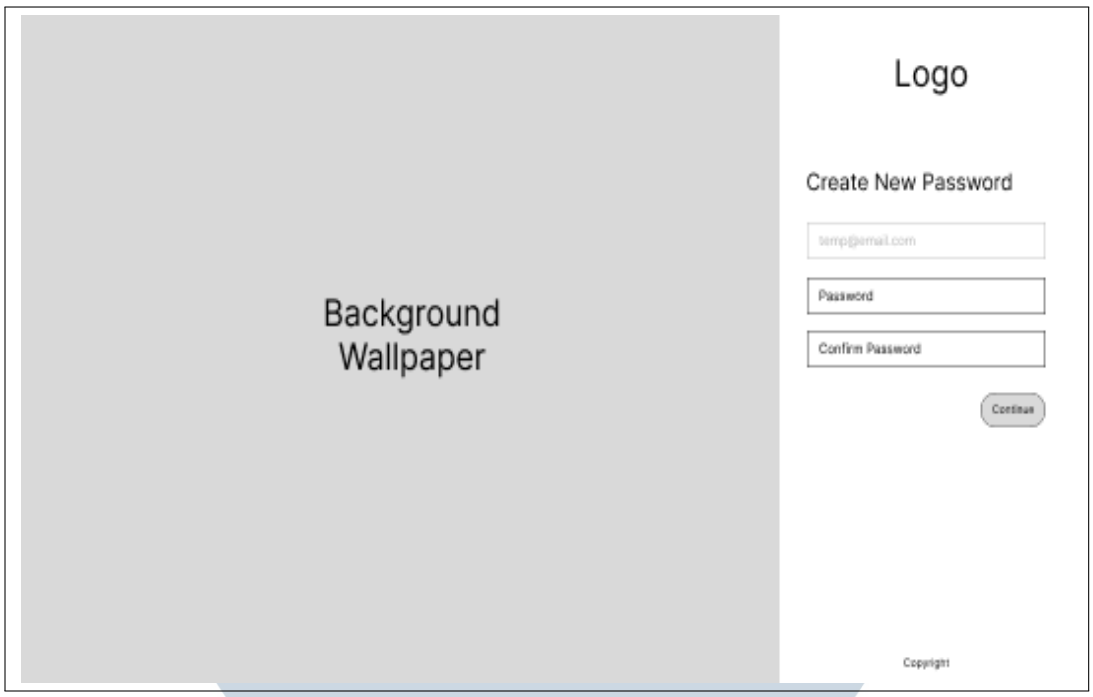
Gambar 3.44. *Wireframe* Login

Pada Gambar 3.44 merupakan wireframe dari halaman Login. Halaman login merupakan halaman bagi user untuk masuk ke halaman menu utama sistem ERP. Halaman login juga tempat di mana user dapat mengakses halaman Forgot Password. Halaman login memiliki background wallpaper dan tooltips untuk memudahkan user untuk login. Pada sisi kanan halaman login, terdapat logo perusahaan, tombol untuk mengganti bahasa, form login berupa email dan password, serta button forgot password dan login.



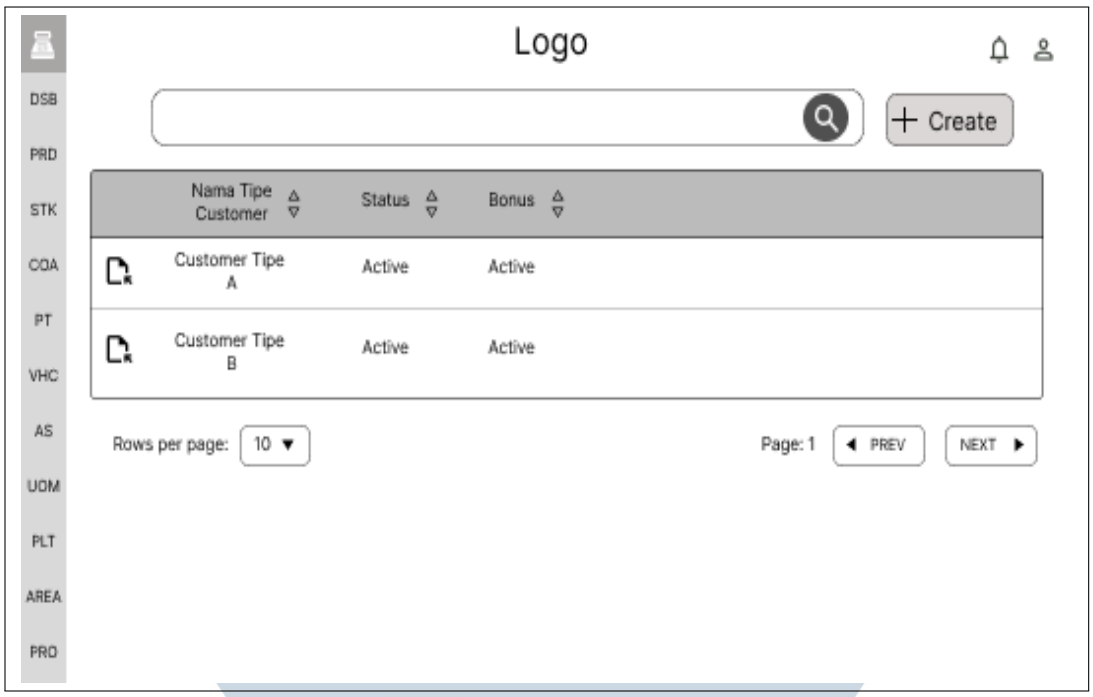
Gambar 3.45. *Wireframe* Forgot Password

Pada Gambar 3.45 merupakan wireframe dari halaman Forgot Password. Halaman forgot password merupakan halaman bagi user untuk melakukan reset password. Halaman forgot password memiliki struktur yang mirip dengan halaman login tetapi tidak memiliki tooltips. Form pada halaman forgot password berupa email dan button untuk kembali ke halaman login dan button continue.



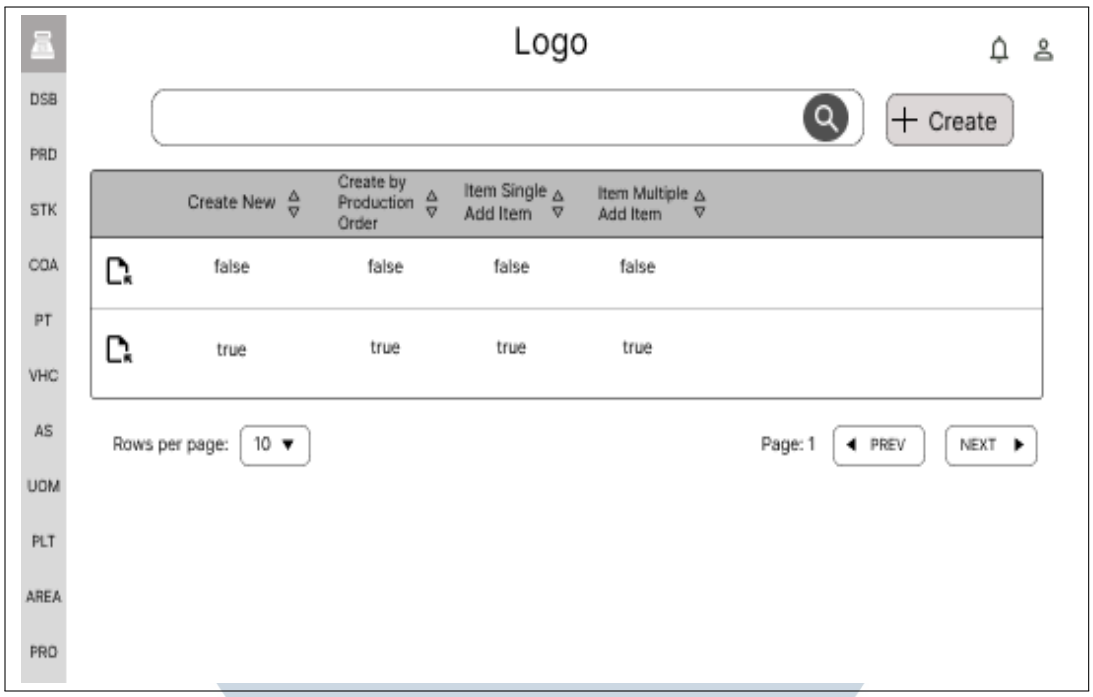
Gambar 3.46. *Wireframe* Confirm Password

Pada Gambar 3.46 merupakan wireframe dari halaman Confirm Password. Halaman confirm password merupakan halaman bagi user untuk melakukan konfirmasi reset password. User dapat mengakses halaman ini dari tautan yang dikirimkan pada email yang user masukkan. Halaman confirm password memiliki struktur yang mirip dengan halaman login tetapi tidak memiliki tooltips. Form pada halaman forgot password berupa email user yang tidak dapat diedit, password, konfirmasi password, dan button continue.



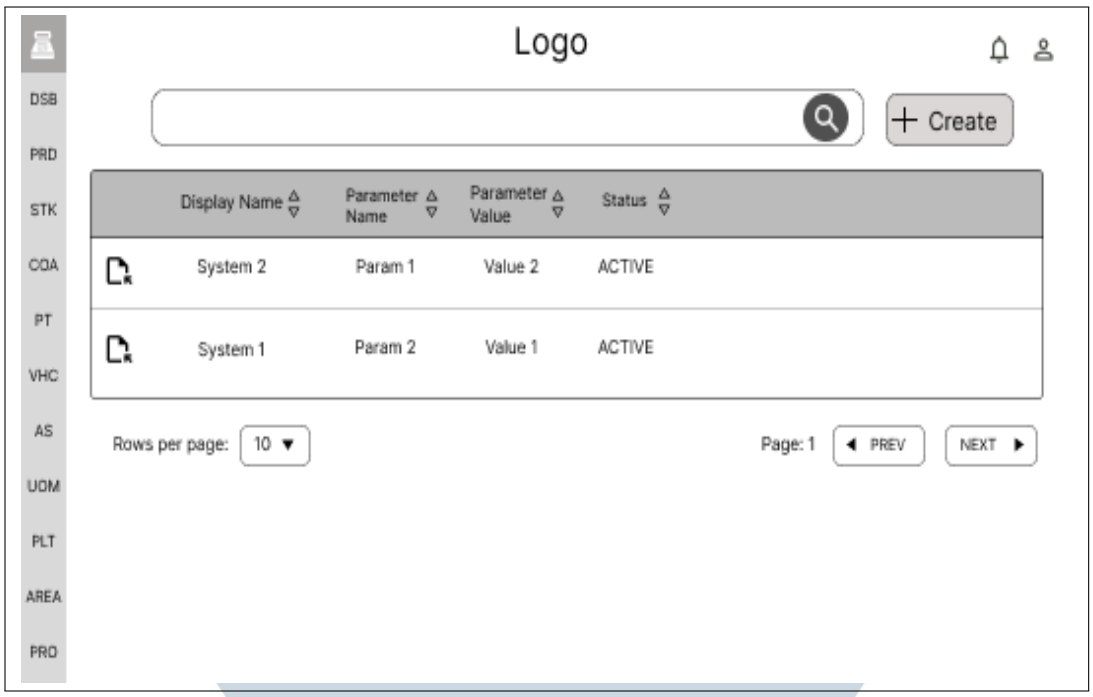
Gambar 3.47. *Wireframe Paging* Customer Type

Pada Gambar 3.47 merupakan *wireframe* dari halaman *Paging* Customer Type. Halaman *paging* customer type menampilkan seluruh data dari *database* pada sebuah *paging table*. Pada halaman paging terdapat button *view* dan button *create*.



Gambar 3.48. *Wireframe Paging* Item Withdraw Reservation Setting

Pada Gambar 3.48 merupakan *wireframe* dari halaman *Paging* Item Withdraw Reservation Setting. Halaman *paging* Item Withdraw Reservation Setting menampilkan seluruh data dari *database* pada sebuah *paging table*. Pada halaman paging terdapat button *view* dan button *create*.

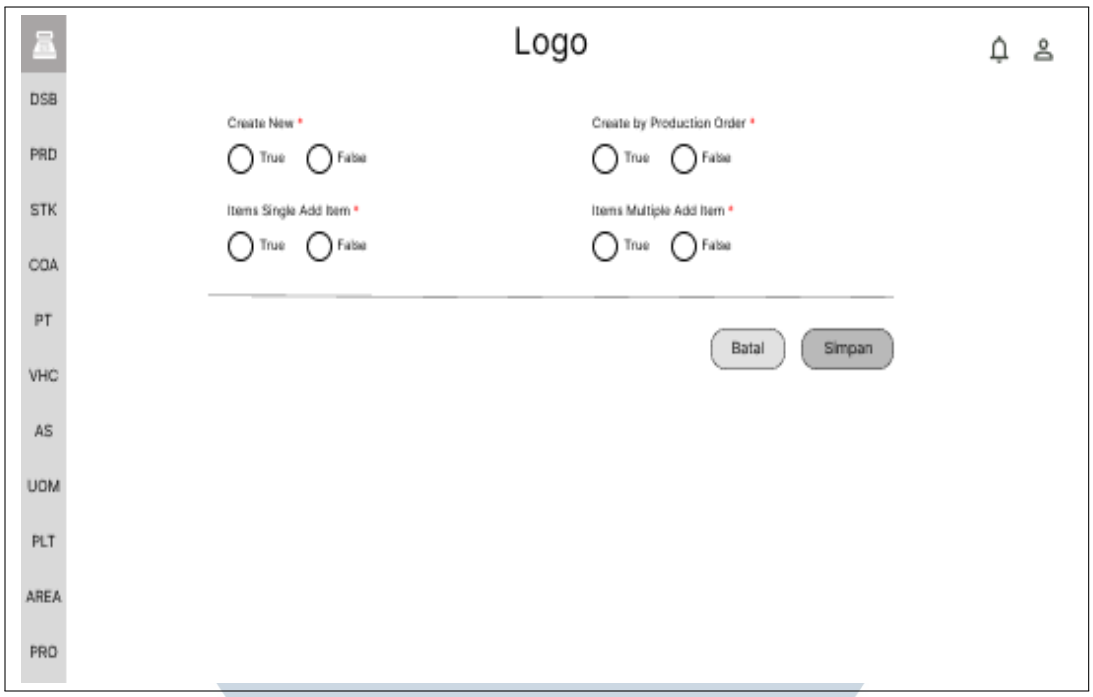


Gambar 3.49. *Wireframe Paging* User Parameter

Pada Gambar 3.49 merupakan *wireframe* dari halaman *Paging* User Parameter. Halaman *paging* user parameter menampilkan seluruh data dari *database* pada sebuah *paging table*. Pada halaman paging terdapat button *view* dan button *create*.

