BAB 3
METODOLOGI

3.1 Kedudukan dan Koordinasi

Pelaksanaan kerja magang dilakukan di PT Hexaon Business Mitrasindo dengan penempatan posisi sebagai Fullstack Developer. Dalam pelaksanaan magang, kegiatan pekerjaan berada di bawah supervisi Nur Hamdalah Kahfi selaku pembimbing/supervisor, dengan koordinasi bersama tim pengembang yang terlibat pada proyek JEBRET.

Koordinasi kerja dilakukan secara terstruktur melalui komunikasi rutin dengan tim frontend dan backend. Pola koordinasi ini bertujuan untuk memastikan implementasi fitur sesuai kebutuhan bisnis, menjaga konsistensi integrasi antar-lapisan sistem, serta mempercepat proses penyelesaian kendala teknis yang muncul selama pengembangan. Selain itu, selama masa magang dilakukan evaluasi progres secara berkala melalui rapat mingguan untuk membahas capaian pekerjaan, kendala, serta rencana pengembangan berikutnya.

Kegiatan magang dilaksanakan dengan jadwal kerja hari Senin–Jumat pukul 09.00–17.30 WIB dan dilakukan *on-site* di kantor Hexaon dengan fleksibilitas pembagian waktu sesuai kebutuhan proyek.

3.2 Tugas yang Dilakukan

Selama pelaksanaan magang di PT Hexaon Business Mitrasindo, peserta magang ditempatkan sebagai Fullstack Developer dan berfokus pada perancangan serta pengembangan aplikasi web JEBRET (Jalin Electronic Billing and Realtime Tracking). Pengembangan sistem dilakukan menggunakan Angular pada sisi frontend dan Spring Boot pada sisi backend, serta didukung oleh PostgreSQL sebagai basis data [7, 8]. Dalam prosesnya, dilakukan analisis kebutuhan untuk memahami alur bisnis dan kebutuhan sistem, kemudian perancangan antarmuka (UI/UX) agar alur penggunaan aplikasi sesuai dengan proses bisnis yang berjalan.

Pada tahap implementasi, dilakukan pengembangan komponen dan halaman pada frontend, termasuk validasi input, pengolahan state/data, serta integrasi dengan API. Di sisi backend, dilakukan pengembangan layanan dan endpoint API, implementasi logika bisnis, serta integrasi dan pengelolaan data pada basis data sesuai skema yang dirancang. Selain itu, dilakukan pengujian fitur, debugging, dan

perbaikan kesalahan untuk memastikan fungsionalitas berjalan sesuai kebutuhan pengguna, serta penyusunan dokumentasi teknis dan laporan magang sebagai bagian dari pertanggungjawaban kegiatan.

3.3 Uraian Pelaksanaan Magang

Pelaksanaan magang dilakukan secara bertahap dimulai dari orientasi pekerjaan dan pemahaman alur kerja pengembangan yang digunakan di perusahaan. Tahap berikutnya berfokus pada pemahaman kebutuhan sistem dan pembagian tugas yang dikoordinasikan bersama supervisor. Setelah kebutuhan dipahami, kegiatan dilanjutkan dengan implementasi fitur pada frontend menggunakan Angular dan pengembangan layanan backend menggunakan Spring Boot, termasuk proses integrasi keduanya melalui API [8]. Selama proses pengembangan, dilakukan pengujian untuk memastikan fungsi berjalan sesuai kebutuhan, kemudian dilakukan perbaikan jika ditemukan kendala atau ketidaksesuaian. Pada akhir setiap tahap pekerjaan, dilakukan pengecekan hasil dan pembahasan lanjutan untuk menentukan perbaikan maupun pengembangan fitur berikutnya.

Pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

Tabel 3.1. Pekerjaan Mingguan Selama Magang

Minggu Ke -	**Pekerjaan yang Dilakukan**
1	Pengenalan teknologi yang digunakan, khususnya JavaScript, TypeScript, dan framework Angular. Selain itu dilakukan eksplorasi library PrimeNG untuk membangun antarmuka pengguna, termasuk pembuatan tabel User dan User Role, dialog detail user, serta pengaturan struktur CSS agar tetap terpisah dari global styling.
2	Dilakukan penyelesaian fitur CRUD untuk User dan Role menggunakan komponen dialog PrimeNG. Selain itu mulai dipelajari penerapan state management menggunakan NGXS, termasuk pengaturan struktur folder, pembuatan service, dan integrasi state ke komponen frontend seperti Data Retention dan Application Log.
Lanjutan di halaman berikutnya...	

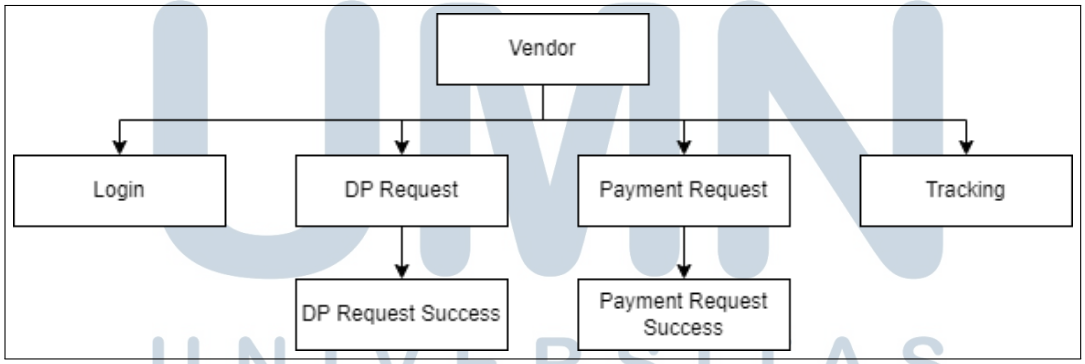
Minggu Ke -	Pekerjaan yang Dilakukan
3	Inisialisasi proyek utama JEBRET. Kegiatan meliputi setup Angular, PrimeNG, dan Tailwind CSS, konfigurasi routing, pembuatan tampilan login vendor, layout vendor, sidebar, navbar, serta pengembangan awal fitur DP Request dengan dialog dinamis yang reusable.
4	Dilakukan standardisasi tampilan vendor sesuai guideline perusahaan. Selain itu dikembangkan fitur Payment yang terintegrasi dengan DP Request, upload file dengan mekanisme drag and drop, pembuatan PDF menggunakan jsPDF, fitur tracking status vendor, serta inisialisasi tampilan dan request flow untuk Admin.
5	Perbaikan layout Admin dan Vendor, penambahan notifikasi pada sidebar, serta fitur upload data Admin. Selain itu dilakukan inisialisasi backend menggunakan Spring Boot, perbaikan dependency, pembuatan REST API awal, integrasi Swagger, serta perancangan template response backend.
6	Dikembangkan backend untuk fitur inti bisnis, yaitu DP Request dan Tracking Vendor. Kegiatan mencakup pembuatan endpoint CREATE dan READ DP Request, penanganan error enumerasi, implementasi pagination, serta inisialisasi endpoint Payment dan Tracking di sisi backend.
7	Dilakukan pembuatan endpoint Admin Request, implementasi autentikasi menggunakan JWT stateless, pengujian login vendor, refaktor endpoint vendor, serta transisi dari JWT stateless ke JWT stateful untuk Vendor dan Admin, termasuk konfigurasi CORS.
8	Dilakukan kegiatan bimbingan serta pengembangan fitur approval flow. Kegiatan meliputi implementasi login user (Admin, Finance, Accounting), autentikasi JWT multi-role, pembuatan fitur Comments untuk status approval, DELETE Billing Request, serta pengembangan fitur Upload Bupot File.
	Lanjutan di halaman berikutnya...

Minggu Ke -	Pekerjaan yang Dilakukan
9	Pengembangan modul administratif dan integrasi frontend–backend. Kegiatan mencakup pembuatan Admin Report, User Management, Vendor Management, integrasi backend Spring Boot ke Angular, implementasi login terintegrasi, HTTP Interceptor untuk JWT, serta Route Guard untuk proteksi halaman.
10	Dilakukan integrasi penuh fitur Vendor. Kegiatan meliputi penyempurnaan Guard, integrasi login, integrasi DP Request beserta pagination dan proses CREATE, pembuatan halaman DP Request Success, integrasi Payment, serta penyelesaian integrasi fitur Tracking.
11	Difokuskan pada penyempurnaan alur Admin. Kegiatan meliputi finalisasi integrasi Tracking, pembuatan dan integrasi login Admin, perbaikan Guard Admin, integrasi Request Admin, penambahan validasi user di backend, serta penambahan parameter filter pada Admin Billing Request.
12	Dilakukan penguatan modul upload dan attachment. Kegiatan meliputi integrasi Request Admin, implementasi fitur download attachment di backend, integrasi Admin Dialog, pengelolaan Upload Attachment dan Bupot File, serta pembuatan service request dan attachment untuk mendukung integrasi frontend–backend.
13	Finalisasi dan optimasi sistem. Kegiatan mencakup penyelesaian service request dan attachment, pengembangan Vendor dan User Management, integrasi filter tanggal untuk report, pembuatan service download Bupot, implementasi notifikasi real-time menggunakan WebSocket, serta perbaikan autentikasi.
14	Dilakukan integrasi lanjutan fitur real-time dan manajemen data. Kegiatan meliputi integrasi WebSocket notification, pengembangan dan integrasi Vendor Management, perbaikan validasi Guard, serta inisialisasi CRUD SAP sebagai data pendukung proses bisnis.
	Lanjutan di halaman berikutnya...

Minggu Ke -	Pekerjaan yang Dilakukan
15	Penyelesaian dan validasi sistem. Kegiatan meliputi pengembangan dan validasi CRUD SAP, integrasi validasi SAP ke Vendor, penambahan parameter User, serta pengujian akhir untuk memastikan sistem berjalan stabil dan sesuai kebutuhan bisnis.
16	Dilakukan revisi dan penyempurnaan antarmuka sistem pada sisi Vendor, meliputi tampilan login, sidebar, modal, serta tracking yang diubah menjadi komponen progress timeline. Selain itu dilakukan penyesuaian tampilan tabel dan paginasi sesuai standar antarmuka perusahaan, pengembangan fitur pencarian dengan dialog, serta perbaikan service Vendor dan integrasi modul User Management.

3.3.1 SiteMap

Sitemap Vendor (3.1) menggambarkan struktur navigasi yang dapat diakses oleh pengguna dengan peran Vendor setelah berhasil melakukan proses login. Pada bagian paling atas, terdapat halaman Login yang berfungsi sebagai gerbang awal autentikasi sebelum pengguna dapat mengakses fitur utama sistem.

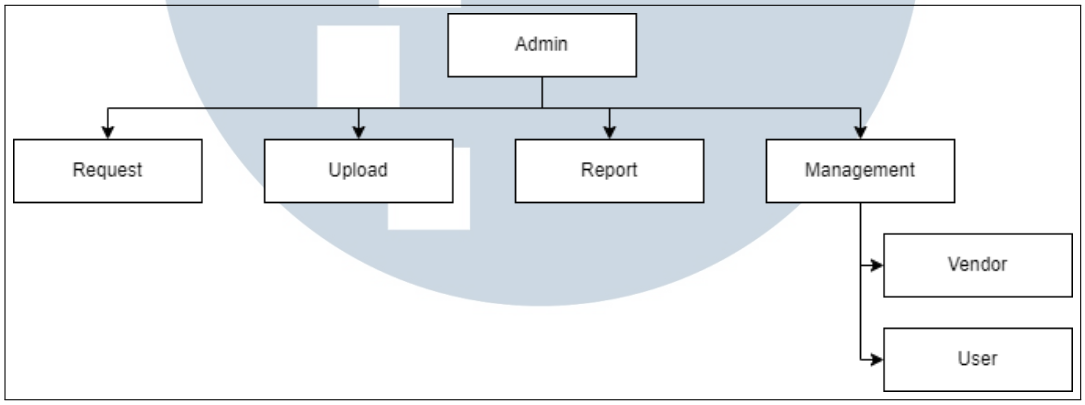


Gambar 3.1. SiteMap Vendor

Setelah berhasil login, Vendor akan diarahkan ke halaman utama yang menyediakan beberapa menu fungsional. Menu DP Request digunakan oleh Vendor untuk mengajukan permintaan uang muka (down payment) kepada pihak Admin. Setelah pengajuan berhasil diproses dan dikirimkan ke sistem, Vendor akan diarahkan ke halaman DP Request Success sebagai indikator bahwa permintaan telah berhasil diajukan.

Selain itu, terdapat menu Payment Request yang berfungsi untuk mengajukan permintaan pembayaran. Melalui menu ini, Vendor dapat mengirimkan dokumen serta data pendukung pembayaran. Setelah permintaan pembayaran berhasil dikirim, sistem akan menampilkan halaman Payment Request Success sebagai konfirmasi bahwa proses pengajuan telah berhasil dilakukan.

Vendor juga dapat mengakses menu Tracking, yang digunakan untuk memantau status permintaan DP maupun Payment Request yang telah diajukan. Melalui halaman ini, Vendor dapat mengetahui perkembangan proses verifikasi dan persetujuan yang dilakukan oleh Admin. Struktur sitemap Vendor dirancang agar alur pengajuan dan pemantauan status dapat dilakukan secara sederhana, jelas, dan mudah dipahami.



Gambar 3.2. SiteMap Admin

Sitemap Admin menggambarkan struktur navigasi yang dapat diakses oleh pengguna dengan peran Admin setelah melakukan login ke dalam sistem. Pada level utama, Admin memiliki akses ke beberapa menu inti yang mendukung proses pengelolaan dan verifikasi data.

Menu Request berfungsi sebagai pusat pengelolaan permintaan yang masuk dari Vendor, baik permintaan DP maupun Payment Request. Melalui halaman ini, Admin dapat meninjau detail permintaan dan melakukan proses verifikasi sesuai dengan prosedur yang berlaku.

Menu Upload digunakan untuk mengunggah dokumen pendukung yang berkaitan dengan proses administrasi dan pembayaran. Fitur ini mendukung kelengkapan data serta memastikan setiap proses terdokumentasi dengan baik.