Gambar 3.50. *Wireframe Create* Customer Type

Pada Gambar 3.50 merupakan *wireframe* dari halaman *Create* Customer Type. Halaman *create* customer type menampilkan form untuk melakukan *create* dengan 3 (tiga) *field* yang terdiri dari sebuah *text field* dan 2 (dua) *dropdown*. Pada halaman create terdapat button batal dan simpan.

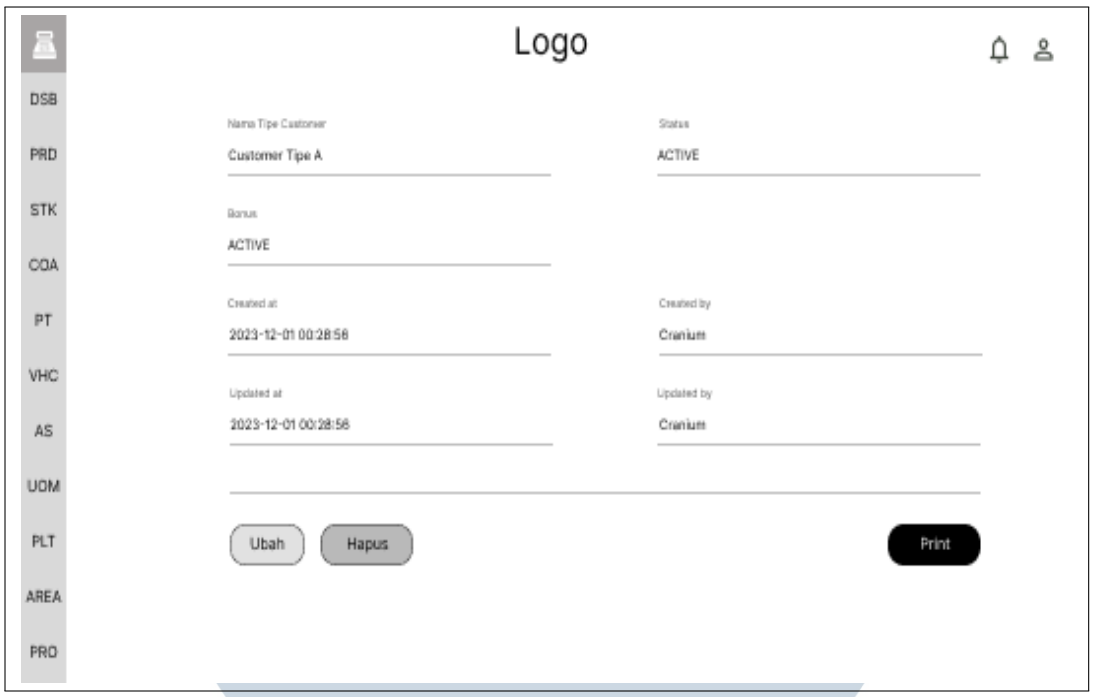


Gambar 3.51. *Wireframe Create* Item Withdraw Reservation Setting

Pada Gambar 3.51 merupakan *wireframe* dari halaman *Create* Item Withdraw Reservation Setting. Halaman *create* Item Withdraw Reservation Setting menampilkan form untuk melakukan *create* dengan 4 (empat) *input field* yang terdiri dari 4 (empat) *radio button* dengan *value true* dan *false*. Pada halaman create terdapat button batal dan simpan.

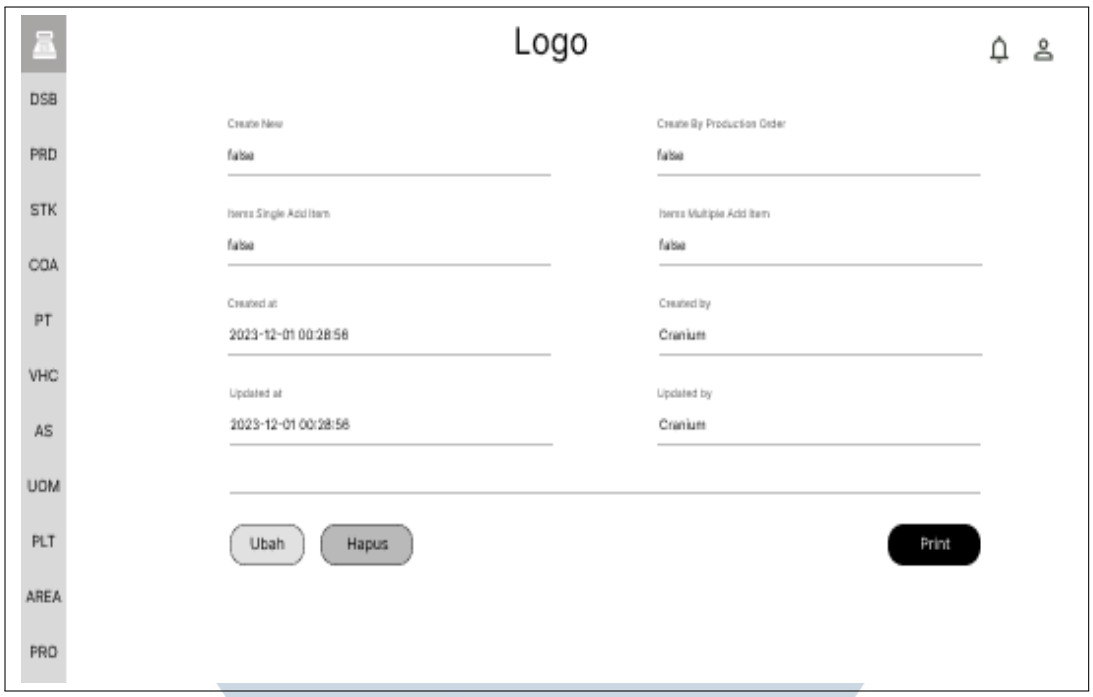
Gambar 3.52. *Wireframe Create* User Parameter

Pada Gambar 3.52 merupakan *wireframe* dari halaman *Create* User Parameter. Halaman *create* user parameter menampilkan form untuk melakukan *create* dengan 4 (field) *field* yang terdiri dari sebuah *text field* dan 3 (tiga) *dropdown*. Pada halaman create terdapat button batal dan simpan.



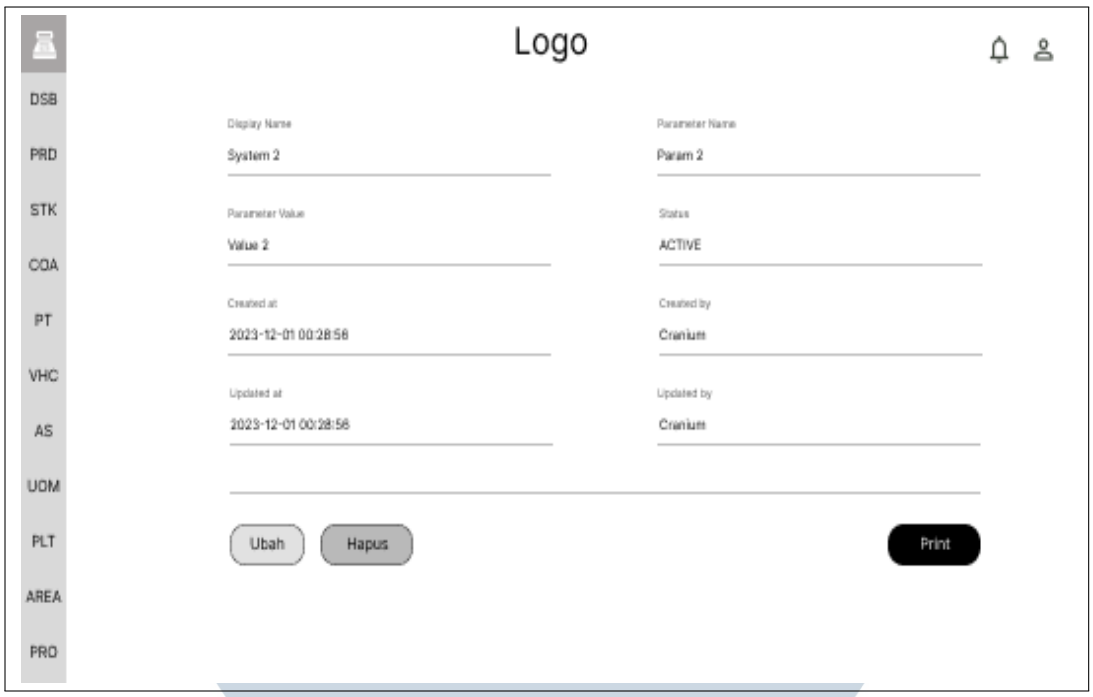
Gambar 3.53. *Wireframe View* Customer Type

Pada Gambar 3.53 merupakan *wireframe* dari halaman *View* Customer Type. Halaman *view* customer type menampilkan form untuk melakukan *view* data berdasarkan id yang dikirimkan dari halaman paging. Pada halaman ini ditampilkan isi data yang sudah didapatkan dari database beserta button ubah, button hapus, dan button print.



Gambar 3.54. *Wireframe View* Item Withdraw Reservation Setting

Pada Gambar 3.54 merupakan *wireframe* dari halaman *View* Item Withdraw Reservation Setting. Halaman *view* Item Withdraw Reservation Setting menampilkan form untuk melakukan *view* data berdasarkan id yang dikirimkan dari halaman paging. Pada halaman ini ditampilkan isi data yang sudah didapatkan dari database beserta button ubah, button hapus, dan button print.

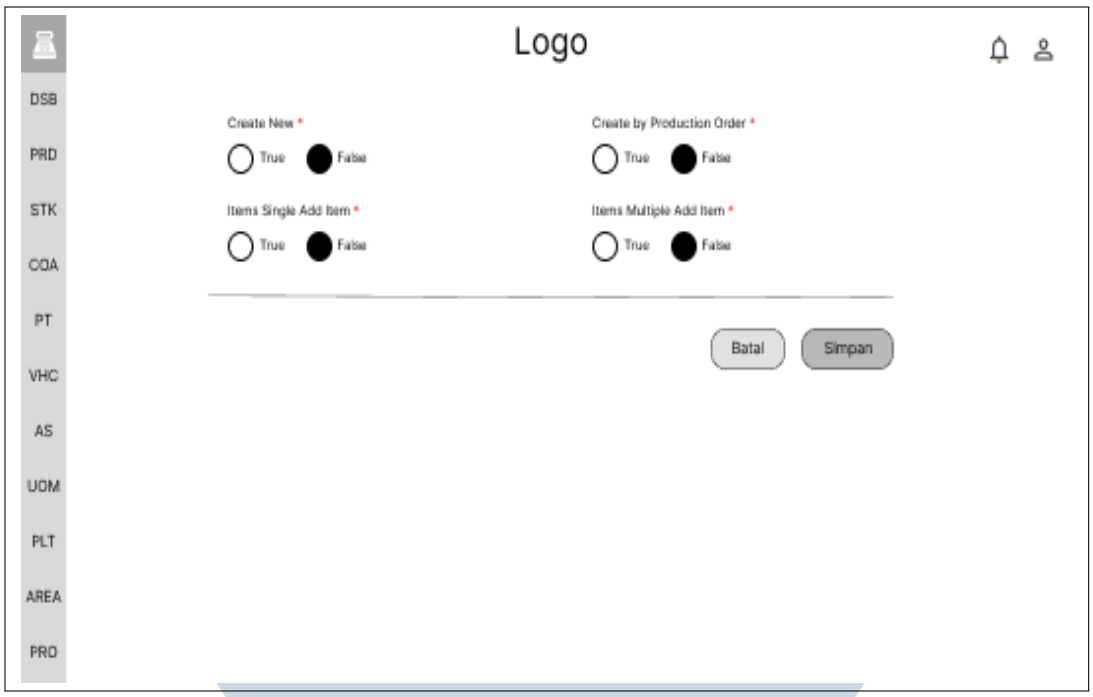


Gambar 3.55. *Wireframe View* User Parameter

Pada Gambar 3.55 merupakan *wireframe* dari halaman *View* User Parameter. Halaman *view* user parameter menampilkan form untuk melakukan *view* data berdasarkan id yang dikirimkan dari halaman paging. Pada halaman ini ditampilkan isi data yang sudah didapatkan dari database beserta button ubah, button hapus, dan button print.

Gambar 3.56. *Wireframe Update* Customer Type

Pada Gambar 3.56 merupakan *wireframe* dari halaman *Update* Customer Type. Halaman *update* customer type menampilkan form untuk melakukan *update* dengan 3 (tiga) *field* yang terdiri dari sebuah *text field* dan 2 (dua) *dropdown*. Pada halaman *update* terdapat button batal dan simpan. *Input field* pada form *update* sudah terisi dengan data yang sudah dicreate sebelumnya.



Gambar 3.57. *Wireframe Update* Item Withdraw Reservation Setting

Pada Gambar 3.57 merupakan *wireframe* dari halaman *Update* Item Withdraw Reservation Setting. Halaman *update* Item Withdraw Reservation Setting menampilkan form untuk melakukan *update* dengan 4 (empat) *input field* yang terdiri dari 4 (empat) *radio button* dengan *value true* dan *false*. Pada halaman *update* terdapat button batal dan simpan. *Input field* pada *form update* sudah terisi dengan data yang sudah dicreate sebelumnya.

Gambar 3.58. *Wireframe Update* User Parameter

Pada Gambar 3.58 merupakan *wireframe* dari halaman *Update* User Parameter. Halaman *update* user parameter menampilkan form untuk melakukan *update* dengan 4 (field) *field* yang terdiri dari sebuah *text field* dan 3 (tiga) *dropdown*. Pada halaman update terdapat button batal dan simpan. *Input field* pada *form update* sudah terisi dengan data yang sudah dicreate sebelumnya.

Gambar 3.59. *Wireframe Delete* Customer Type

Pada Gambar 3.59 merupakan *wireframe* dari halaman *Delete* Customer Type. Halaman *delete* customer type menampilkan dialog konfirmasi untuk penghapusan data dengan tombol Tidak untuk membatalkan penghapusan data dan Ya untuk mengkonfirmasi penghapusan data.