Selanjutnya, menu Report menyediakan akses ke halaman laporan yang berisi rekapitulasi data transaksi dan aktivitas sistem. Halaman ini membantu Admin dalam memantau dan mengevaluasi proses yang berjalan secara

keseluruhan.

Menu Management berfungsi sebagai pusat pengelolaan data sistem. Di dalam menu ini, Admin dapat mengakses fitur Vendor Management untuk mengelola data Vendor serta User Management untuk mengatur data pengguna sistem. Struktur ini mempermudah Admin melakukan pengelolaan sistem secara terpusat dan efisien.

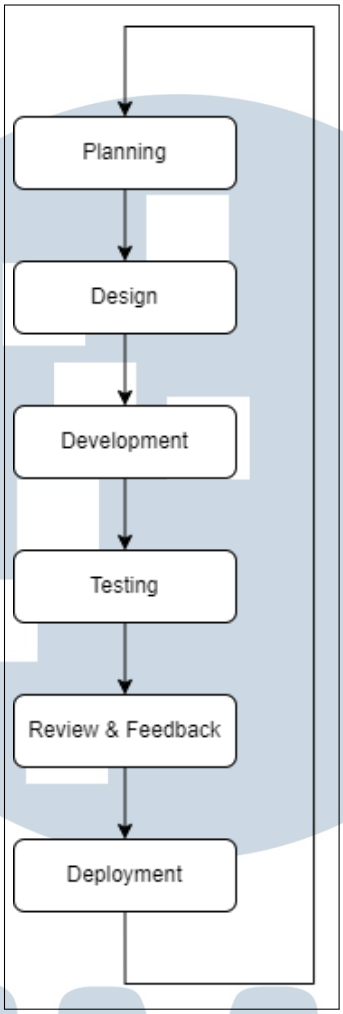
Sitemap Admin dirancang untuk mendukung alur kerja administratif yang sistematis, mulai dari pengelolaan permintaan, pengunggahan dokumen, pelaporan, hingga manajemen pengguna dan vendor.

3.3.2 Metode Pengembangan Sistem

Dalam proyek JEBRET (Jalin Electronic Billing and Realtime Tracking), perusahaan menerapkan pendekatan pengembangan *Agile Software Development Lifecycle* (SDLC) dengan kerangka kerja Scrum [9].

Model Agile dipilih karena proses pengembangan dilakukan secara bertahap dan berulang (iteratif), serta melibatkan komunikasi dan review rutin antara pengembang dan pembimbing proyek. Setiap fitur dikerjakan dalam siklus pengembangan singkat yang disebut Sprint, di mana hasilnya langsung dievaluasi untuk menentukan prioritas pengerjaan berikutnya.

Model Agile merupakan pengembangan dilakukan secara berulang (iterative). Setiap siklus menghasilkan increment produk baru yang siap diuji dan dikembangkan lebih lanjut berdasarkan masukan pembimbing.

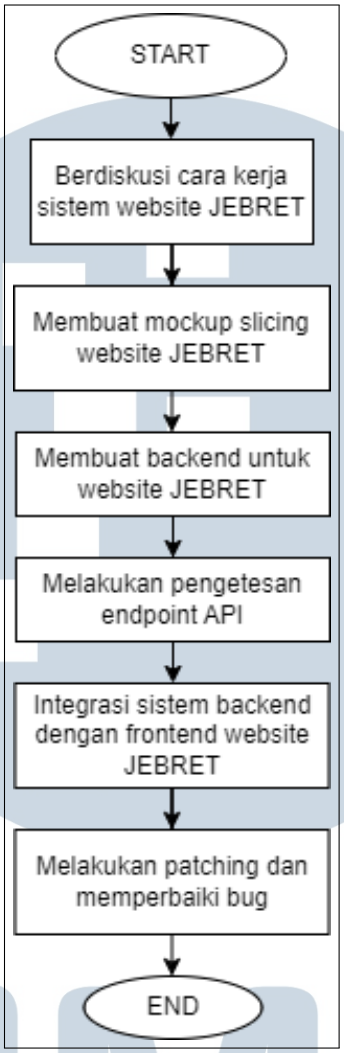


Gambar 3.3. Alur model *Agile Software Development Lifecycle* (SDLC)

Model Agile pada gambar 3.3 yaitu alur pengembangan yang dilakukan secara berulang (iteratif). Setiap siklus menghasilkan *increment* produk baru yang siap diuji dan dikembangkan lebih lanjut berdasarkan masukan pembimbing.

3.3.3 Flow Proses Rancang Bangun Website JEBRET

Flow pembangunan website JEBRET diperlukan untuk memastikan proses pengembangan sistem berjalan secara terstruktur, sistematis, dan terkontrol, sehingga setiap tahapan pengembangan dapat dilakukan secara efisien dan selaras dengan kebutuhan perusahaan.



Gambar 3.4. Flow proses rancang bangun website JEBRET

Gambar 3.4 Flow proses pengembangan website JEBRET menggambarkan alur pengembangan sistem yang dilakukan secara bertahap dan berurutan selama pelaksanaan magang di PT Hexaon Business Mitrasindo. Flow ini menunjukkan proses pengembangan website JEBRET yang dimulai dari tahap perencanaan hingga tahap penyelesaian sistem.

Proses pengembangan website JEBRET diawali dengan tahap diskusi mengenai cara kerja sistem. Pada tahap ini dilakukan pembahasan bersama tim terkait alur bisnis, kebutuhan pengguna, serta fitur-fitur utama yang akan diterapkan pada website JEBRET agar sesuai dengan kebutuhan perusahaan.

Setelah kebutuhan sistem dipahami, proses dilanjutkan dengan pembuatan mockup dan slicing website JEBRET. Mockup digunakan sebagai gambaran awal antarmuka sistem, sedangkan slicing dilakukan untuk mengimplementasikan desain

tersebut ke dalam bentuk frontend sebagai dasar pengembangan sistem.

Tahap berikutnya adalah pembuatan backend website JEBRET. Pada tahap ini dilakukan pengembangan logika bisnis sistem, pengelolaan data, serta pembuatan endpoint API yang mendukung fungsi-fungsi utama website JEBRET.

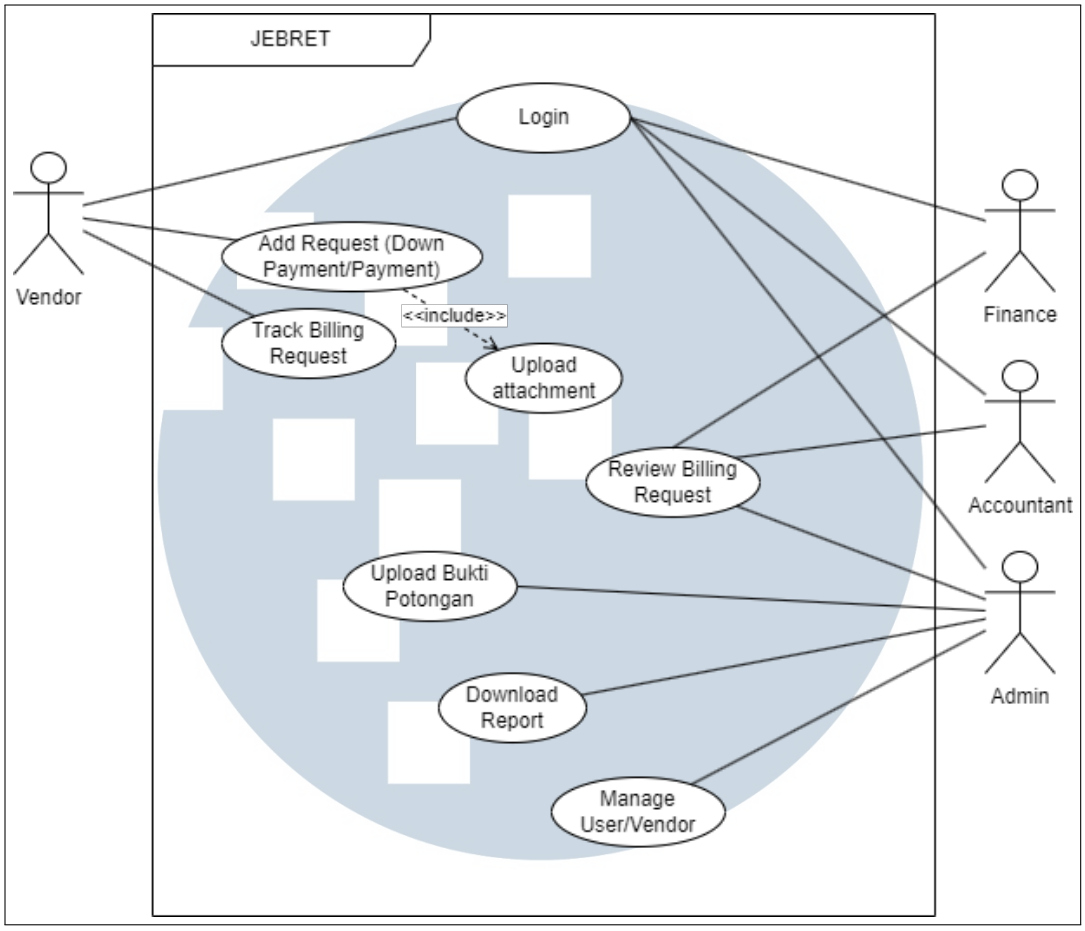
Setelah backend selesai dikembangkan, dilakukan pengujian terhadap endpoint API untuk memastikan setiap fungsi berjalan dengan baik dan sesuai dengan kebutuhan sistem sebelum dilakukan integrasi secara menyeluruh.

Selanjutnya dilakukan integrasi sistem backend dengan frontend website JEBRET. Tahap ini memastikan seluruh fitur yang telah dikembangkan dapat berjalan secara terintegrasi melalui antarmuka website.

Tahap akhir dalam proses pengembangan website JEBRET adalah melakukan patching dan perbaikan bug yang ditemukan selama proses pengujian dan integrasi. Setelah seluruh perbaikan dilakukan dan sistem berjalan dengan stabil, maka proses pengembangan website JEBRET dinyatakan selesai.

3.3.4 Use Case Diagram

Use Case Diagram digunakan untuk menggambarkan interaksi antara aktor dengan sistem JEBRET dalam menjalankan fungsi-fungsi utama yang tersedia. Diagram ini membantu memvisualisasikan kebutuhan fungsional sistem serta peran masing-masing aktor, sehingga alur penggunaan sistem dapat dipahami secara jelas sesuai dengan proses bisnis yang berjalan.



Gambar 3.5. Use Case Diagram JEBRET

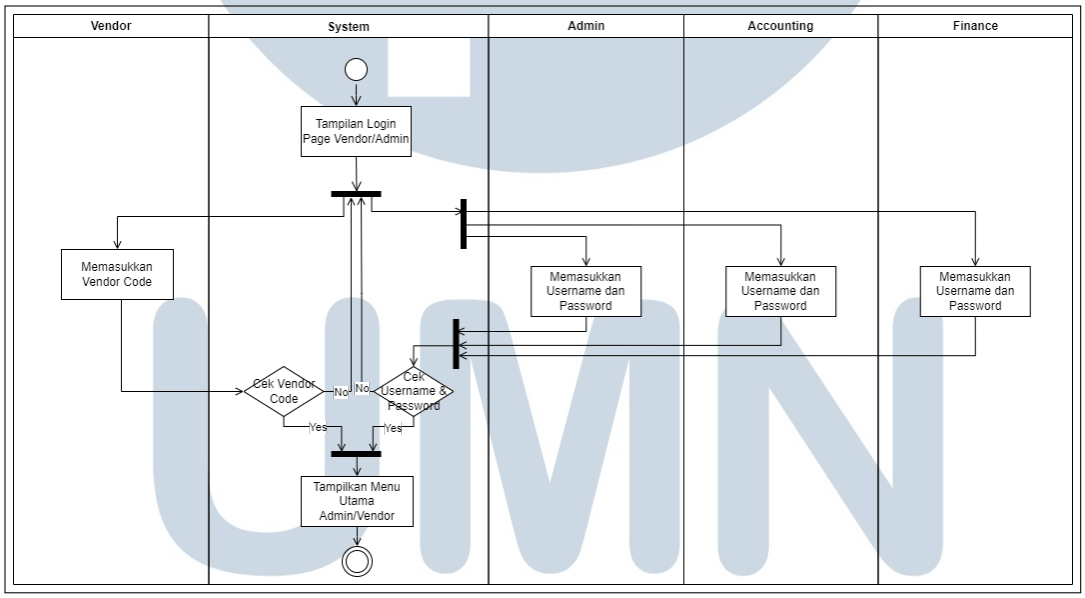
Pada Gambar 3.5, aktor Vendor dapat melakukan proses Login untuk mengakses sistem JEBRET. Setelah berhasil masuk, Vendor dapat melakukan Add Request yang mencakup pengajuan Down Payment Request dan Payment Request. Dalam proses pengajuan tersebut, sistem mewajibkan Vendor untuk melakukan Upload Attachment sebagai dokumen pendukung pengajuan. Selain melakukan pengajuan, Vendor juga dapat mengakses fitur Track Billing Request untuk memantau status dan perkembangan request yang telah diajukan secara real-time.

Di sisi internal perusahaan, aktor Accounting, Finance, dan Admin memiliki akses untuk melakukan Review Billing Request terhadap pengajuan yang masuk sesuai dengan tahapan verifikasi dan kewenangan masing-masing. Aktor Admin juga memiliki hak akses tambahan untuk melakukan Upload Bukti Potongan sebagai bagian dari kelengkapan administrasi pembayaran. Selain itu, sistem menyediakan fitur Download Report yang digunakan untuk keperluan pelaporan

dan dokumentasi data billing. Pada aspek pengelolaan sistem, aktor Admin dapat melakukan Manage User/Vendor untuk mengelola data pengguna dan vendor yang terdaftar dalam sistem. Seluruh aktor yang telah login dapat mengakhiri sesi penggunaan sistem setelah seluruh aktivitas selesai dilakukan.

3.3.5 Login Activity Diagram

Pada Gambar 3.6, Activity Diagram Login menggambarkan alur autentikasi pengguna pada sistem JEBRET. Proses dimulai saat sistem menampilkan halaman login. Aktor Vendor memasukkan Vendor Code, sedangkan aktor Admin, Accounting, dan Finance memasukkan Username dan Password. Sistem kemudian melakukan validasi data login. Jika data tidak sesuai, pengguna dikembalikan ke halaman login. Jika validasi berhasil, sistem menampilkan Menu Utama sesuai dengan peran pengguna dan proses login dinyatakan selesai.



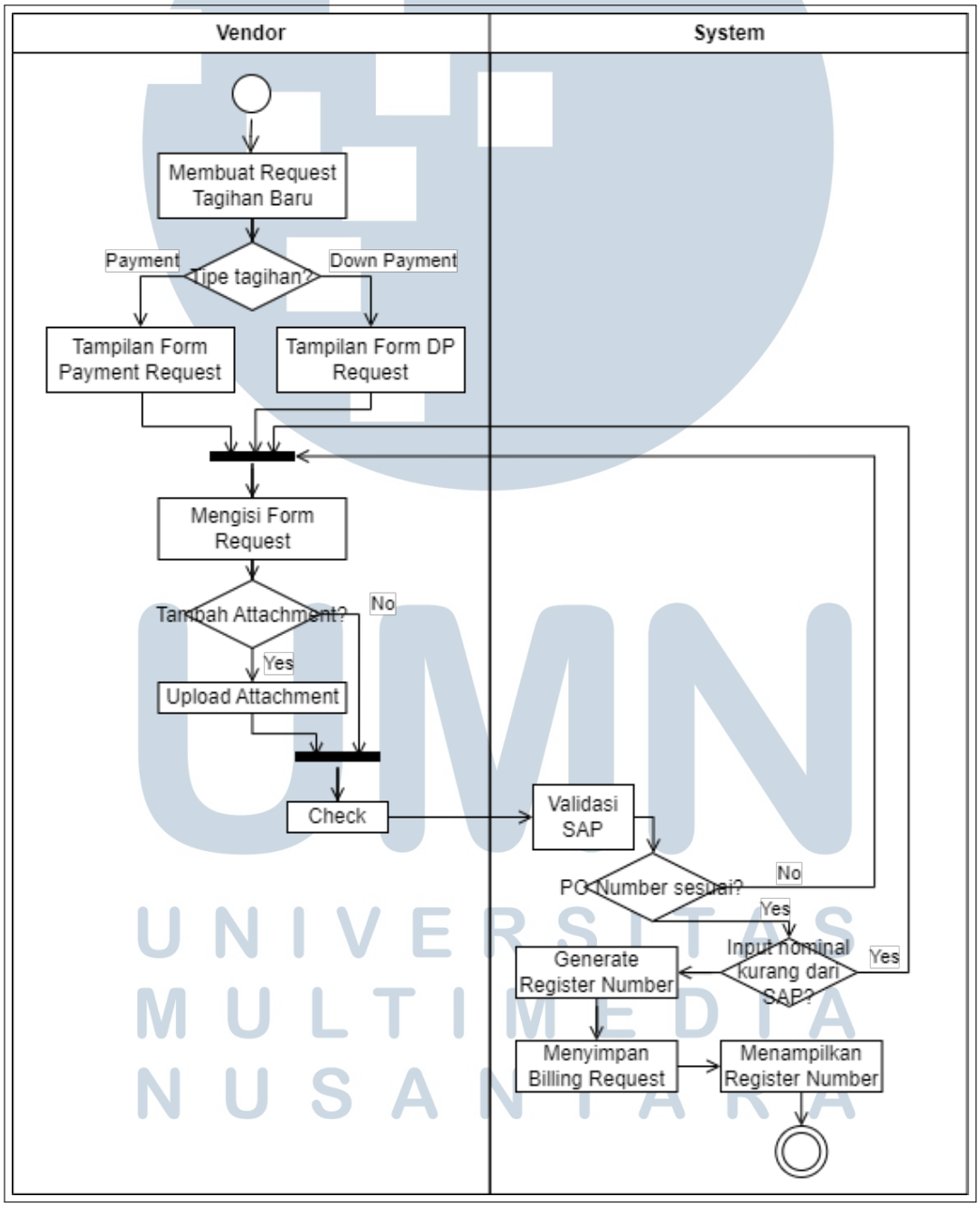
Gambar 3.6. Login Activity Diagram

3.3.6 Add Request Activity Diagram

Pada Gambar 3.7, Activity Diagram Add Request menggambarkan alur pengajuan tagihan oleh aktor Vendor. Proses dimulai ketika Vendor membuat request tagihan baru dan memilih jenis tagihan, yaitu Down Payment atau Payment. Sistem kemudian menampilkan form sesuai dengan jenis request yang dipilih.

Vendor mengisi form pengajuan dan melakukan Upload Attachment sebagai dokumen pendukung.

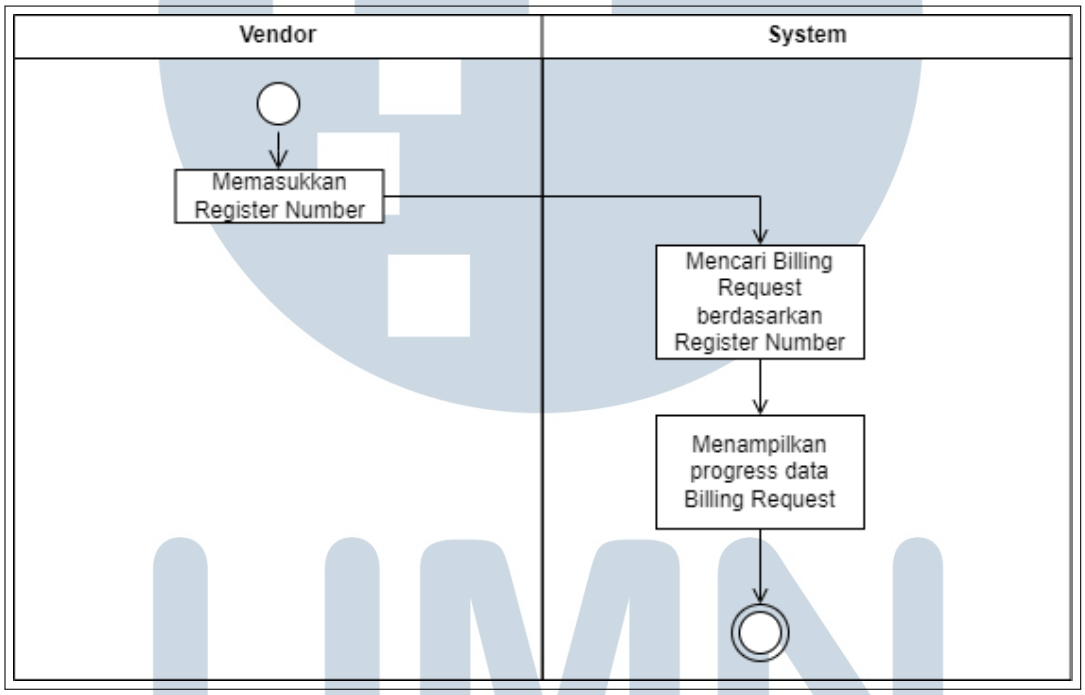
Setelah form dikirim, sistem melakukan proses pengecekan dan validasi data SAP. Apabila data tidak sesuai, proses pengajuan dihentikan. Jika validasi berhasil, sistem melakukan generate register number, menyimpan data billing request, dan menampilkan register number sebagai tanda bahwa pengajuan berhasil dilakukan.



Gambar 3.7. Add Request Activity Diagram

3.3.7 Tracking Activity Diagram

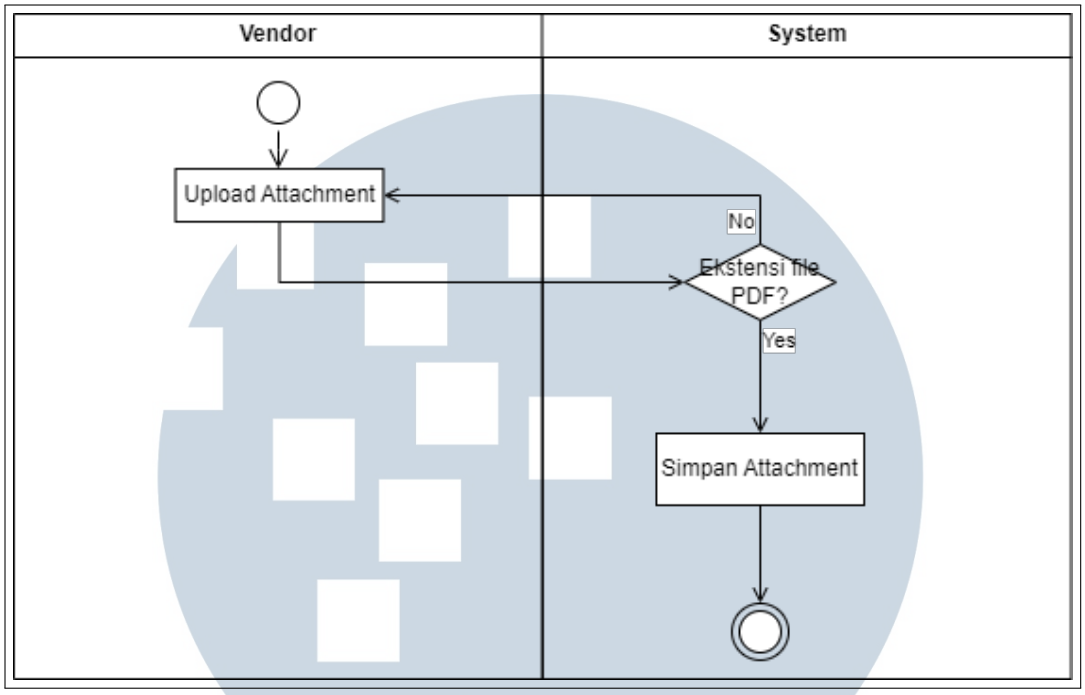
Pada Gambar 3.8, Activity Diagram Track Billing Request menggambarkan alur pemantauan status tagihan oleh aktor Vendor. Proses dimulai ketika Vendor memasukkan Register Number pada sistem. Selanjutnya, sistem mencari data Billing Request berdasarkan register number yang dimasukkan. Setelah data ditemukan, sistem menampilkan progress dan status Billing Request kepada Vendor sebagai informasi perkembangan pengajuan. Proses berakhir setelah data billing request berhasil ditampilkan.



Gambar 3.8. Tracking Billing Request Activity Diagram

3.3.8 Upload Attachment Activity Diagram

Pada Gambar 3.9, Activity Diagram Upload Attachment menggambarkan alur proses pengunggahan dokumen pendukung oleh aktor Vendor. Proses dimulai ketika Vendor melakukan Upload Attachment ke dalam sistem. Selanjutnya, sistem melakukan pengecekan terhadap ekstensi file yang diunggah. Apabila file bukan berformat PDF, proses pengunggahan tidak dilanjutkan. Jika file berformat PDF, sistem menyimpan attachment ke dalam sistem dan proses pengunggahan dinyatakan selesai.

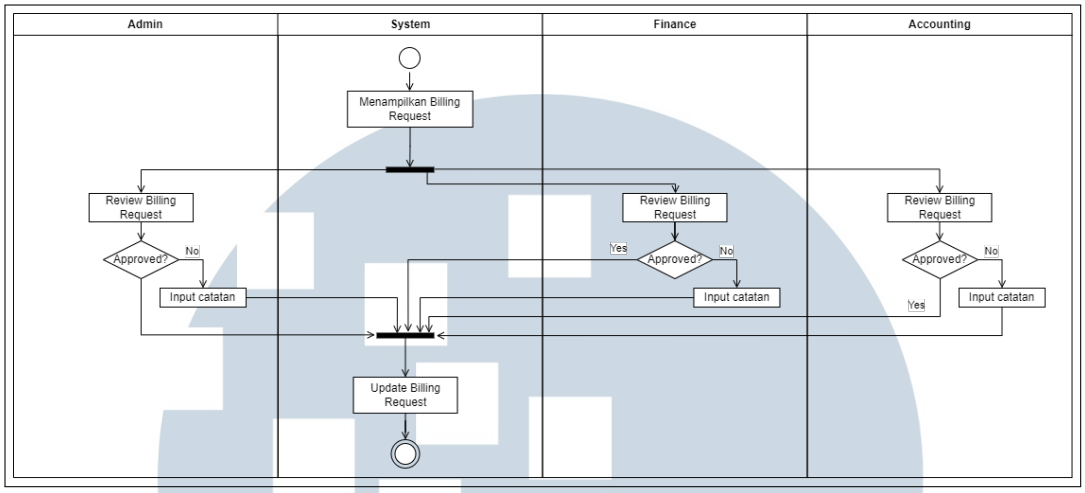


Gambar 3.9. Upload Attachment Activity Diagram

3.3.9 Review Billing Request Activity Diagram

Pada Gambar 3.10, Activity Diagram Review Billing Request menggambarkan alur proses peninjauan tagihan oleh aktor Accounting, Finance, dan Admin. Proses dimulai ketika sistem menampilkan data Billing Request yang masuk. Setiap aktor melakukan Review Billing Request sesuai dengan peran masing-masing.