Gambar 3.60. *Wireframe Delete* Item Withdraw Reservation Setting

Pada Gambar 3.60 merupakan *wireframe* dari halaman *Delete* Item Withdraw Reservation Setting. Halaman *delete* Item Withdraw Reservation Setting menampilkan dialog konfirmasi untuk penghapusan data dengan tombol Tidak untuk membatalkan penghapusan data dan Ya untuk mengkonfirmasi penghapusan data.

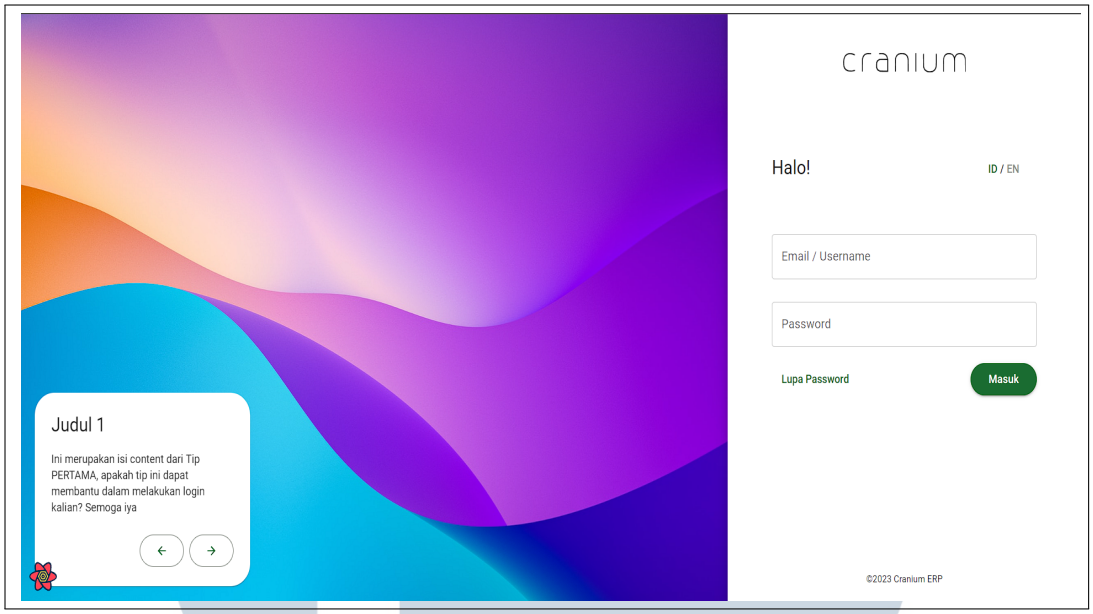
Gambar 3.61. *Wireframe Delete* User Parameter

Pada Gambar 3.61 merupakan *wireframe* dari halaman *Delete* User Parameter. Halaman *delete* user parameter menampilkan dialog konfirmasi untuk penghapusan data dengan tombol Tidak untuk membatalkan penghapusan data dan Ya untuk mengkonfirmasi penghapusan data.

3.4.7 Hasil Implementasi

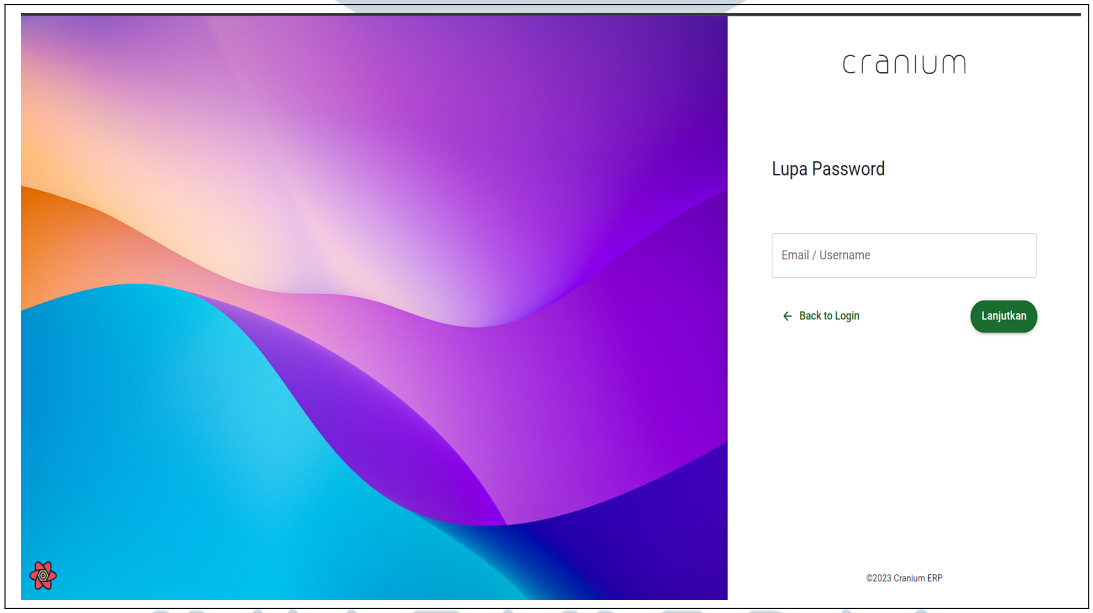
Berikut adalah beberapa tangkapan layar dari situs web ERP Cranium yang telah dibuat berdasarkan perancangan yang dijelaskan sebelumnya. Terdapat beberapa perbedaan antara wireframe dan tampilan yang ditampilkan pada bagian hasil implementasi, terutama pada ikon dan warna yang digunakan. Perbedaan ini disebabkan oleh kebutuhan untuk menyesuaikan ikon dengan komponen MUI (Material UI), yang merupakan salah satu dari pustaka React. Penyesuaian ukuran, warna, dan penempatan juga direvisi bersama dengan desainer seiring berjalannya waktu.

Hasil tampilan Login dapat dilihat pada Gambar 3.62.



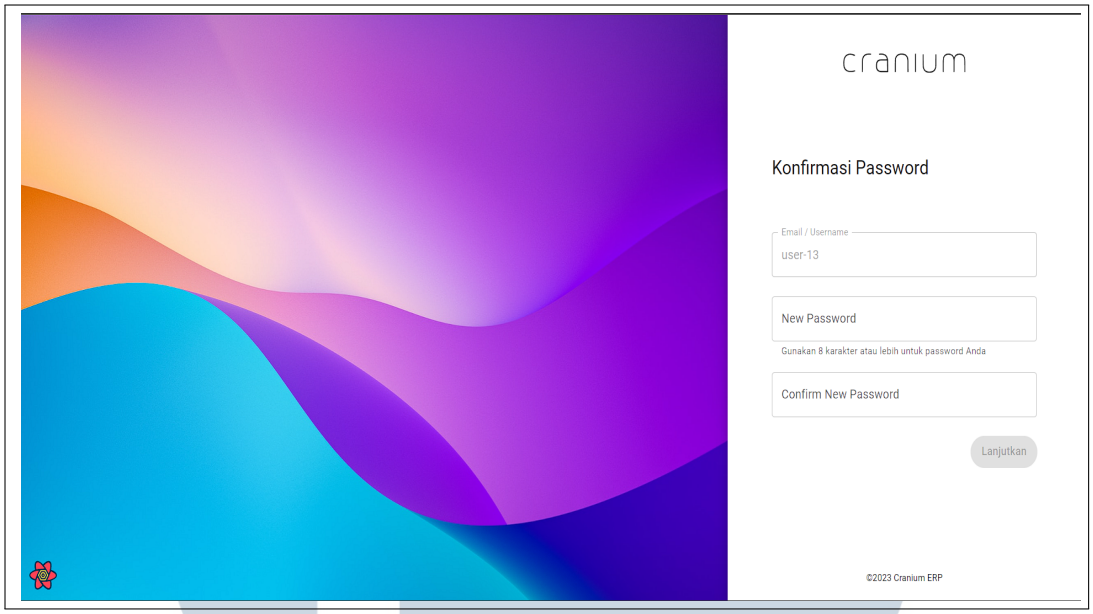
Gambar 3.62. Tampilan Login

Hasil tampilan Forgot Password dapat dilihat pada Gambar 3.63.



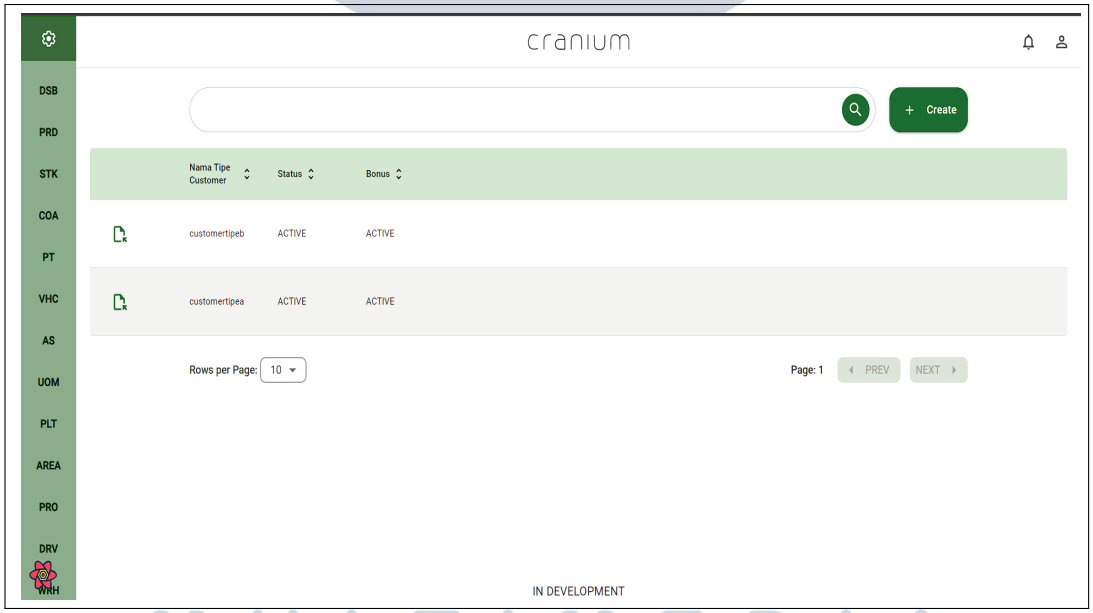
Gambar 3.63. Tampilan Forgot Password

Hasil tampilan Confirm Password dapat dilihat pada Gambar 3.64.



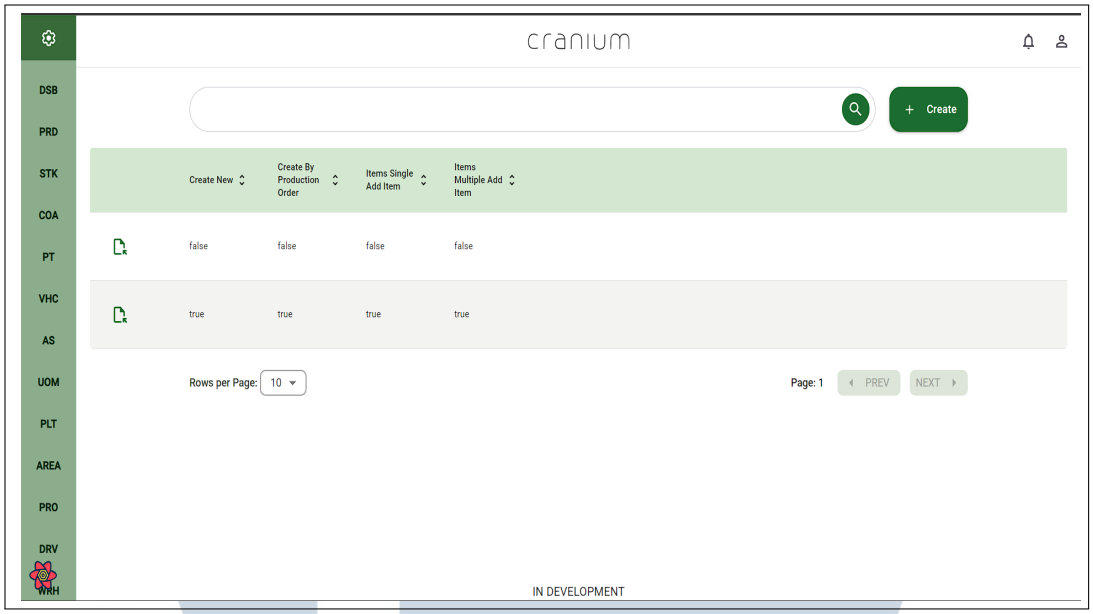
Gambar 3.64. Tampilan Confirm Password

Hasil tampilan Paging Customer Type dapat dilihat pada Gambar 3.65.



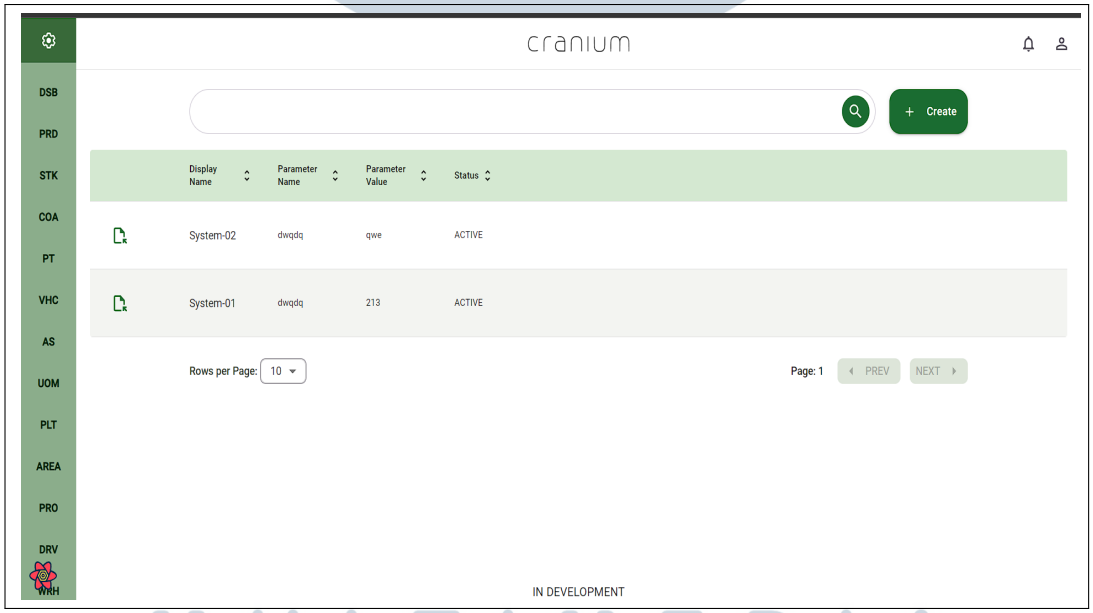
Gambar 3.65. Tampilan *Paging* Customer Type

Hasil tampilan Paging Item Withdraw Reservation Setting dapat dilihat pada Gambar 3.66.