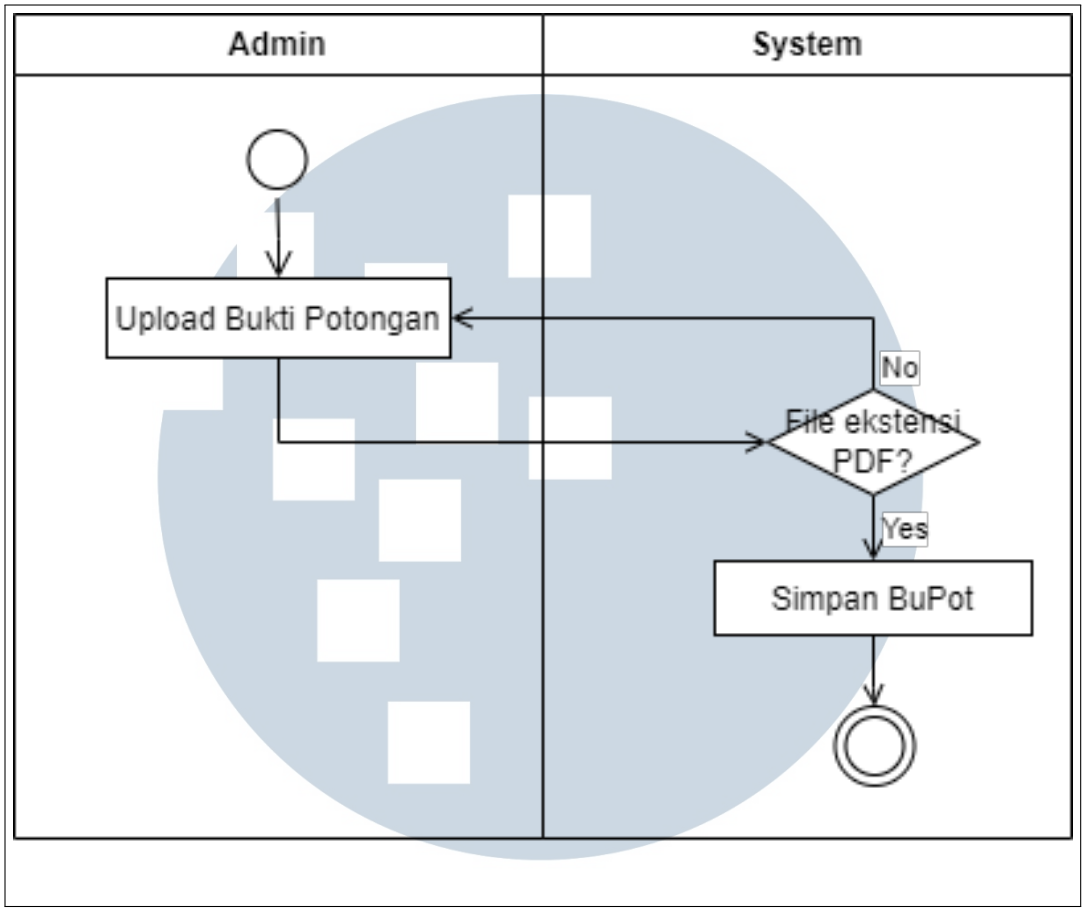
Apabila hasil review tidak disetujui, aktor memberikan catatan sebagai alasan penolakan. Jika disetujui, proses dilanjutkan ke tahap berikutnya hingga seluruh tahapan verifikasi selesai. Sistem kemudian melakukan update Billing Request berdasarkan hasil review dan proses dinyatakan selesai.



Gambar 3.10. Review Billing Request Activity Diagram

3.3.10 Upload Bukti Potongan Activity Diagram

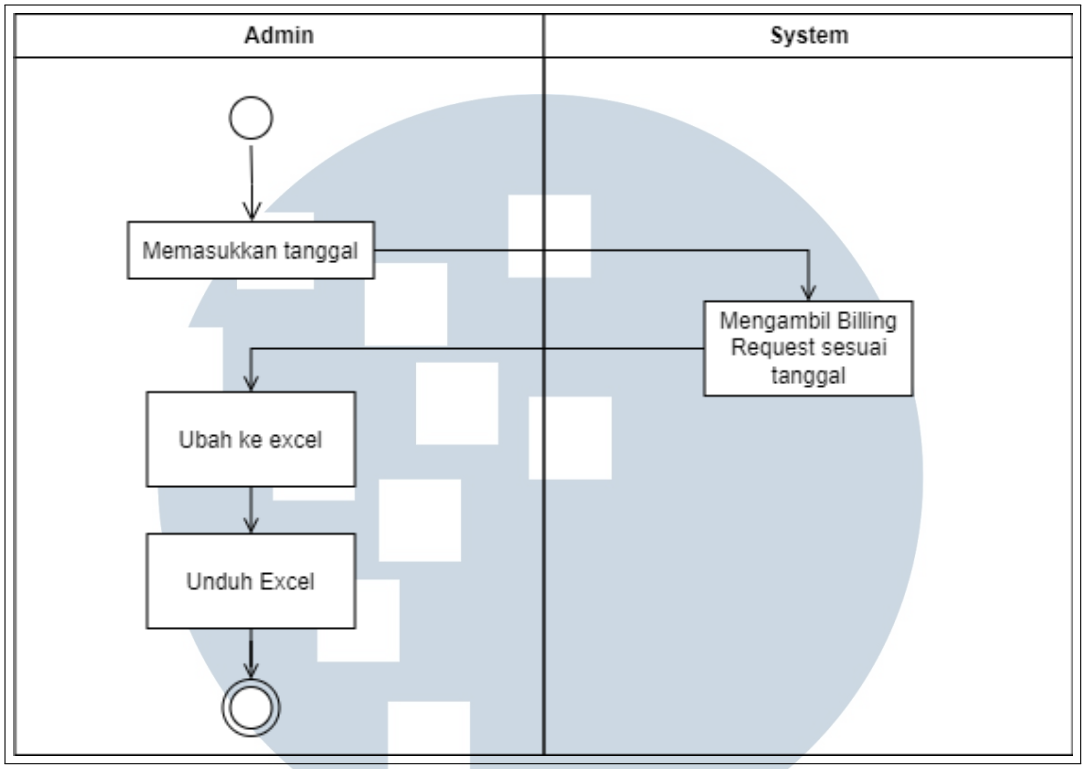
Pada Gambar 3.11, Activity Diagram Upload Bukti Potongan menggambarkan alur pengunggahan dokumen bukti potongan oleh aktor Admin. Proses dimulai ketika Admin melakukan Upload Bukti Potongan ke dalam sistem. Selanjutnya, sistem melakukan pengecekan terhadap ekstensi file yang diunggah. Apabila file tidak berformat PDF, proses pengunggahan tidak dilanjutkan. Jika file berformat PDF, sistem menyimpan Bukti Potongan (BuPot) dan proses dinyatakan selesai.



Gambar 3.11. Upload Bukti Potongan Activity Diagram

3.3.11 Download Report Activity Diagram

Pada Gambar 3.12, Activity Diagram Download Report menggambarkan alur pengunduhan laporan billing oleh aktor Admin. Proses dimulai ketika Admin memasukkan tanggal sebagai parameter pencarian laporan. Sistem kemudian mengambil data Billing Request sesuai dengan tanggal yang ditentukan. Setelah data diperoleh, Admin melakukan proses ubah ke format Excel dan melanjutkan dengan unduh file Excel sebagai laporan billing. Proses berakhir setelah file laporan berhasil diunduh.

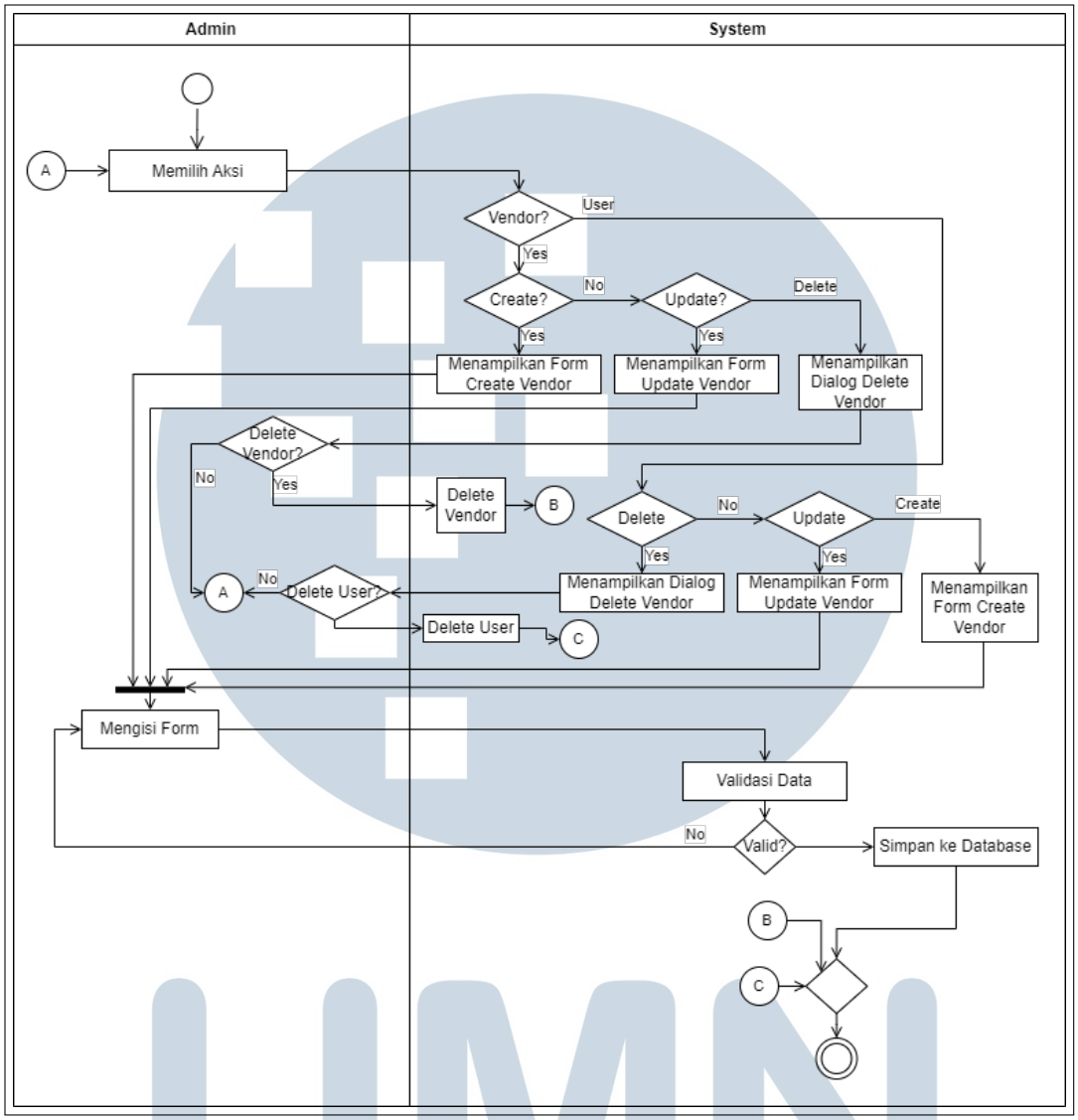


Gambar 3.12. Download Report Activity Diagram

3.3.12 Manage User/Vendor Activity Diagram

Pada Gambar 3.13, Activity Diagram Manage User/Vendor menggambarkan alur pengelolaan data User dan Vendor oleh aktor Admin. Proses dimulai ketika Admin memilih aksi pengelolaan data. Admin menentukan jenis data yang akan dikelola, yaitu User atau Vendor, kemudian memilih aksi Create, Update, atau Delete. Sistem menampilkan form atau dialog sesuai dengan aksi yang dipilih.

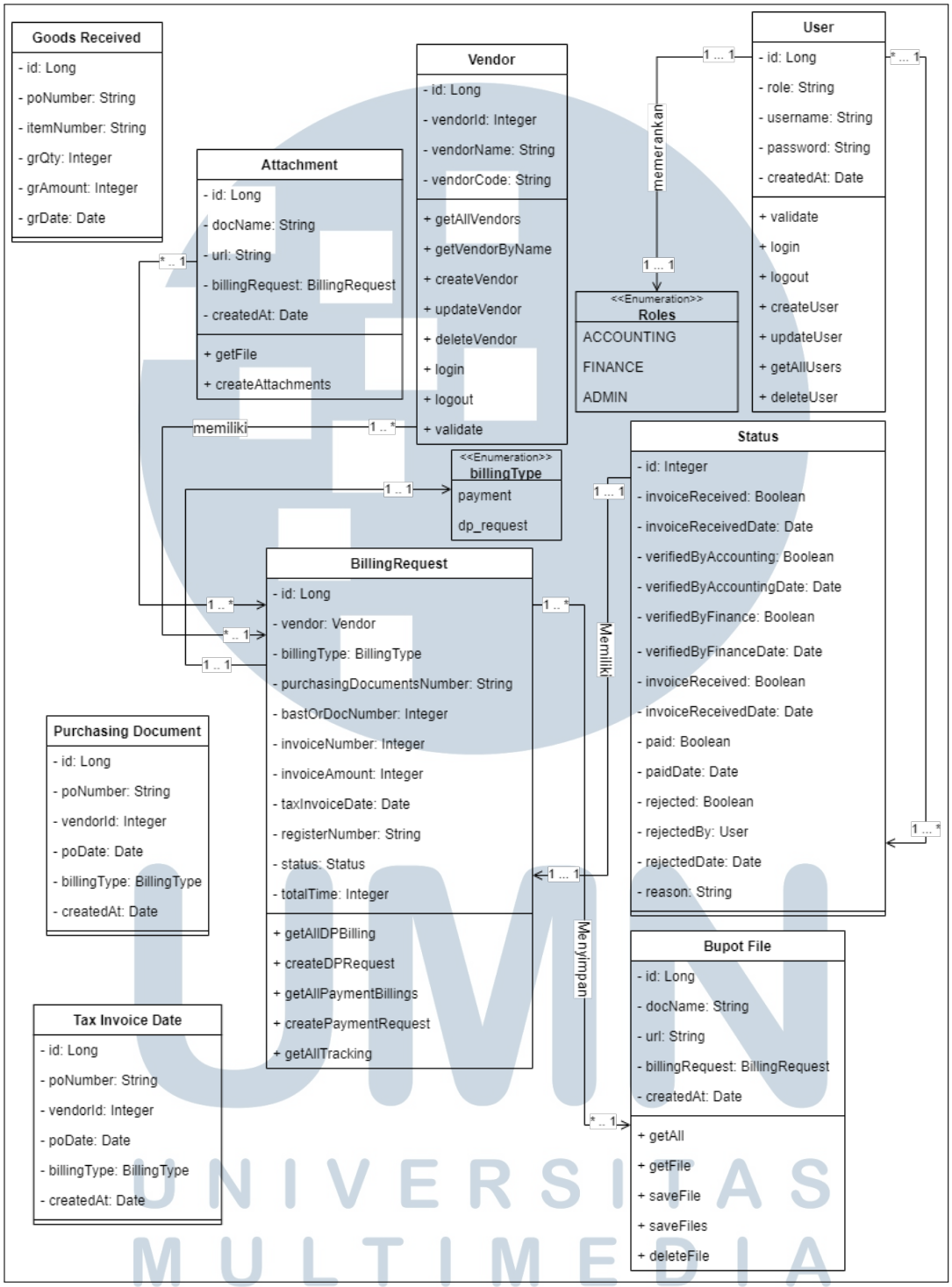
Admin mengisi atau mengubah data pada form yang ditampilkan, kemudian sistem melakukan validasi data. Apabila data tidak valid, Admin diarahkan kembali ke form untuk melakukan perbaikan. Jika data valid, sistem menyimpan perubahan ke dalam database. Pada proses Delete, sistem menampilkan dialog konfirmasi sebelum data dihapus. Proses berakhir setelah aksi pengelolaan data berhasil diselesaikan.