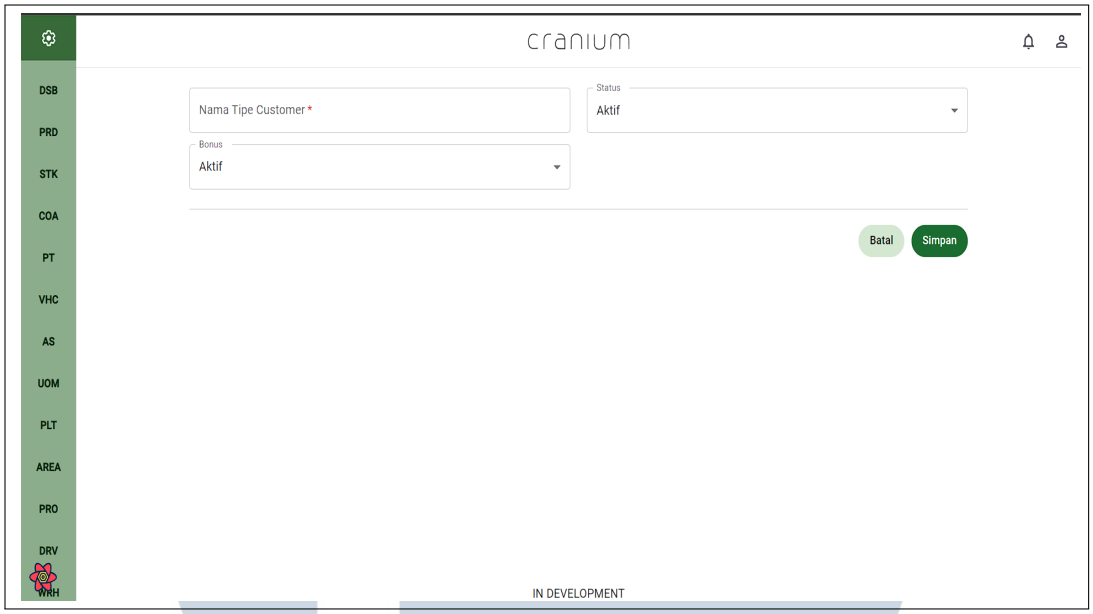
Gambar 3.66. Tampilan *Paging* Item Withdraw Reservation Setting

Hasil tampilan Paging User Parameter dapat dilihat pada Gambar 3.67.



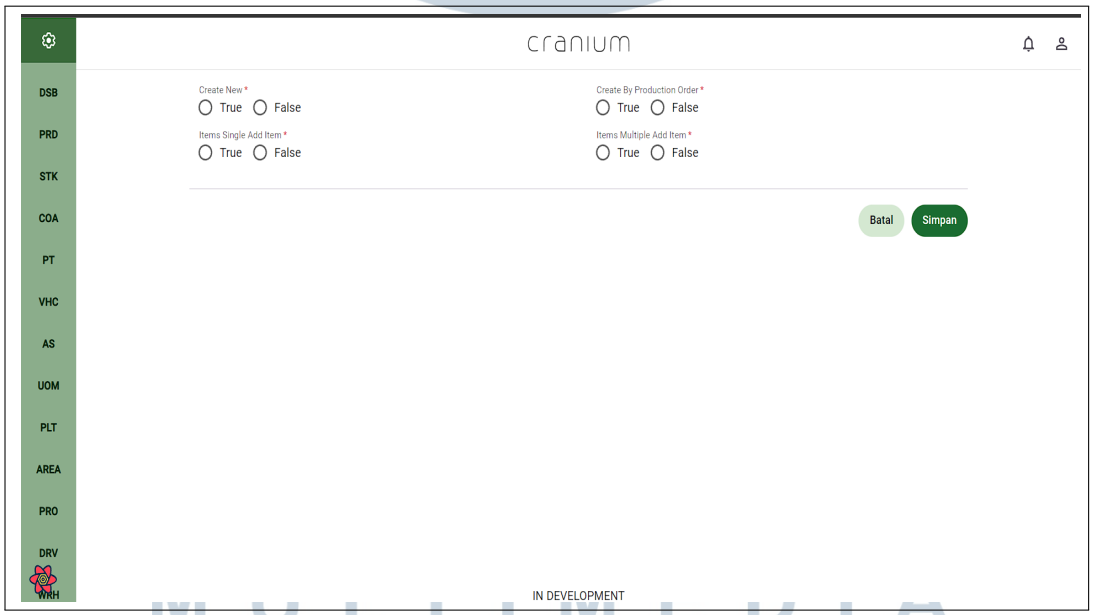
Gambar 3.67. Tampilan *Paging* User Parameter

Hasil tampilan Create Customer Type dapat dilihat pada Gambar 3.68.



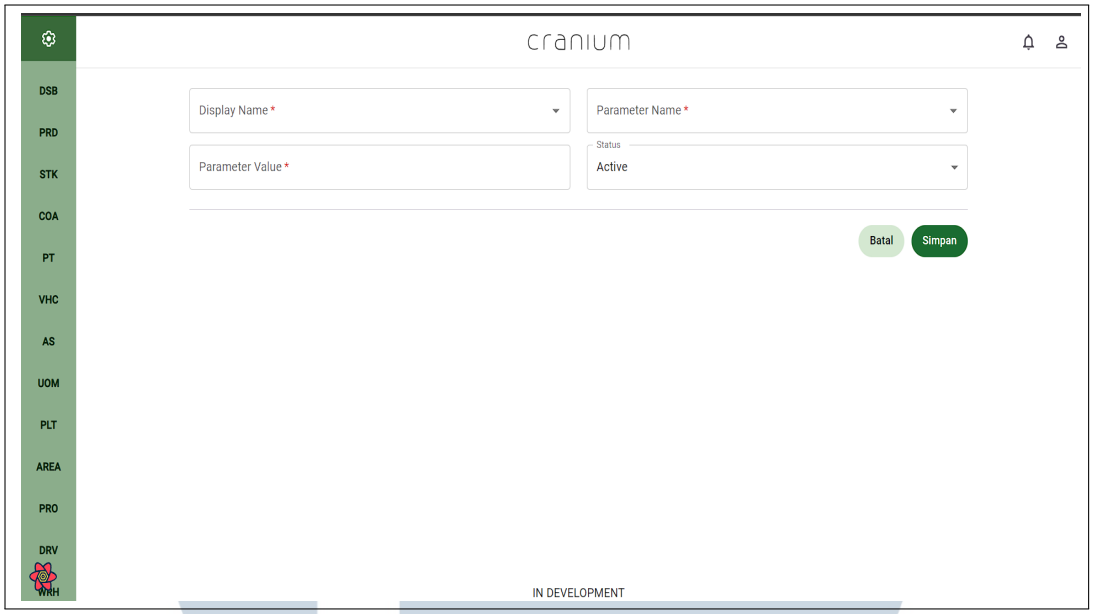
Gambar 3.68. Tampilan *Create* Customer Type

Hasil tampilan Create Item Withdraw Reservation Setting dapat dilihat pada Gambar 3.69.



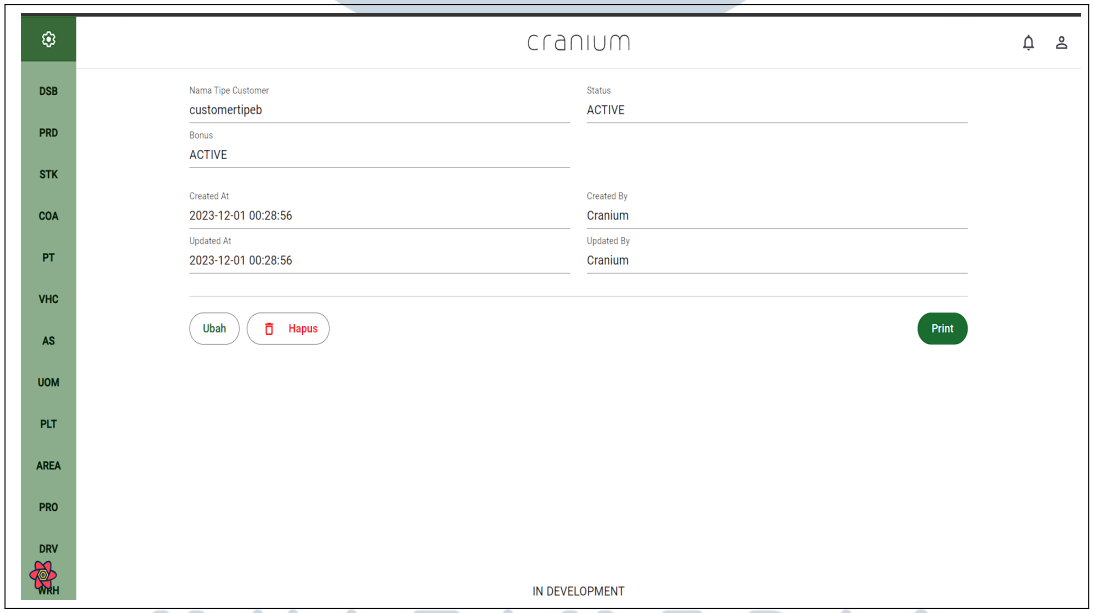
Gambar 3.69. Tampilan *Create* Item Withdraw Reservation Setting

Hasil tampilan Create User Parameter dapat dilihat pada Gambar 3.70.



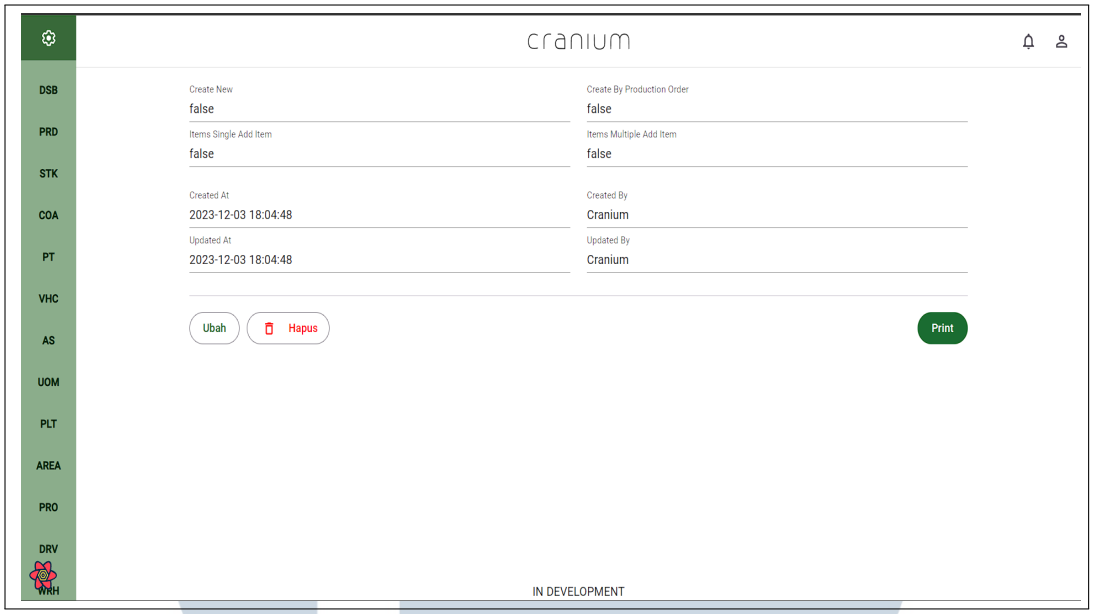
Gambar 3.70. Tampilan *Create* User Parameter

Hasil tampilan View Customer Type dapat dilihat pada Gambar 3.71.



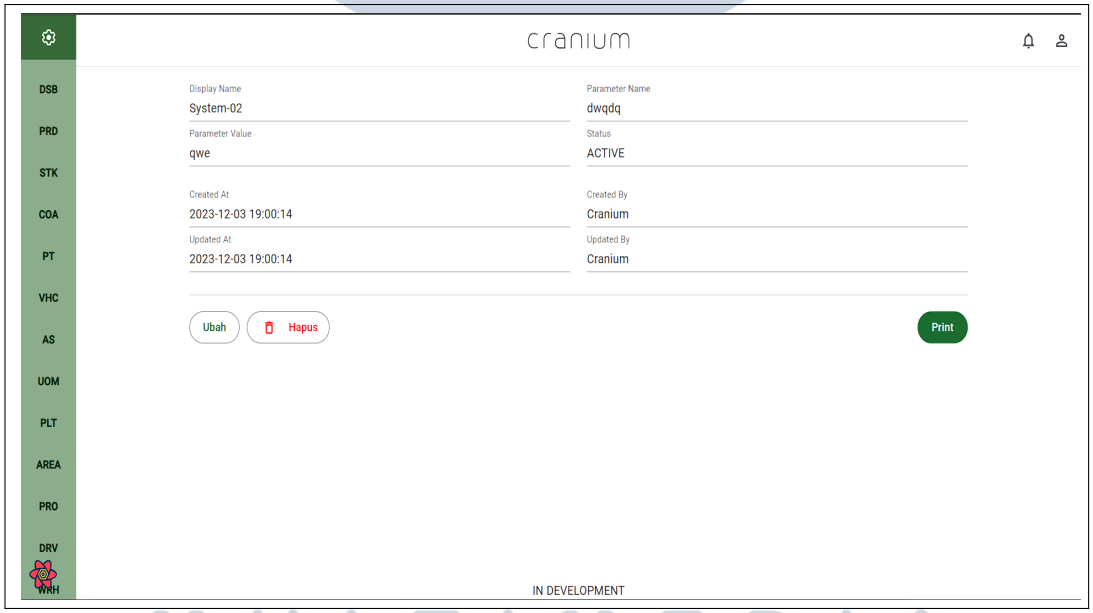
Gambar 3.71. Tampilan *View* Customer Type

Hasil tampilan View Item Withdraw Reservation Setting dapat dilihat pada Gambar 3.72.



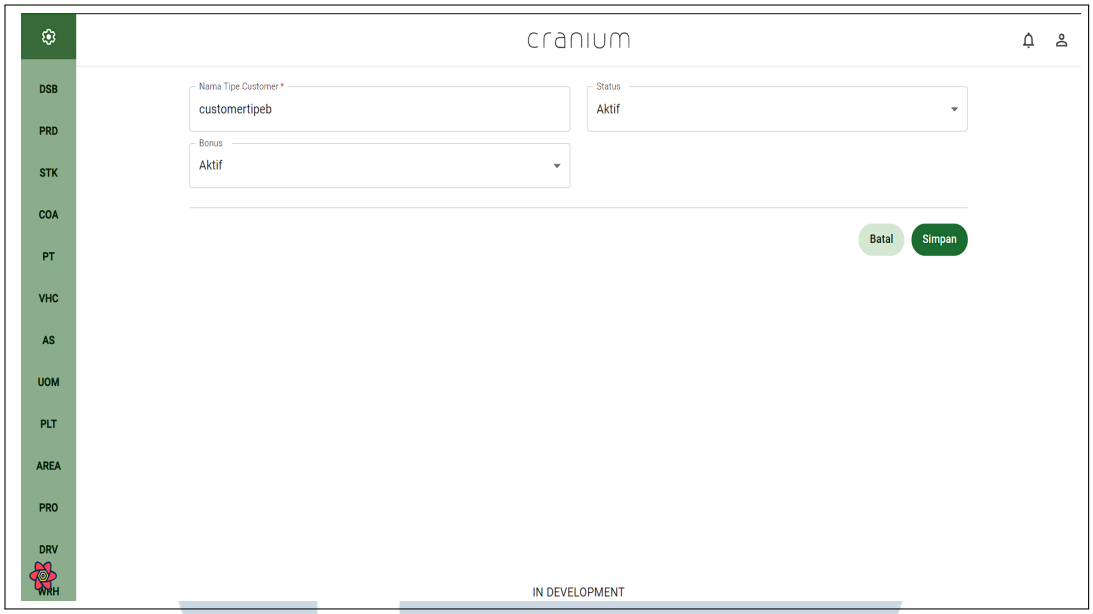
Gambar 3.72. Tampilan *View* Item Withdraw Reservation Setting

Hasil tampilan View User Parameter dapat dilihat pada Gambar 3.73.



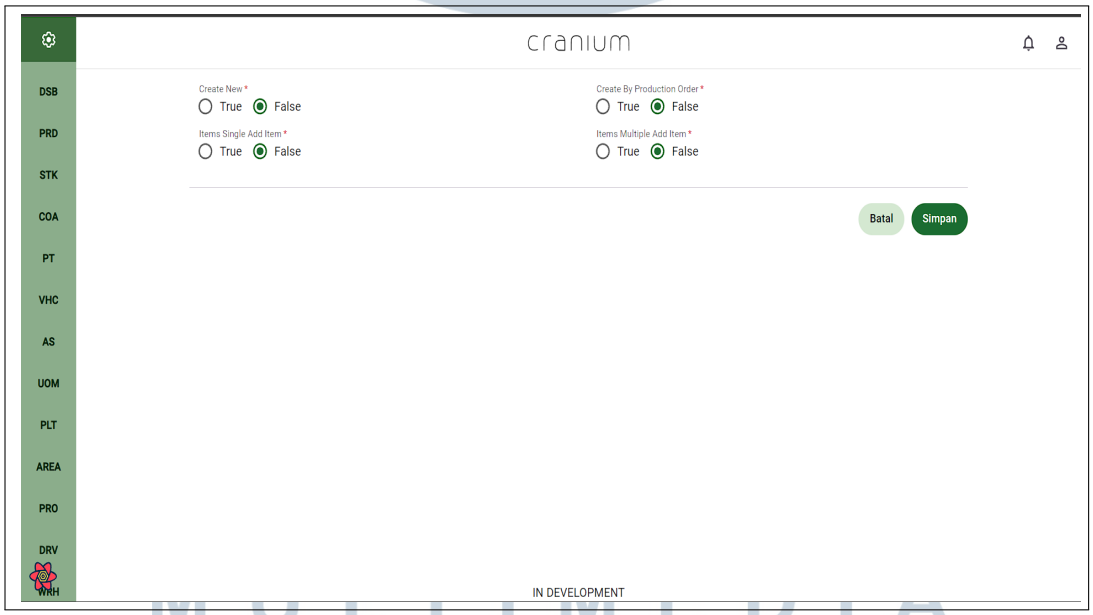
Gambar 3.73. Tampilan *View* User Parameter

Hasil tampilan Update Customer Type dapat dilihat pada Gambar 3.74.



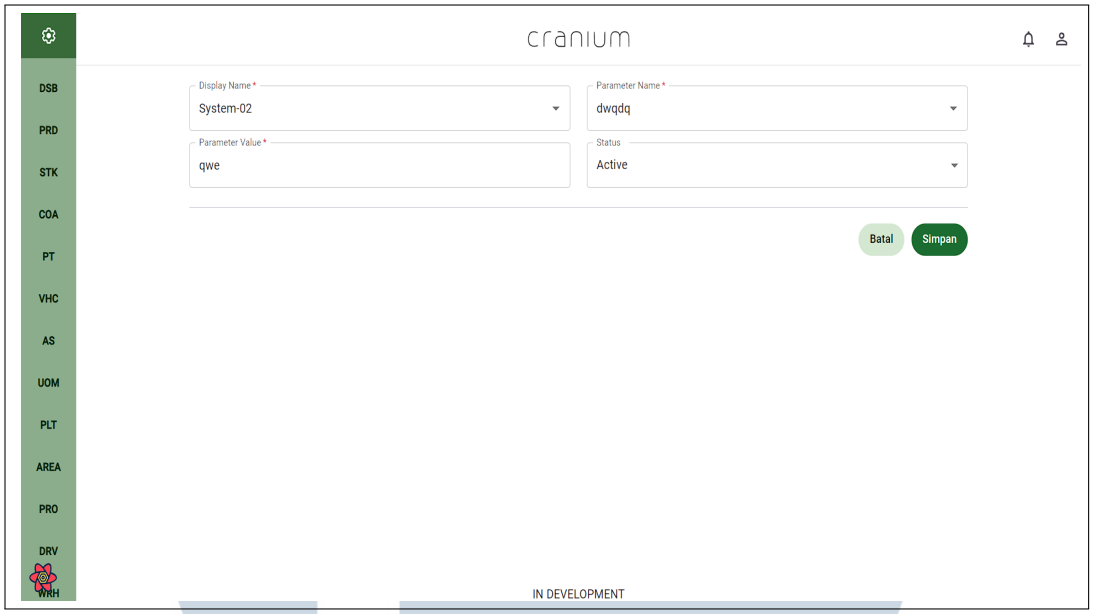
Gambar 3.74. Tampilan *Update* Customer Type

Hasil tampilan Update Item Withdraw Reservation Setting dapat dilihat pada Gambar 3.75.



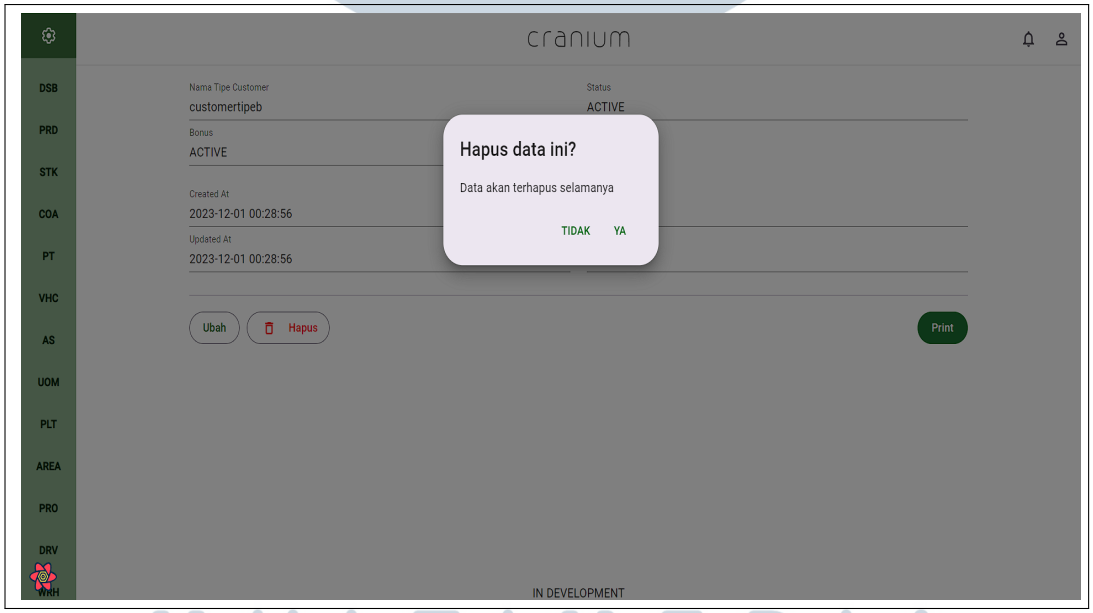
Gambar 3.75. Tampilan *Update* Item Withdraw Reservation Setting

Hasil tampilan Update User Parameter dapat dilihat pada Gambar 3.76.



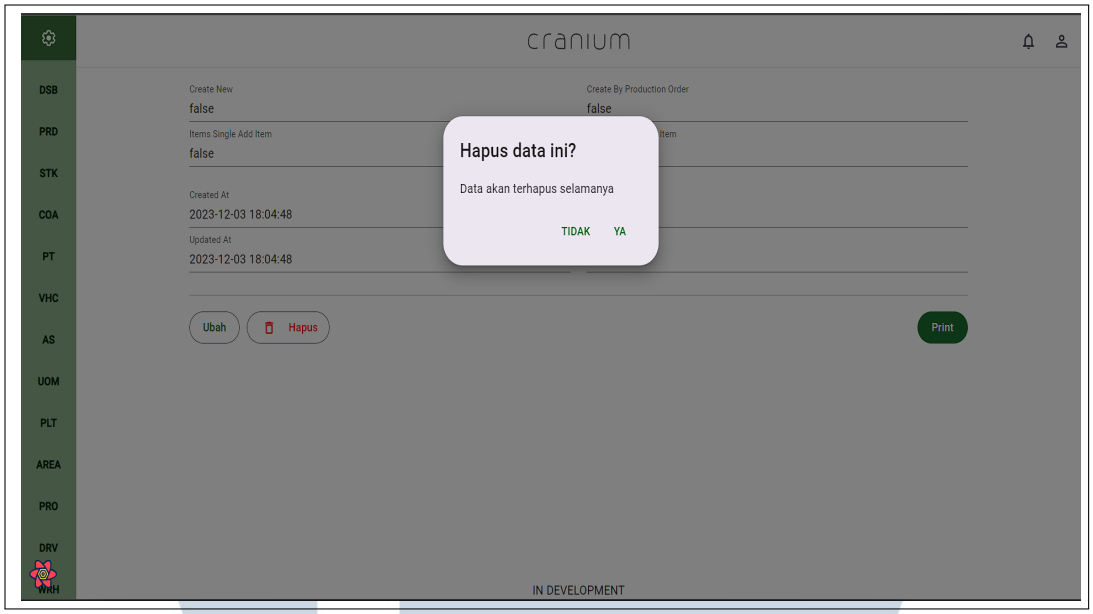
Gambar 3.76. Tampilan *Update* User Parameter

Hasil tampilan Delete Customer Type dapat dilihat pada Gambar 3.77.



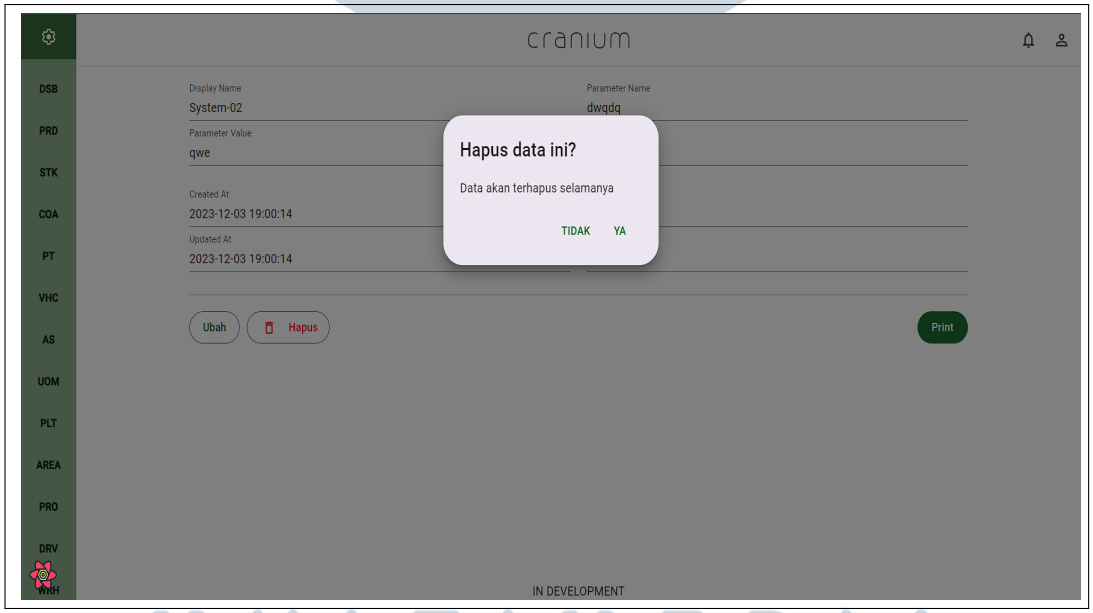
Gambar 3.77. Tampilan *Delete* Customer Type

Hasil tampilan Delete Item Withdraw Reservation Setting dapat dilihat pada Gambar 3.78.



Gambar 3.78. Tampilan *Delete* Item Withdraw Reservation Setting

Hasil tampilan Delete User Parameter dapat dilihat pada Gambar 3.79.