Gambar 3.13. Manage User/Vendor

3.3.13 Class Diagram

Class Diagram digunakan untuk menggambarkan struktur kelas, atribut, serta hubungan antar kelas yang membentuk sistem JEBRET secara keseluruhan. Perancangan Class Diagram diperlukan untuk memberikan gambaran yang jelas mengenai representasi data dan keterkaitan antar entitas dalam sistem, sehingga mendukung proses pengembangan aplikasi yang terstruktur, konsisten, dan mudah dikembangkan.



Gambar 3.14. Class Diagram JEBRET

Class Diagram pada Gambar 3.14 menggambarkan struktur kelas serta hubungan antar entitas utama dalam sistem JEBRET. Kelas BillingRequest berperan sebagai entitas inti yang menyimpan data pengajuan tagihan dan memiliki relasi

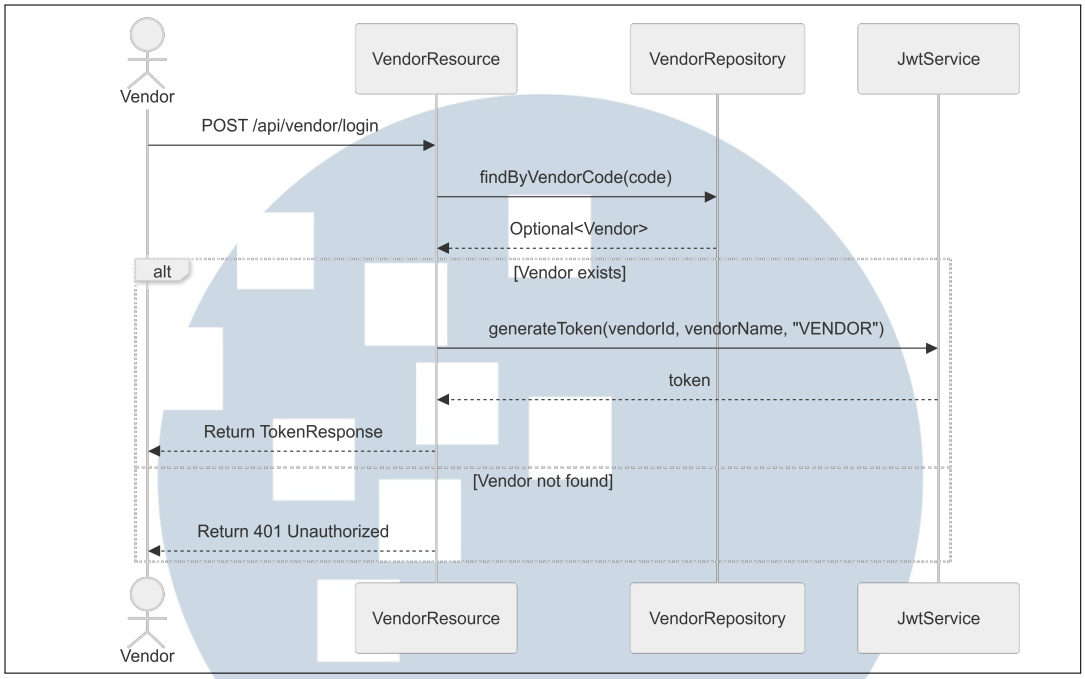
dengan beberapa kelas lain. Setiap BillingRequest terhubung dengan satu Vendor sebagai pihak pengaju, satu Status untuk mencatat tahapan verifikasi, serta satu User yang berperan dalam proses pemeriksaan dan persetujuan. Selain itu, BillingRequest memiliki relasi dengan Attachment sebagai dokumen pendukung pengajuan dan BupotFile sebagai bukti potongan pajak yang diunggah oleh Admin.

Kelas Purchasing Document, Goods Received, dan Tax Invoice Date merepresentasikan data pendukung yang bersumber dari sistem SAP dan digunakan sebagai referensi dalam proses validasi tagihan. Hubungan antar kelas menunjukkan bahwa satu BillingRequest dapat memiliki lebih dari satu data pendukung, sementara setiap data tersebut tetap terikat pada satu BillingRequest. Struktur class diagram ini menggambarkan keterkaitan data yang terorganisir untuk mendukung proses pengajuan, verifikasi, dan dokumentasi billing secara terintegrasi.

3.3.14 Login Sequence Diagram (Vendor)

Pada Gambar 3.15, Sequence Diagram Login Vendor menggambarkan alur autentikasi Vendor pada sistem JEBRET. Proses dimulai ketika aktor Vendor mengirimkan request POST `/api/vendor/login` ke VendorResource dengan menyertakan vendor code. Selanjutnya, VendorResource meneruskan proses dengan memanggil VendorRepository untuk melakukan pencarian data vendor berdasarkan vendor code yang diberikan.

Apabila data vendor ditemukan, VendorResource memanggil JwtService untuk melakukan proses pembuatan token dengan menyertakan informasi vendor dan peran sebagai VENDOR. Token yang dihasilkan kemudian dikembalikan ke VendorResource dan diteruskan kepada Vendor dalam bentuk Token Response. Sebaliknya, apabila data vendor tidak ditemukan, sistem mengembalikan respons 401 Unauthorized sebagai tanda bahwa proses login gagal. Proses berakhir setelah sistem memberikan respons sesuai dengan hasil validasi data vendor.

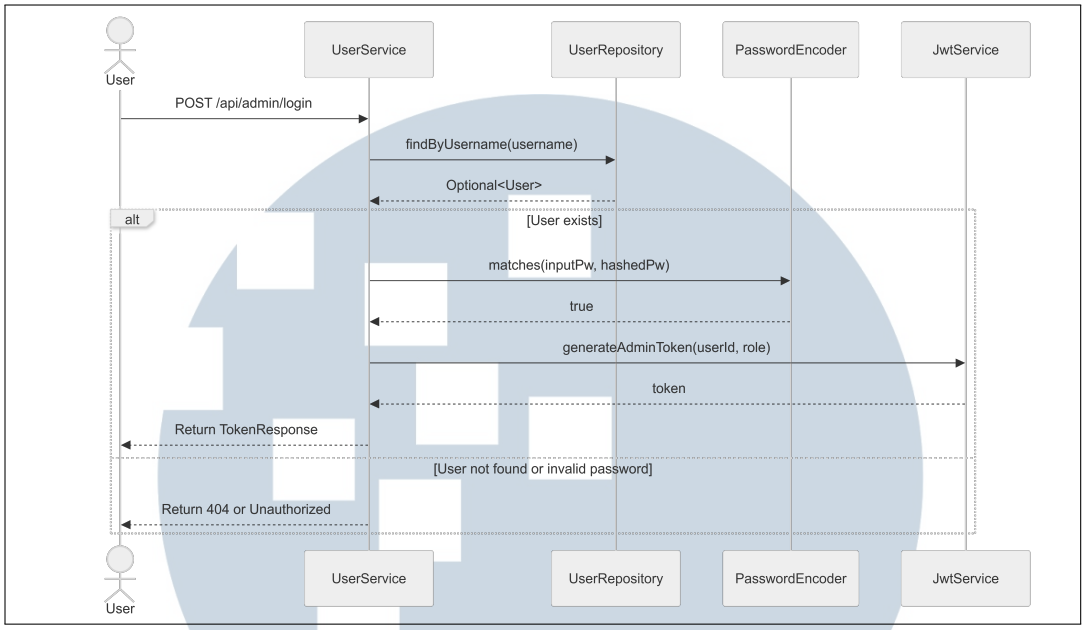


Gambar 3.15. Login Sequence Diagram (Vendor)

3.3.15 Login Sequence Diagram (User)

Pada Gambar 3.16, Sequence Diagram Login User menggambarkan alur autentikasi pengguna internal pada sistem JEBRET, yaitu Admin, Accounting, dan Finance. Proses dimulai ketika aktor User mengirimkan request POST `/api/admin/login` ke UserService dengan menyertakan username dan password. Selanjutnya, UserService memanggil UserRepository untuk mencari data pengguna berdasarkan username yang diberikan.

Apabila data user ditemukan, UserService melakukan verifikasi password dengan memanggil PasswordEncoder untuk mencocokkan password yang diinput dengan password yang tersimpan dalam database. Jika hasil verifikasi valid, UserService memanggil JwtService untuk menghasilkan JWT token berdasarkan userId dan role pengguna. Token yang dihasilkan kemudian dikembalikan kepada User dalam bentuk Token Response. Sebaliknya, apabila data user tidak ditemukan atau password tidak sesuai, sistem mengembalikan respons 404 Not Found atau 401 Unauthorized sebagai tanda bahwa proses login tidak berhasil. Proses berakhir setelah sistem memberikan respons sesuai dengan hasil autentikasi.



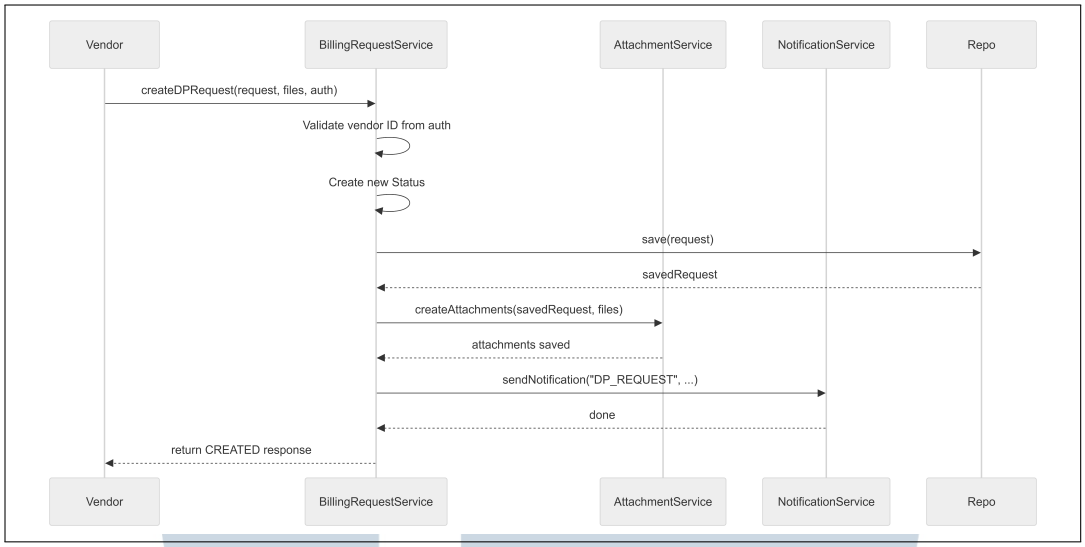
Gambar 3.16. Login Sequence Diagram (User)

3.3.16 Add Request Sequence Diagram

Pada Gambar 3.17, Sequence Diagram Add DP Request menggambarkan alur proses pengajuan Down Payment Request oleh aktor Vendor pada sistem JEBRET. Proses dimulai ketika Vendor mengirimkan request pembuatan DP Request beserta data pengajuan dan file attachment ke BillingRequestService. Selanjutnya, sistem melakukan validasi vendor ID berdasarkan data autentikasi yang digunakan.

Setelah validasi berhasil, BillingRequestService membuat data Status baru sebagai penanda tahapan awal pengajuan, kemudian menyimpan data Billing Request ke dalam database melalui Repository. Setelah data utama tersimpan, sistem memanggil AttachmentService untuk menyimpan seluruh file attachment yang terkait dengan billing request tersebut.

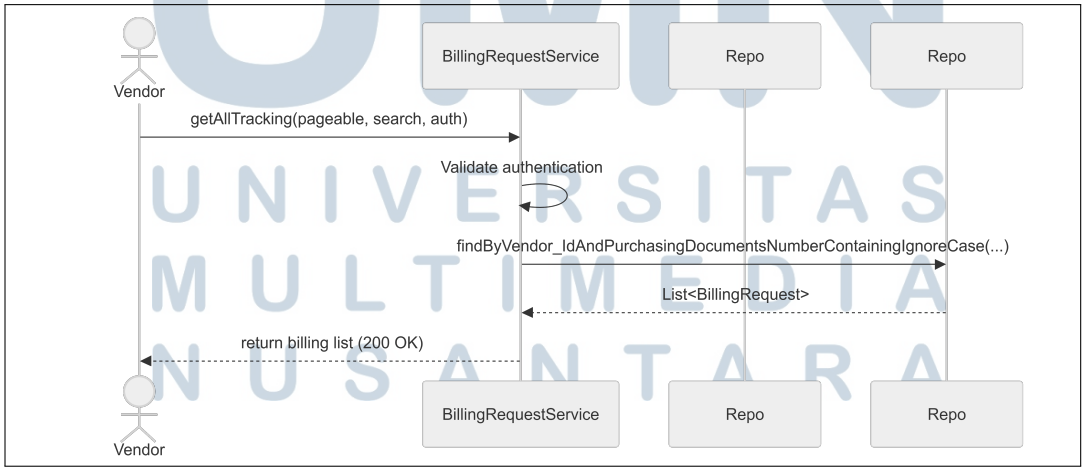
Setelah proses penyimpanan attachment selesai, BillingRequestService memanggil NotificationService untuk mengirimkan notifikasi dengan tipe **DP_REQUEST** sebagai informasi adanya pengajuan baru. Seluruh proses dinyatakan selesai setelah sistem mengembalikan respons CREATED kepada Vendor sebagai tanda bahwa pengajuan DP Request berhasil dilakukan.