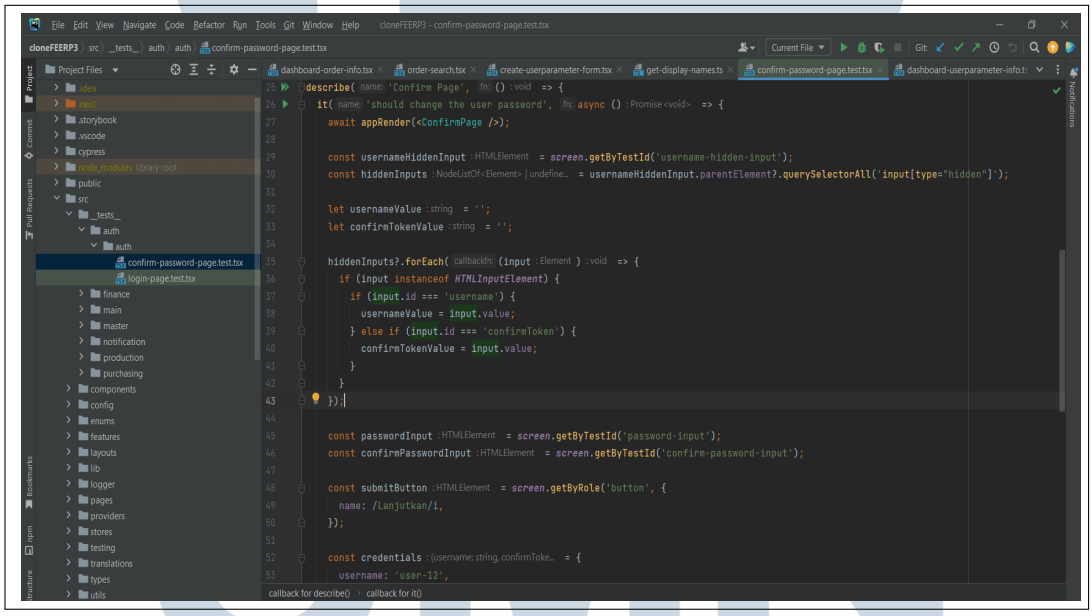
Gambar 3.79. Tampilan *Delete* User Parameter

3.4.8 Unit Test

Pembuatan unit test dilakukan untuk memastikan bahwa code yang sudah dibuat selama pengembangan aplikasi dapat berjalan sebagaimana mestinya.

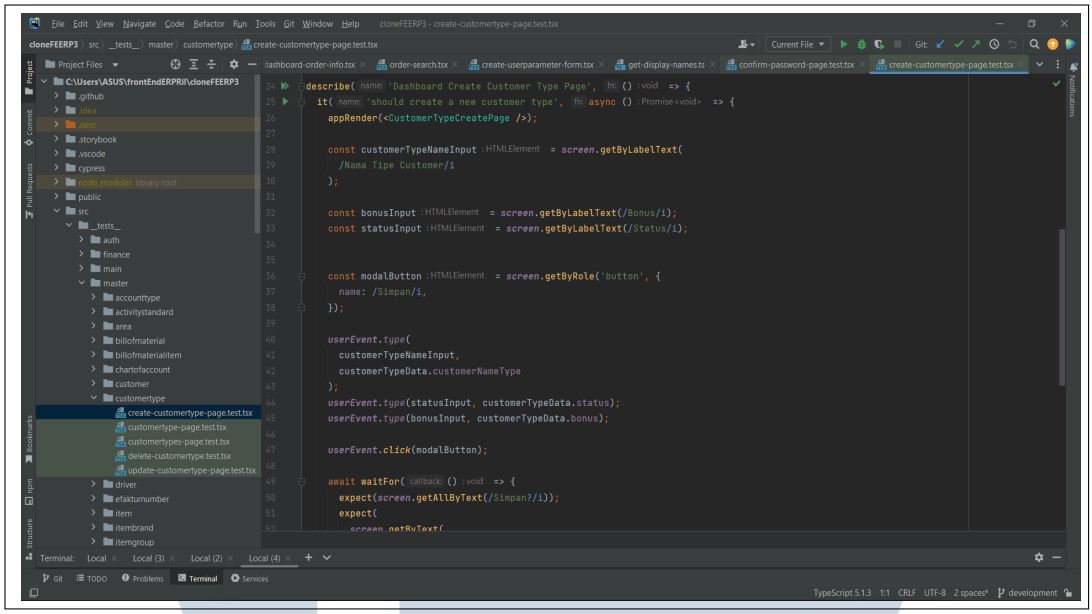
Unit test dibuat dengan memperhatikan alur bagaimana user berhasil mengakses suatu fitur. Dalam pengujian unit pada *frontend*, *mocking* pada *handler* API dimanfaatkan untuk membuat halaman seolah-olah memanggil API, padahal nilai yang dikembalikan berasal dari fungsi yang memanggil API tersebut. Pengujian unit dibuat seolah-olah pengguna melakukan input pada bidang-bidang yang diperlukan dengan memanfaatkan *data-test-id* pada komponen yang digunakan. Sementara itu, pengujian unit pada *backend* digunakan untuk menguji API yang telah dibuat. Terdapat dua kondisi pengujian, yaitu *positive case* yang menggambarkan proses pemanggilan API yang berhasil, dan *negative case* yang menggambarkan proses pemanggilan API yang gagal dan menghasilkan *error*.

Untuk *unit test* yang dibuat untuk Confirm Password dapat dilihat pada Gambar 3.80.

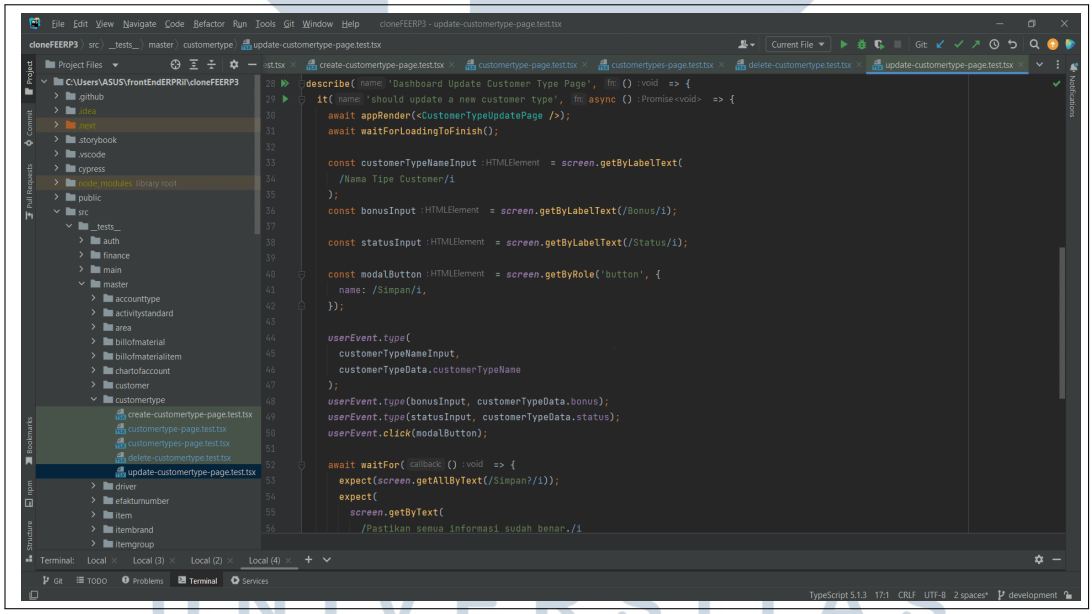


Gambar 3.80. *Unit Test* Confirm Password

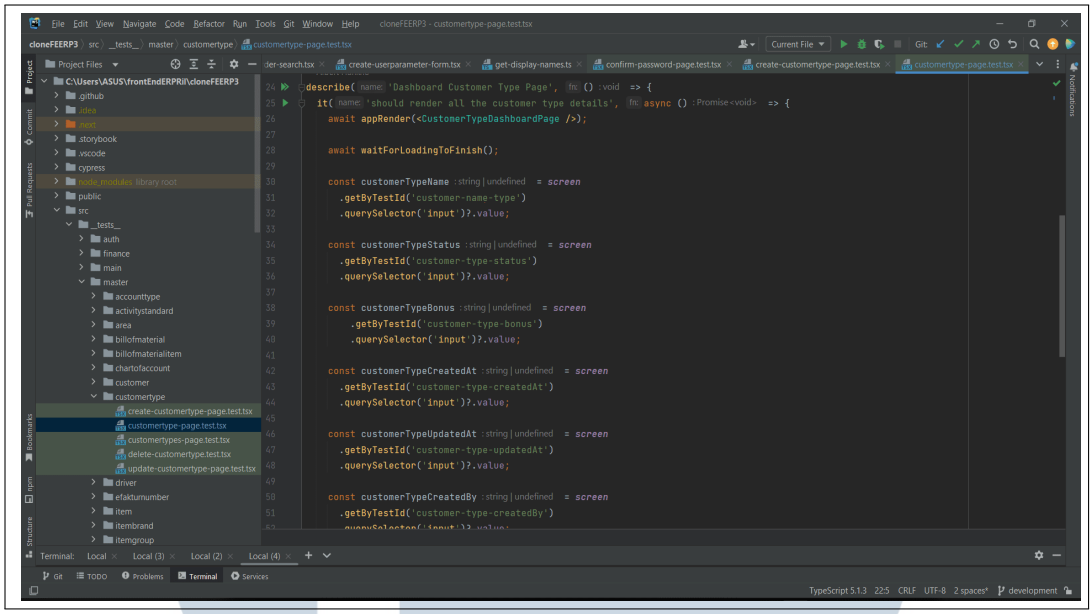
Untuk *unit test* yang dibuat untuk Customer Type pada method *create*, *read*, *update*, *delete*, dan *paging* dapat dilihat pada Gambar 3.81, 3.82, 3.83, 3.84, 3.85.



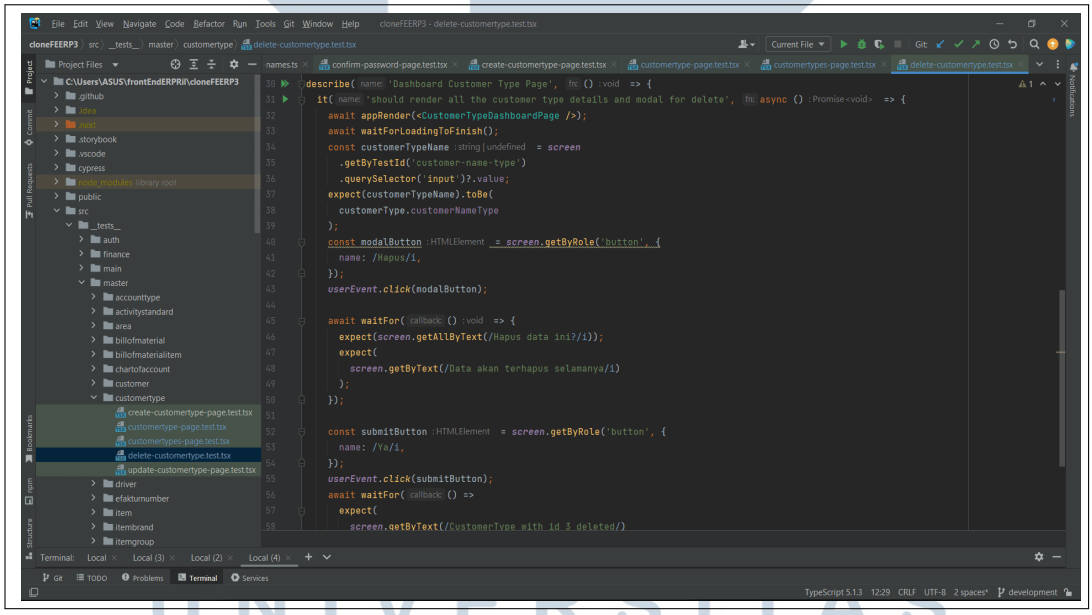
Gambar 3.81. *Unit Test Create* Customer Type



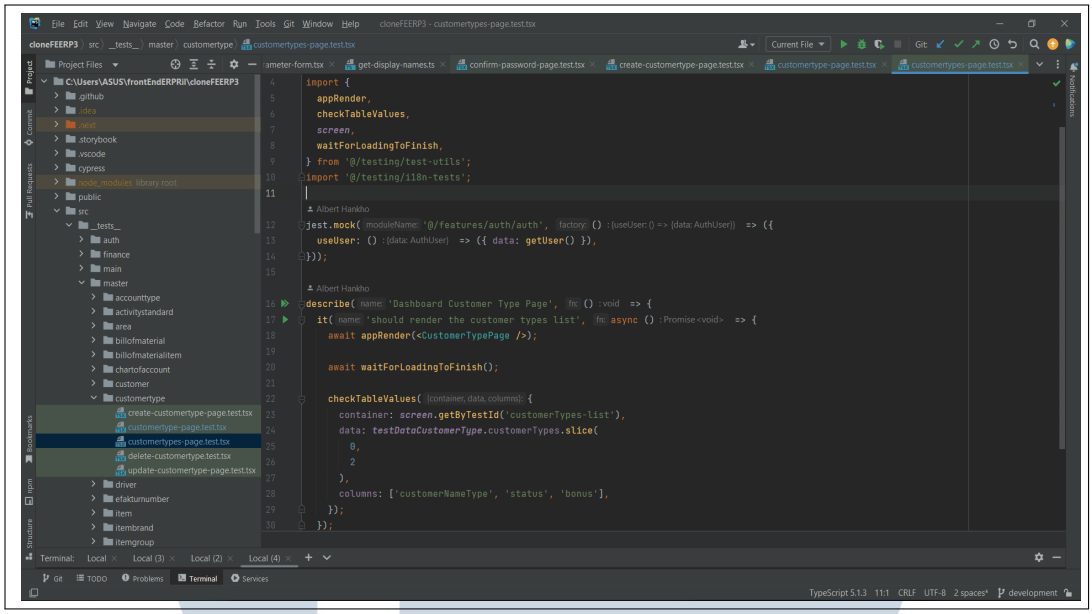
Gambar 3.82. *Unit Test Update* Customer Type



Gambar 3.83. *Unit Test Read* Customer Type

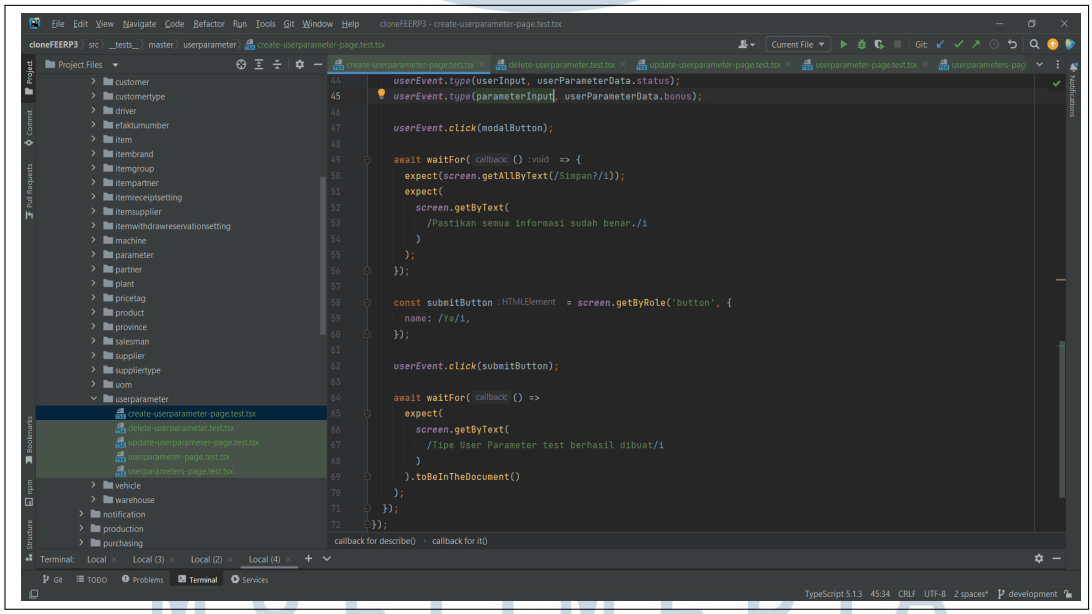


Gambar 3.84. *Unit Test Delete* Customer Type

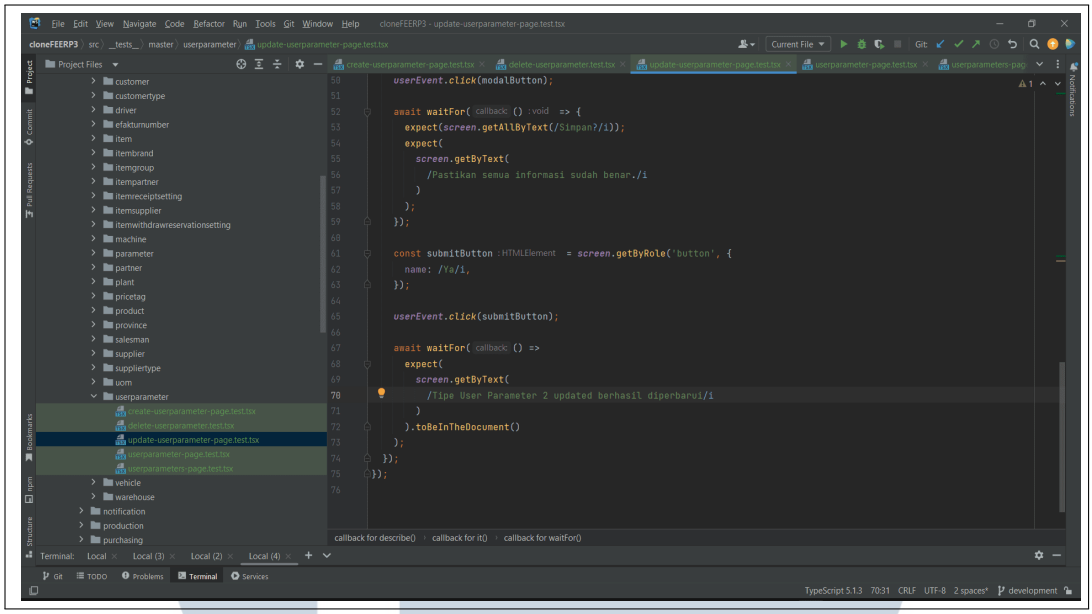


Gambar 3.85. *Unit Test Paging* Customer Type

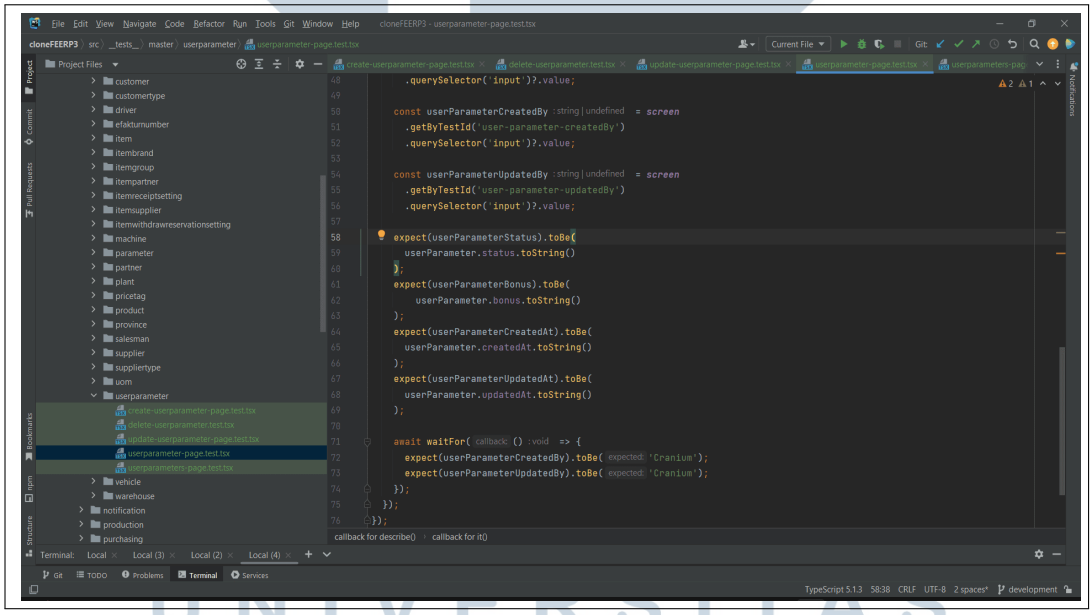
Untuk *unit test* yang dibuat untuk User Parameter pada method *create*, *read*, *update*, *delete*, dan *paging* dapat dilihat pada Gambar 3.86, 3.87, 3.88, 3.89, 3.90.



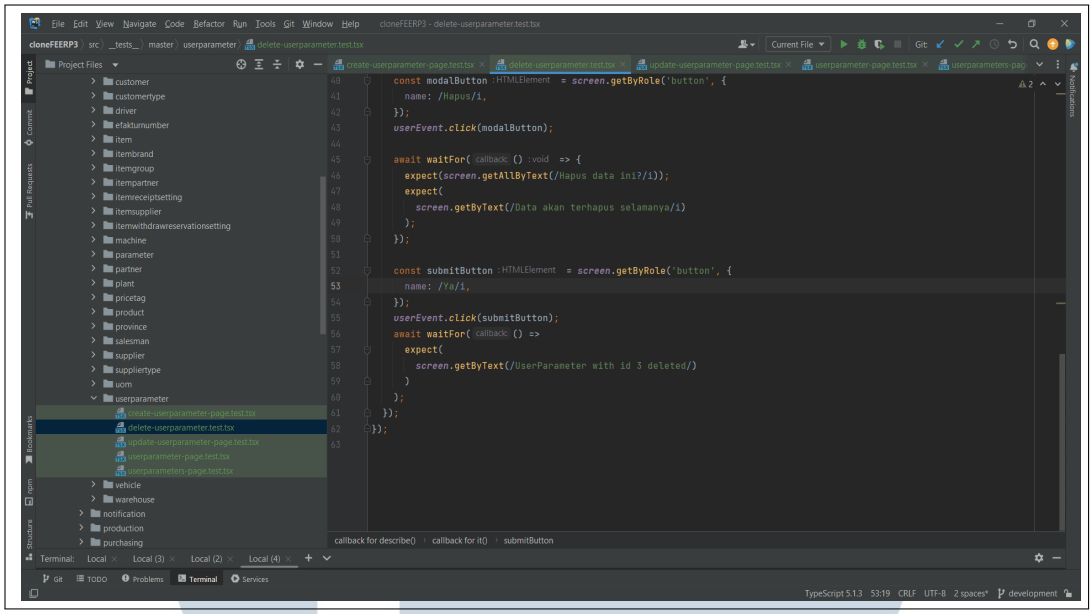
Gambar 3.86. *Unit Test Create* User Parameter



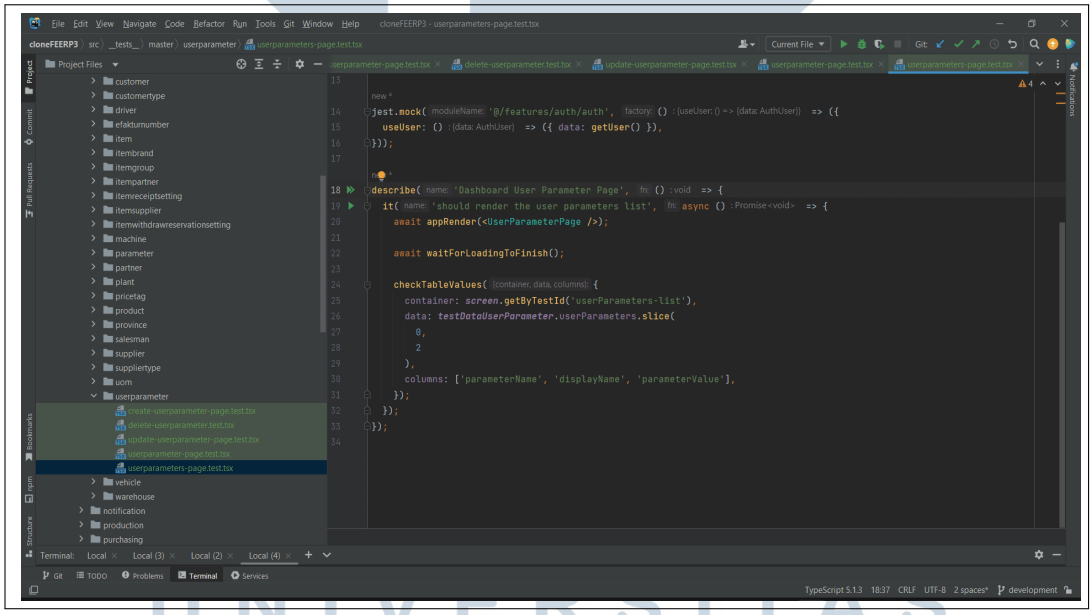
Gambar 3.87. *Unit Test Update* User Parameter