Gambar 3.17. Add Request Sequence Diagram

3.3.17 Get Tracking Sequence Diagram

Pada Gambar 3.18, Sequence Diagram Get Tracking Billing Request menggambarkan alur pengambilan data tracking billing oleh Vendor. Vendor mengirimkan request getAllTracking dengan parameter paginasi dan pencarian ke BillingRequestService. Selanjutnya, sistem melakukan validasi autentikasi vendor, kemudian mengambil data Billing Request dari database melalui Repository berdasarkan vendor ID dan kata kunci pencarian. Data billing request yang diperoleh dikembalikan kepada Vendor dalam bentuk daftar billing dengan respons 200 OK.

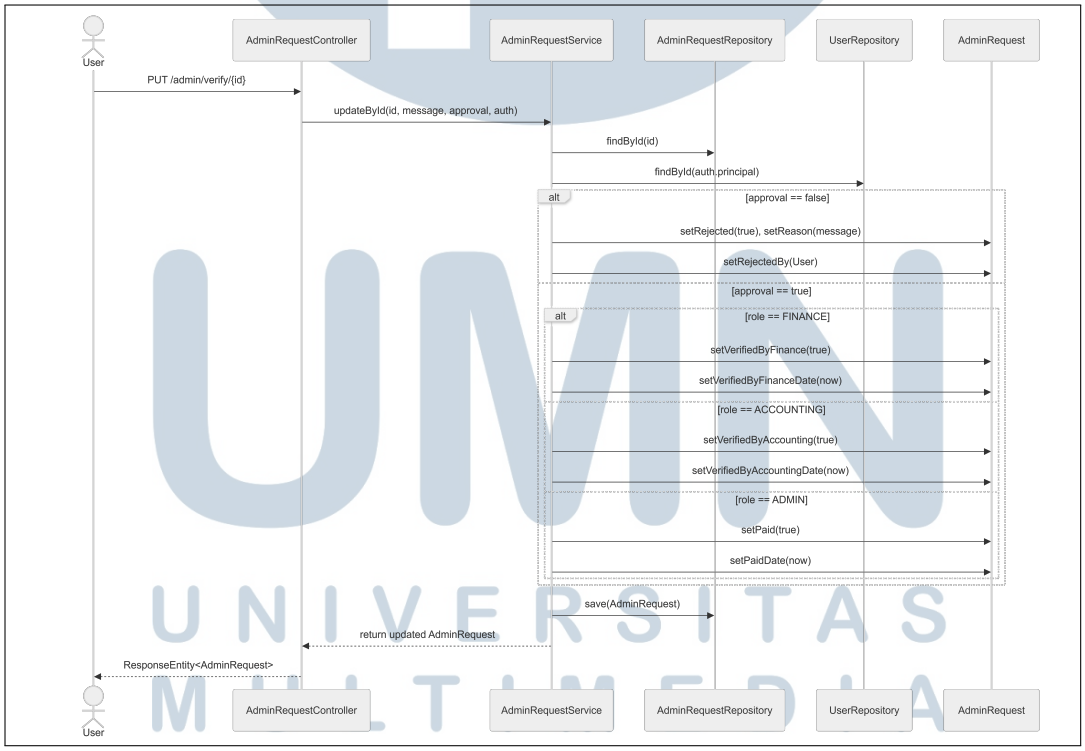


Gambar 3.18. Get Tracking Sequence Diagram

3.3.18 Review Billing Sequence Diagram

Pada Gambar 3.19, Sequence Diagram Review Billing Request menggambarkan alur proses verifikasi dan persetujuan billing oleh aktor Accounting, Finance, dan Admin. Proses dimulai ketika User mengirimkan request PUT `/admin/verify/id` ke AdminRequestController. Request tersebut diteruskan ke AdminRequestService untuk diproses lebih lanjut.

Selanjutnya, sistem mengambil data AdminRequest berdasarkan ID dan data User yang sedang login. Apabila pengajuan tidak disetujui, sistem menetapkan status rejected, menyimpan alasan penolakan, serta mencatat user yang melakukan penolakan. Jika pengajuan disetujui, sistem melakukan pembaruan status sesuai dengan peran user, yaitu *verified by Accounting*, *verified by Finance*, atau *paid* untuk Admin, beserta pencatatan waktu verifikasi. Setelah proses pembaruan selesai, data AdminRequest disimpan ke database dan sistem mengembalikan respons berupa data AdminRequest yang telah diperbarui.

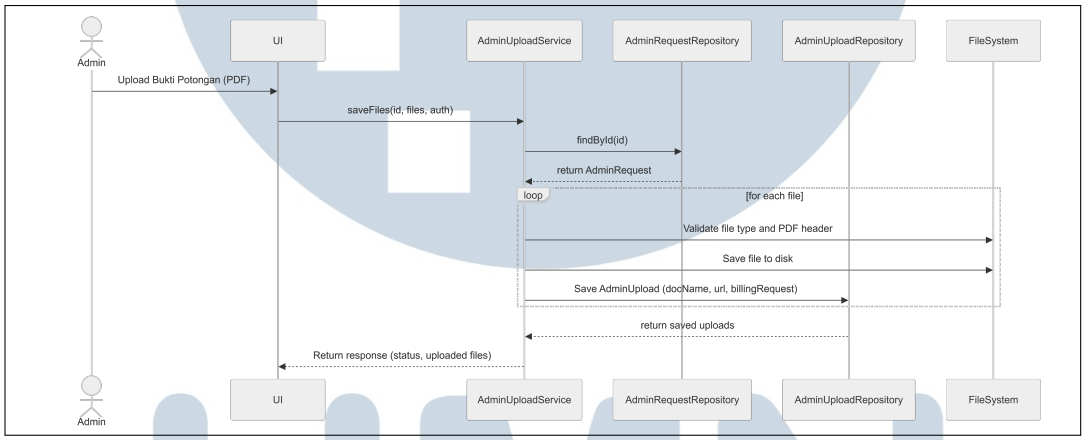


Gambar 3.19. Review Billing Sequence Diagram

3.3.19 Upload Bukti Potongan Squence Diagram

Pada Gambar 3.20, Sequence Diagram Upload Bukti Potongan menggambarkan alur pengunggahan dokumen Bupot oleh aktor Admin. Proses dimulai ketika Admin mengunggah file Bupot berformat PDF melalui UI, yang kemudian diteruskan ke AdminUploadService. Selanjutnya, sistem mengambil data AdminRequest berdasarkan ID yang diberikan.

Untuk setiap file yang diunggah, sistem melakukan validasi tipe file dan header PDF, menyimpan file ke File System, serta mencatat informasi dokumen ke dalam database melalui AdminUploadRepository yang terhubung dengan billing request terkait. Setelah seluruh file diproses, sistem mengembalikan respons berisi status dan daftar file yang berhasil diunggah kepada Admin.



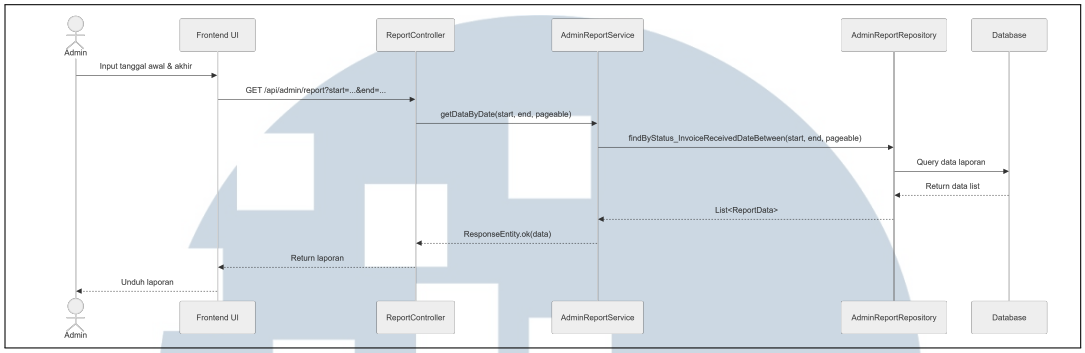
Gambar 3.20. Upload Bukti Potongan Squence Diagram

3.3.20 Download Report Sequence Diagram

Pada Gambar 3.21, Sequence Diagram Download Report menggambarkan alur pengambilan data laporan billing oleh aktor Admin. Proses dimulai ketika Admin memasukkan tanggal awal dan akhir melalui Frontend UI, kemudian sistem mengirimkan request GET `/api/admin/report` ke ReportController. Selanjutnya, ReportController meneruskan permintaan ke AdminReportService untuk mengambil data laporan berdasarkan rentang tanggal yang diberikan.

AdminReportService mengambil data laporan melalui AdminReportRepository dengan melakukan *query* ke database sesuai kriteria tanggal. Data laporan yang diperoleh dikembalikan ke ReportController dalam bentuk *response data*, kemudian diteruskan ke Frontend UI. Setelah data diterima,

Admin melakukan unduh laporan sebagai file laporan billing.



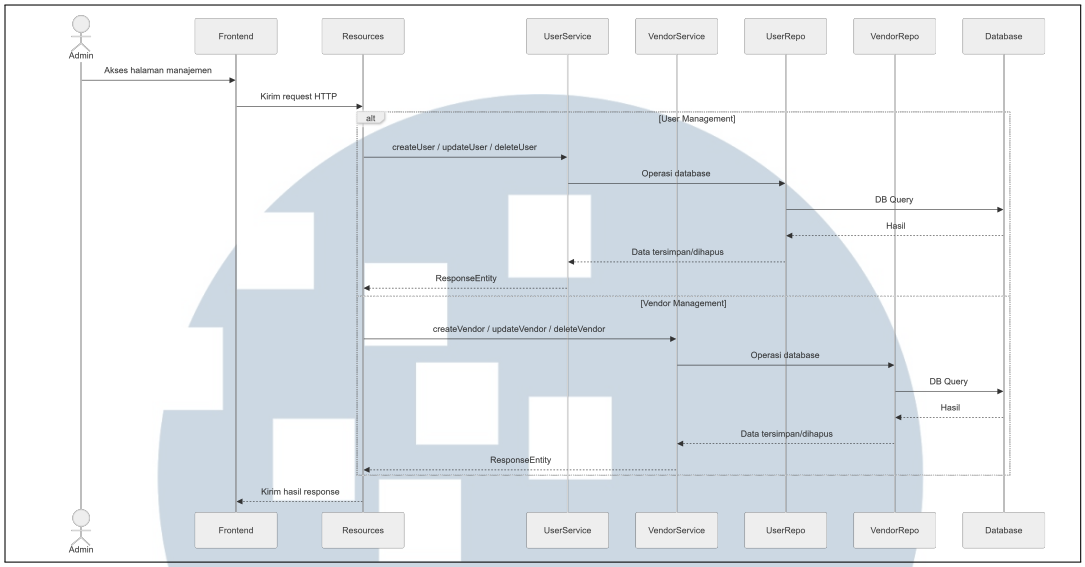
Gambar 3.21. Download Report Sequence Diagram

3.3.21 Manage User/Vendor Sequence Diagram

Pada Gambar 3.22, Sequence Diagram Manage User/Vendor menggambarkan alur pengelolaan data User dan Vendor oleh aktor Admin. Proses dimulai ketika Admin mengakses halaman manajemen melalui Frontend, kemudian frontend mengirimkan request HTTP ke Resources. Berdasarkan jenis pengelolaan yang dipilih, sistem menjalankan proses User Management atau Vendor Management.

Pada proses User Management, request diteruskan ke UserService untuk melakukan aksi create, update, atau delete user. UserService melakukan operasi database melalui UserRepository, kemudian database mengembalikan hasil operasi. Data yang telah disimpan atau dihapus dikembalikan ke Resources dalam bentuk ResponseEntity.

Pada proses Vendor Management, alur serupa dilakukan melalui VendorService dan VendorRepository untuk menjalankan aksi create, update, atau delete vendor. Setelah seluruh proses selesai, sistem mengirimkan hasil respons ke frontend sebagai informasi bahwa pengelolaan data berhasil diselesaikan.



Gambar 3.22. Manage User/Vendor Sequence Diagram

3.3.22 Tampilan Halaman Login Vendor & Admin

Berdasarkan gambar 3.23, desain awal halaman Login Vendor menampilkan struktur dasar antarmuka yang digunakan sebagai pintu masuk mitra ke dalam sistem JEBRET. Pada tahap perancangan ini, fokus utama diarahkan pada penempatan elemen inti seperti logo perusahaan, kolom input Vendor Code, serta tombol Login. Informasi pendukung berupa keterangan bagi vendor yang belum memiliki kode mitra juga disertakan untuk memperjelas alur akses sistem. Elemen visual pada desain ini masih bersifat sederhana dan menekankan keterbacaan serta kejelasan fungsi, sehingga tata letak halaman dan alur interaksi pengguna dapat dipahami dengan baik sebelum masuk ke tahap penyempurnaan desain akhir.