Gambar 3.88. *Unit Test Read* User Parameter

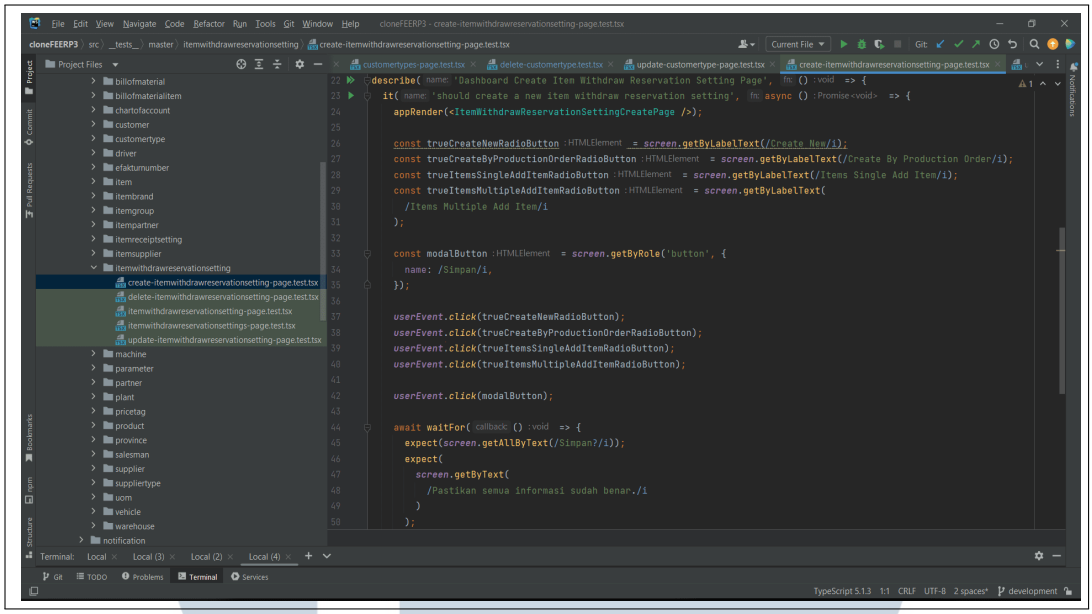


Gambar 3.89. *Unit Test Delete* User Parameter

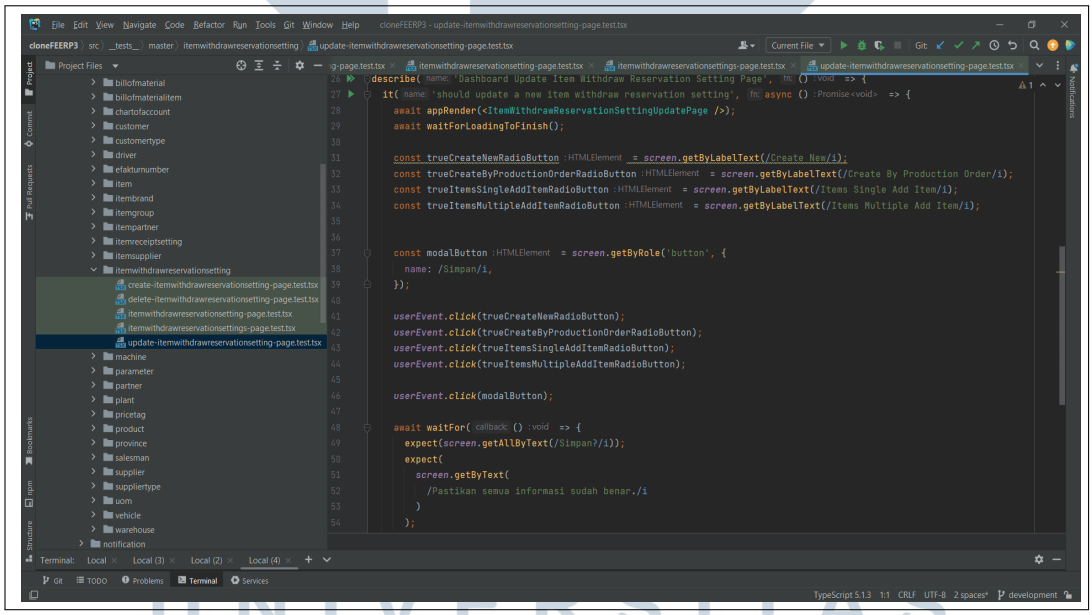


Gambar 3.90. *Unit Test Paging* User Parameter

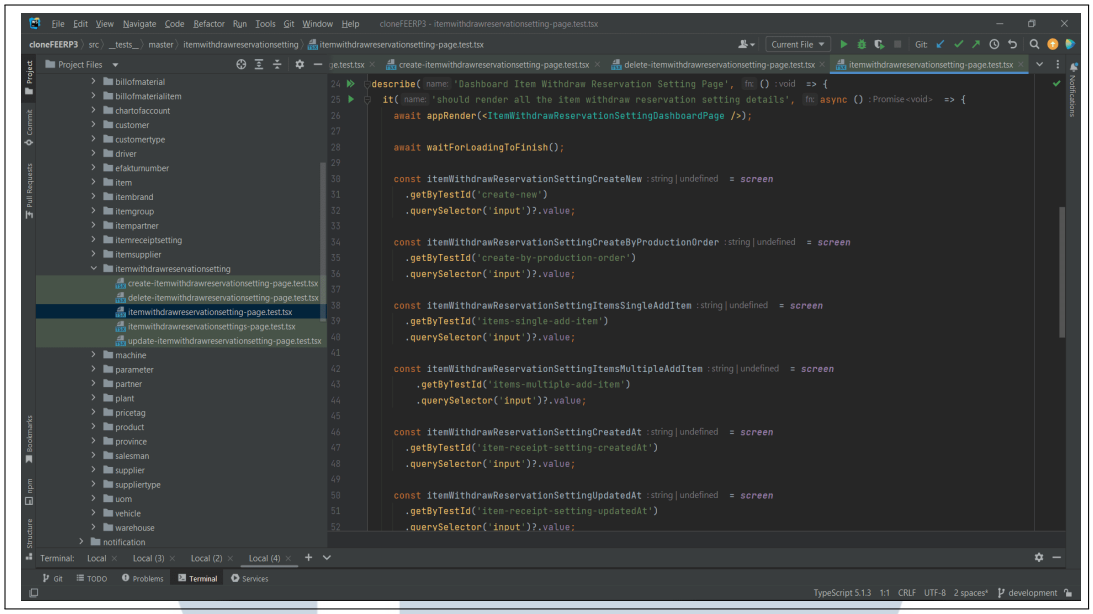
Untuk *unit test* yang dibuat untuk Item Withdraw Reservation Setting pada method *create*, *read*, *update*, *delete*, dan *paging* dapat dilihat pada Gambar 3.91, 3.92, 3.93, 3.94, 3.95.



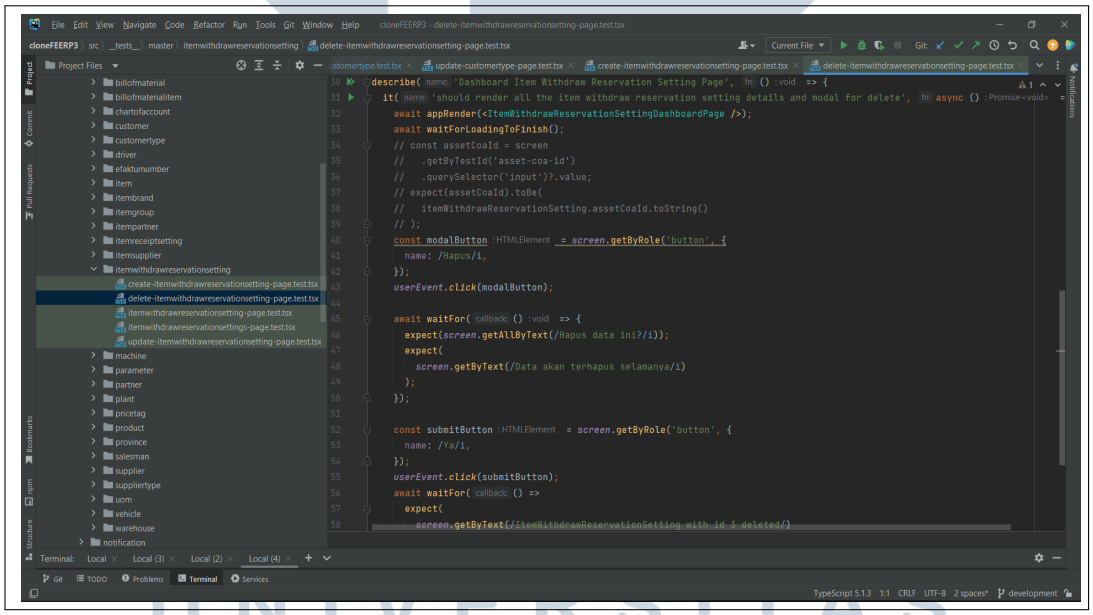
Gambar 3.91. *Unit Test Create* Item Withdraw Reservation Setting



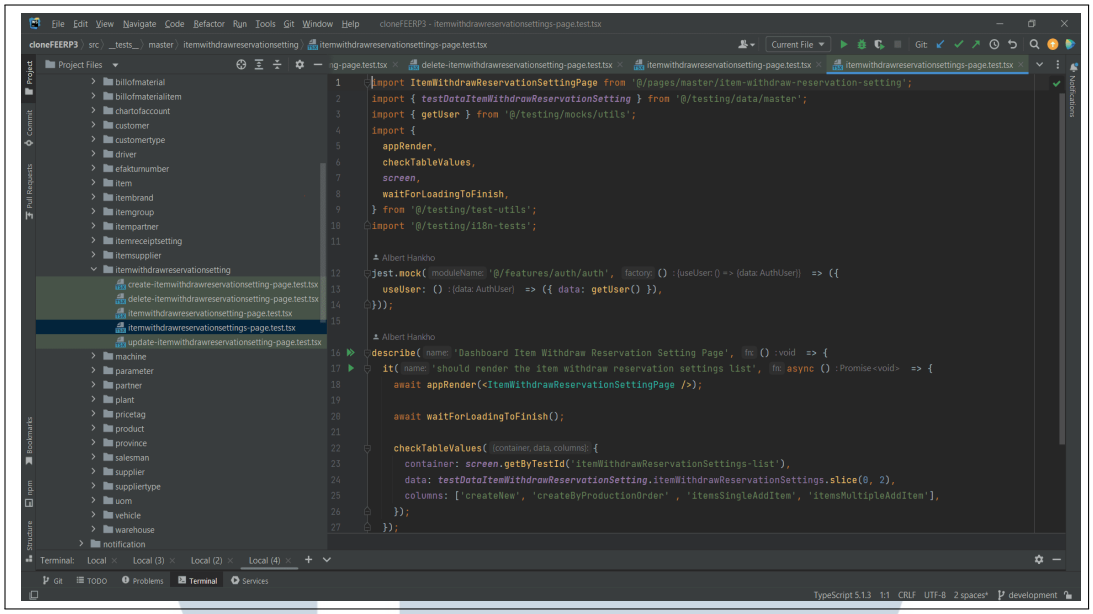
Gambar 3.92. *Unit Test Update* Item Withdraw Reservation Setting



Gambar 3.93. *Unit Test Read* Item Withdraw Reservation Setting

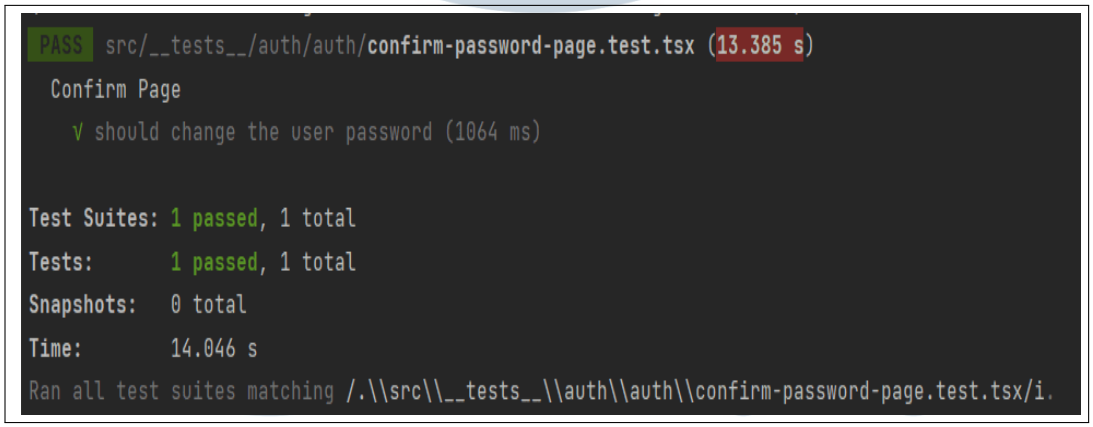


Gambar 3.94. *Unit Test Delete* Item Withdraw Reservation Setting



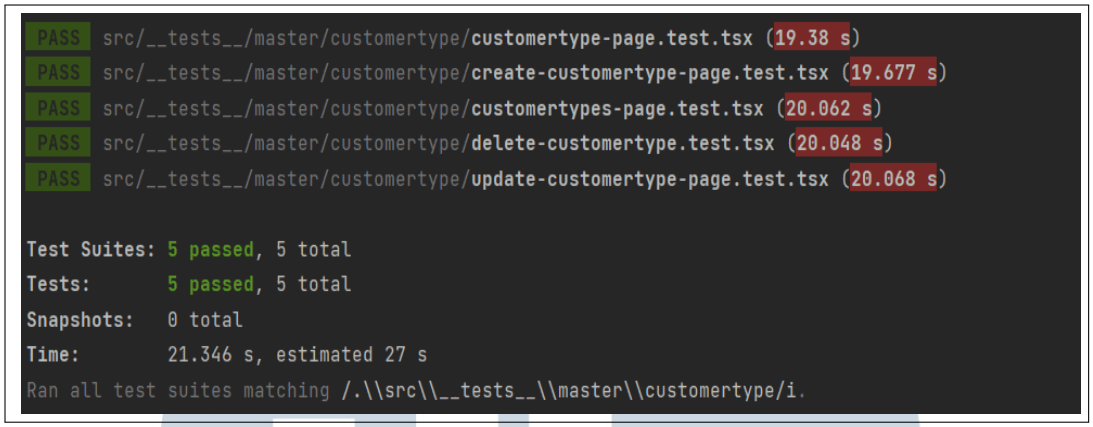
Gambar 3.95. *Unit Test Paging* Item Withdraw Reservation Setting

Berikut hasil dari pengujian unit test modul Confirm Password yang dapat dilihat pada Gambar 3.96.



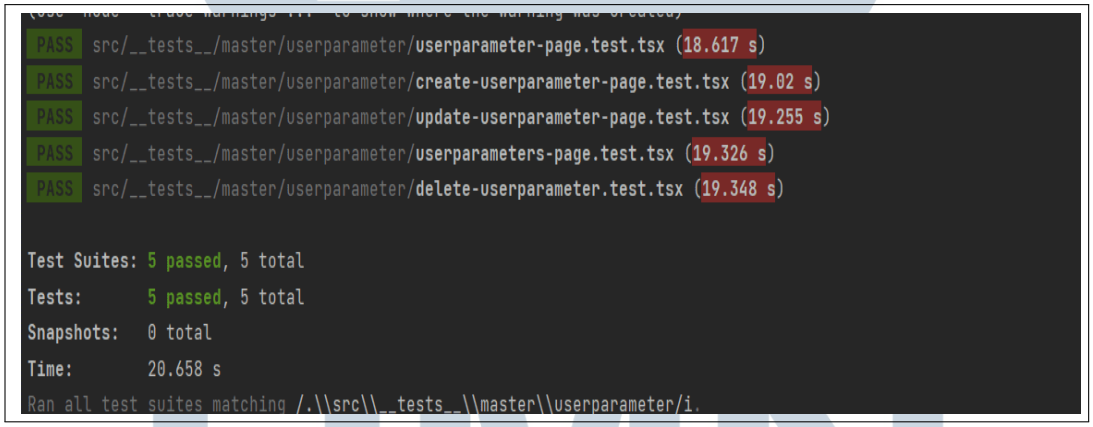
Gambar 3.96. Hasil *Unit Test* Confirm Password

Berikut hasil dari pengujian unit test modul Customer Type yang dapat dilihat pada Gambar 3.97.



Gambar 3.97. Hasil *Unit Test* Customer Type

Berikut hasil dari pengujian unit test modul User Parameter yang dapat dilihat pada Gambar 3.98.



Gambar 3.98. Hasil *Unit Test* User Parameter

Berikut hasil dari pengujian unit test modul Item Withdraw Reservation Setting yang dapat dilihat pada Gambar 3.99.

Gambar 3.99. Hasil *Unit Test* Item Withdraw Reservation Setting

3.5 Kendala dan Solusi yang Ditemukan

Berdasarkan kerja magang yang dilakukan, terdapat beberapa kendala yang ditemukan, seperti:

- Mengalami tantangan dalam menggunakan Visual Studio Code sebagai IDE untuk proyek sistem ERP karena memerlukan integrasi manual dengan plugin seperti Spring dan Postgres, serta ekstensi lainnya. Selain itu, terdapat kesulitan dalam mengatur struktur penyimpanan *file source code* di dalam Visual Studio Code.
- Kesulitan dalam melakukan revisi terhadap *database schema* agar sesuai dengan *business process* serta pengembangan proses forgot password dan confirm password yang kompleks.
- Kesulitan dalam pembuatan unit test yang cukup kompleks harus menampilkan kasus ketika berhasil dan ketika gagal.

Berdasarkan kendala-kendala yang dihadapi selama melakukan kerja magang, berikut merupakan solusi-solusi yang ditemukan untuk mengatasinya.

- Beralih dari penggunaan Visual Studio Code ke Intellij IDEA membawa kemudahan dalam mengintegrasikan *framework* Spring, mengelola *database* Postgres, dan mengatur struktur *file source code* yang dikembangkan dengan terstruktur dan rapi. Untuk *frontend* digunakan IDE WebStorm yang memiliki kemiripan dengan Intellij.

- Mengikuti *training* dengan supervisor dan orang yang mengembangkan sistem ERP sebelumnya untuk mendapatkan pemahaman yang lebih mendalam terkait alur proses bisnis sistem ERP.
- Berdiskusi dengan supervisor mengenai kriteria dan proses *unit test* yang dibuat serta terus melakukan *trial and error* agar mendapatkan hasil yang diinginkan dari unit test yang dibuat.