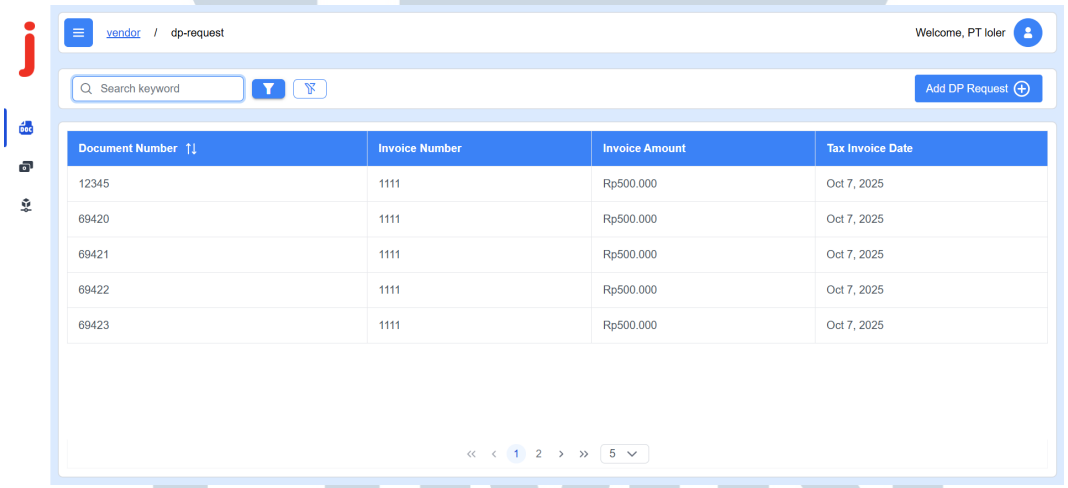
Gambar 3.23. Login Vendor

Berdasarkan gambar 3.24, ditunjukkan tampilan halaman Login Admin yang telah dikembangkan untuk mendukung proses autentikasi pengguna internal dalam mengakses admin dashboard. Halaman ini dirancang dengan struktur yang lebih lengkap, mencakup kolom username, password, serta tombol login yang disusun secara terpusat agar mudah digunakan. Informasi instruksional disertakan untuk memperjelas tujuan halaman, yaitu membatasi akses hanya bagi pengguna yang berwenang. Tampilan akhir ini mengutamakan kemudahan penggunaan, konsistensi identitas visual perusahaan, serta kejelasan alur login guna mendukung keamanan dan efisiensi pengelolaan sistem oleh administrator.

Gambar 3.24. Admin Login

3.3.23 Tampilan Vendor DP Request

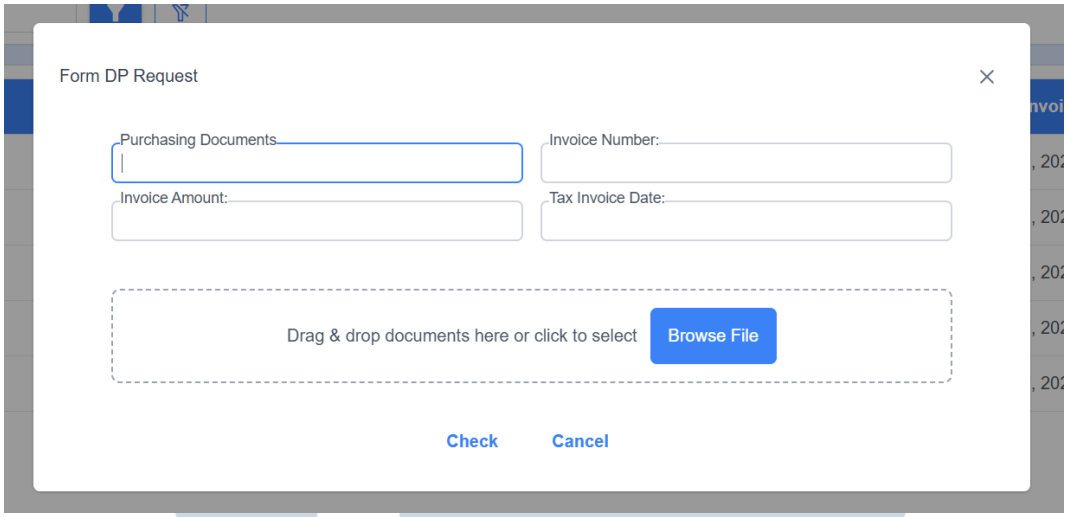
Berdasarkan gambar 3.25, ditunjukkan tampilan halaman yang dirancang untuk mendukung proses pengajuan Down Payment (DP) oleh pihak vendor secara terstruktur. Pada halaman ini, vendor dapat melihat daftar DP Request yang telah diajukan dalam bentuk tabel yang menampilkan informasi utama seperti Document Number, Invoice Number, Invoice Amount, dan Tax Invoice Date. Selain itu, disediakan fitur pencarian dan filter untuk memudahkan vendor dalam menemukan data tertentu berdasarkan kata kunci. Tombol Add DP Request juga disediakan sebagai akses utama bagi vendor untuk menambahkan pengajuan DP baru. Perancangan halaman ini bertujuan untuk memastikan penyajian data DP Request dapat dipahami dengan jelas serta mendukung kemudahan navigasi bagi pengguna.



Gambar 3.25. Vendor DP Request

Selanjutnya, gambar 3.26 menunjukkan tampilan formulir yang digunakan vendor untuk melakukan pengajuan DP secara detail. Formulir ini memuat beberapa field utama seperti Purchasing Documents, Invoice Number, Invoice Amount, dan Tax Invoice Date yang harus diisi oleh vendor sesuai dengan data transaksi. Selain itu, tersedia fitur unggah dokumen dengan metode drag and drop maupun pemilihan file secara langsung, sehingga memudahkan vendor dalam melampirkan dokumen pendukung. Pada bagian bawah formulir disediakan tombol Check untuk melakukan validasi data sebelum diproses serta tombol *Cancel* untuk membatalkan pengajuan. Perancangan formulir ini mengutamakan kejelasan alur pengisian data, kemudahan penggunaan, serta kelengkapan informasi yang

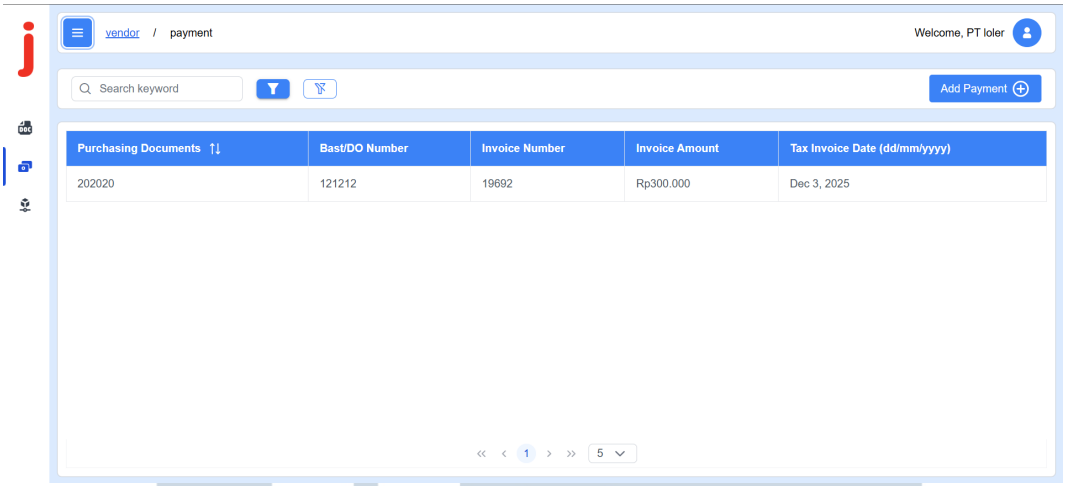
dibutuhkan dalam proses pengajuan DP.



Gambar 3.26. Form DP Request

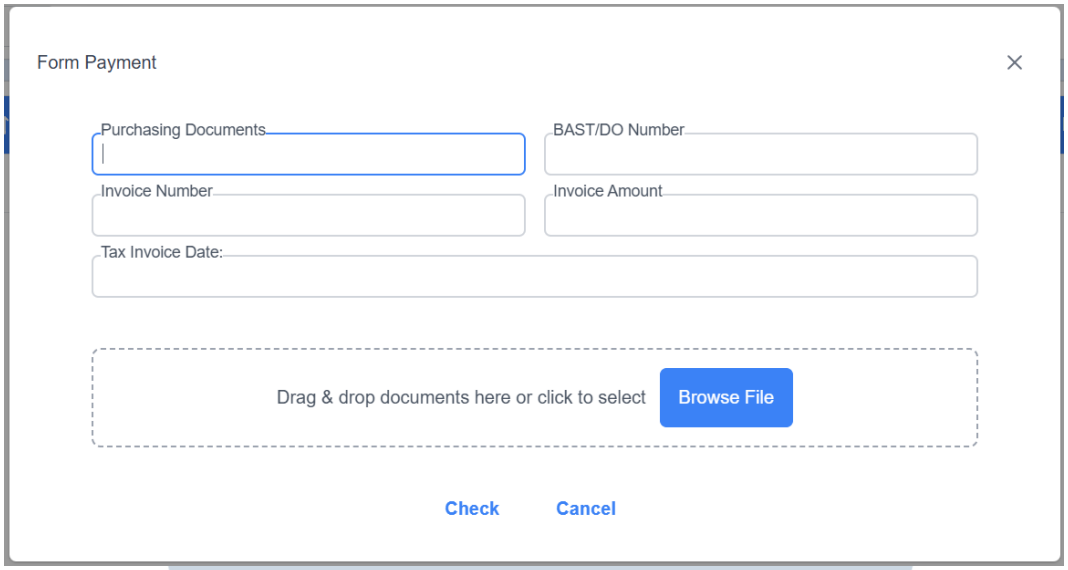
3.3.24 Tampilan Vendor Payment Request

Berdasarkan gambar halaman Vendor Payment, ditunjukkan tampilan halaman yang dirancang untuk memfasilitasi proses pengajuan pembayaran (payment) oleh vendor secara terstruktur dan terdokumentasi. Pada halaman ini, vendor dapat melihat daftar pengajuan pembayaran yang telah dibuat dalam bentuk tabel yang menyajikan informasi utama, antara lain Purchasing Documents, BAST/DO Number, Invoice Number, Invoice Amount, serta Tax Invoice Date. Untuk mendukung kemudahan pencarian data, halaman ini juga dilengkapi dengan fitur pencarian dan filter berdasarkan kata kunci tertentu. Selain itu, tersedia tombol Add Payment yang berfungsi sebagai akses utama bagi vendor untuk menambahkan pengajuan pembayaran baru. Perancangan halaman ini bertujuan untuk memastikan data pembayaran dapat ditampilkan secara jelas, rapi, dan mudah dipahami oleh pengguna.



Gambar 3.27. Vendor Payment Request

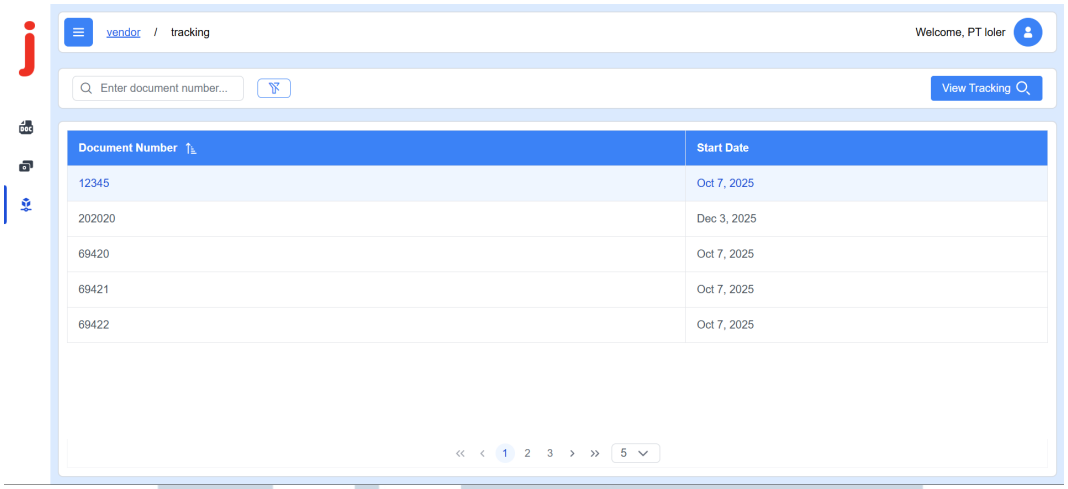
Selanjutnya, gambar 3.28 menunjukkan tampilan formulir yang digunakan vendor untuk melakukan pengajuan pembayaran secara detail. Formulir ini memuat field penting seperti Purchasing Documents, BAST/DO Number, Invoice Number, Invoice Amount, dan Tax Invoice Date yang harus diisi sesuai dengan data transaksi yang berlaku. Selain pengisian data, formulir ini juga menyediakan fitur unggah dokumen pendukung melalui mekanisme drag and drop maupun pemilihan file secara langsung. Pada bagian bawah formulir terdapat tombol Check untuk melakukan pengecekan atau validasi data sebelum diproses lebih lanjut, serta tombol *Cancel* untuk membatalkan proses pengajuan. Perancangan formulir ini mengutamakan kejelasan alur pengisian, kemudahan penggunaan, serta kelengkapan informasi guna mendukung proses pengajuan pembayaran yang efektif dan akurat.



Gambar 3.28. Form Payment Request

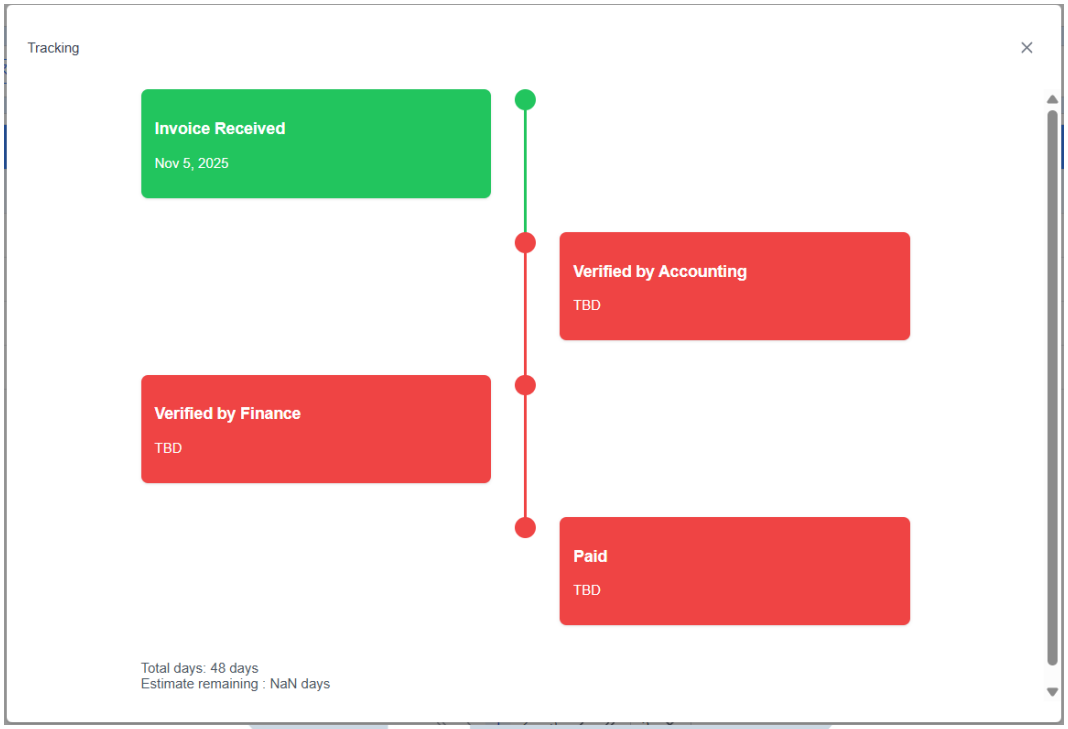
3.3.25 Tampilan Vendor Tracking

Berdasarkan gambar 3.45, ditunjukkan tampilan halaman yang dirancang untuk memfasilitasi vendor dalam memantau status dan progres dokumen yang telah diajukan ke dalam sistem. Pada halaman ini, vendor dapat melihat daftar dokumen dalam bentuk tabel yang menampilkan informasi utama seperti Document Number dan Start Date. Disediakan pula fitur pencarian berdasarkan nomor dokumen untuk memudahkan vendor dalam menemukan data tertentu secara cepat dan efisien. Tombol View Tracking berfungsi sebagai akses utama untuk melihat detail perkembangan dokumen yang dipilih. Perancangan halaman ini bertujuan untuk memberikan transparansi proses serta memudahkan vendor dalam melakukan pemantauan status dokumen secara mandiri.



Gambar 3.29. Vendor Tracking

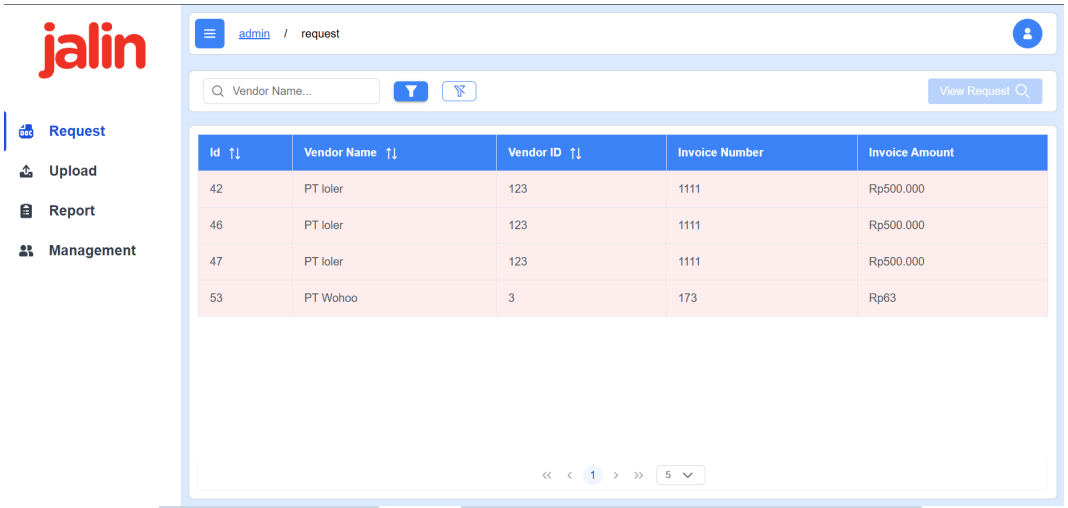
Selanjutnya, gambar 3.30 menunjukkan visualisasi alur proses dokumen dalam bentuk timeline yang menggambarkan setiap tahapan status secara berurutan. Tahapan tersebut meliputi proses Invoice Received, Verified by Accounting, Verified by Finance, hingga Paid. Setiap tahap ditampilkan dengan indikator visual berupa warna yang menunjukkan status, di mana warna hijau menandakan tahap telah selesai, sedangkan warna merah menunjukkan tahap yang belum diproses. Informasi tanggal juga ditampilkan pada setiap tahap untuk memberikan kejelasan waktu proses. Pada bagian bawah tampilan disajikan informasi tambahan berupa total durasi proses dokumen serta estimasi waktu penyelesaian. Perancangan tampilan tracking ini mengutamakan kejelasan informasi, kemudahan pemahaman alur proses, serta transparansi status dokumen bagi vendor.



Gambar 3.30. Vendor Detail Tracking

3.3.26 Tampilan Admin Request

Berdasarkan gambar 3.31, ditunjukkan halaman yang digunakan oleh pihak Accounting dan Finance untuk melakukan pengecekan serta proses approve atau reject terhadap pengajuan DP Request maupun Payment yang diajukan oleh vendor. Pada halaman ini, data pengajuan ditampilkan dalam bentuk tabel yang memuat informasi utama seperti Id, Vendor Name, Vendor ID, Invoice Number, dan Invoice Amount. Disediakan pula fitur pencarian berdasarkan nama vendor untuk memudahkan proses penelusuran data. Tombol View Request berfungsi untuk membuka detail pengajuan yang dipilih sehingga admin dapat melakukan pengecekan secara lebih mendalam. Perancangan halaman ini bertujuan untuk mendukung alur verifikasi berjenjang serta memastikan setiap pengajuan vendor diproses secara terkontrol dan transparan.



Gambar 3.31. Admin Request

Selanjutnya, gambar 3.32 menunjukkan detail informasi dari pengajuan yang dipilih. Pada form ini ditampilkan data penting seperti Purchasing Document Number, Invoice Number, Invoice Amount, serta Attachments berupa dokumen pendukung yang diunggah oleh vendor. Informasi ini digunakan oleh pihak Accounting maupun Finance untuk melakukan validasi kesesuaian data dan kelengkapan dokumen. Apabila pengajuan tidak memenuhi ketentuan, admin dapat mengisi Rejection Note sebagai alasan penolakan. Di bagian bawah form tersedia tombol Approve dan Reject sebagai aksi utama dalam proses pengambilan keputusan.

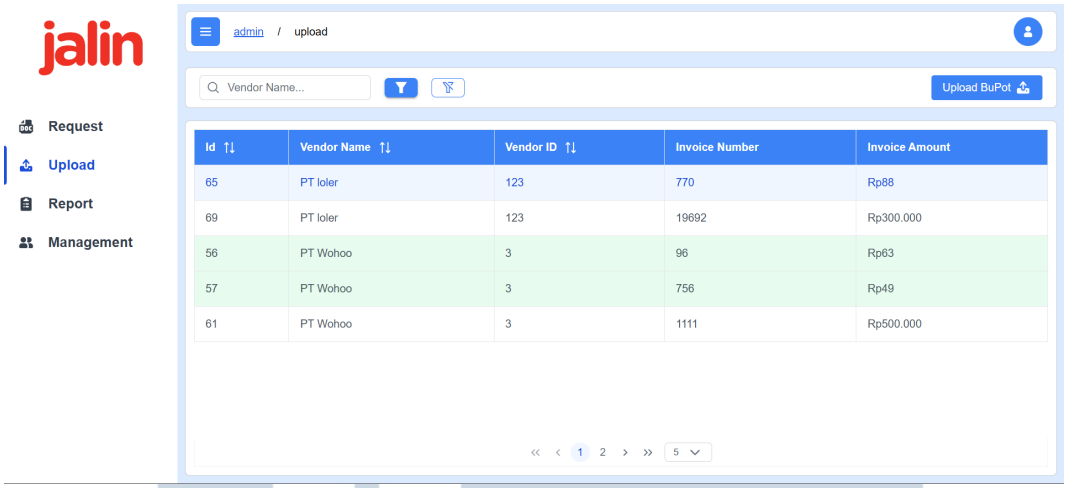
Gambar 3.32. Admin Request Form

Dalam alur prosesnya, Accounting melakukan pengecekan awal terhadap pengajuan yang masuk. Jika pengajuan disetujui oleh Accounting, maka data

tersebut akan diteruskan dan ditampilkan pada tabel ketika Finance melakukan login. Sebaliknya, apabila pengajuan ditolak oleh Accounting, maka data tidak akan diteruskan ke tahap berikutnya dan tidak akan muncul pada tabel Finance. Selanjutnya, Finance melakukan pengecekan lanjutan terhadap pengajuan yang telah disetujui Accounting. Jika Finance menyetujui pengajuan tersebut, maka data akan muncul pada sisi Admin untuk proses lanjutan. Pada tahap admin, dilakukan pengecekan status paid. Apabila pembayaran telah dilakukan (paid), maka data akan otomatis hilang dari tabel. Namun, jika pada tahap ini pengajuan ditolak, maka data akan tetap ditandai dengan status penolakan yang ditampilkan dengan indikator warna merah. Perancangan alur ini bertujuan untuk memastikan proses verifikasi berjalan berlapis, akurat, serta meminimalkan kesalahan dalam pengelolaan pembayaran vendor.

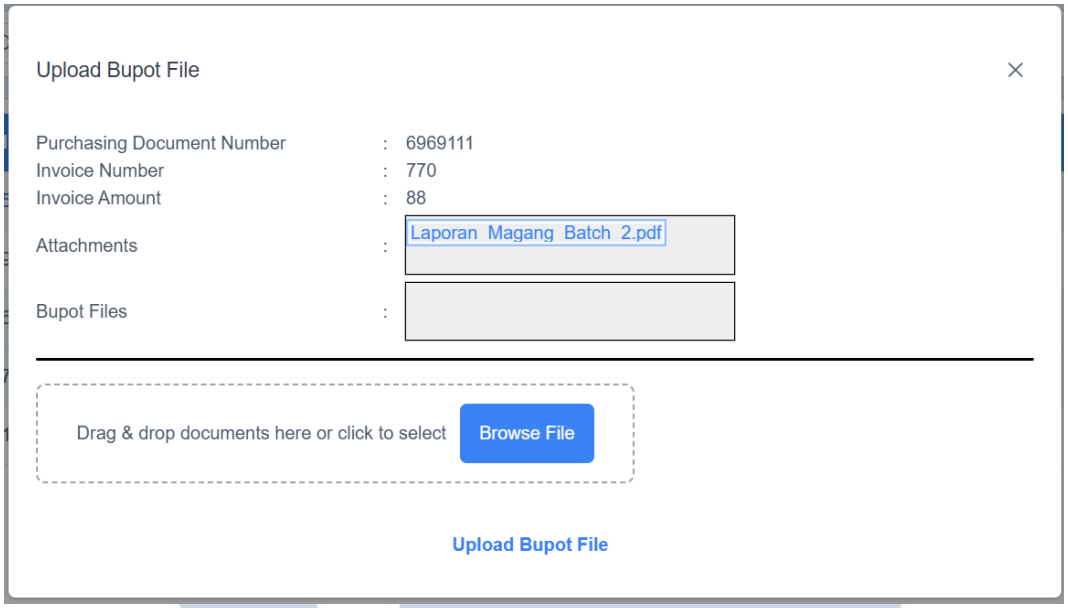
3.3.27 Tampilan Admin Upload

Berdasarkan gambar 3.33, ditunjukkan halaman khusus yang hanya dapat diakses oleh admin untuk melakukan pengelolaan dan unggah dokumen Bukti Potong (Bupot) terhadap pengajuan vendor yang telah melalui seluruh tahapan persetujuan. Data yang ditampilkan pada halaman ini merupakan data request yang sebelumnya telah disetujui secara berurutan oleh Accounting dan Finance, kemudian diteruskan ke admin untuk proses unggah Bupot. Informasi disajikan dalam bentuk tabel yang memuat data utama seperti Id, Vendor Name, Vendor ID, Invoice Number, dan Invoice Amount. Pada tabel tersebut, baris data yang ditandai dengan warna hijau menunjukkan bahwa file Bupot untuk request terkait telah berhasil diunggah, sehingga memudahkan admin dalam membedakan data yang sudah dan belum diproses.



Gambar 3.33. Admin Upload Bupot

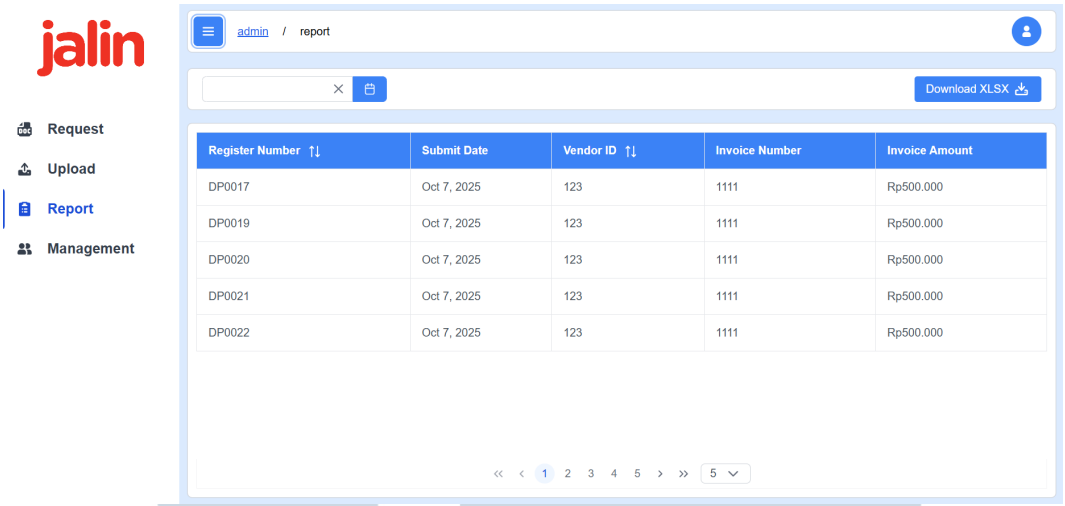
Selanjutnya, gambar 3.34 digunakan oleh admin untuk melakukan proses unggah file Bupot secara detail. Pada form ini ditampilkan informasi ringkas terkait request, seperti Purchasing Document Number, Invoice Number, Invoice Amount, serta Attachments berupa dokumen pendukung yang sebelumnya diunggah oleh vendor. Admin kemudian dapat menambahkan file Bupot melalui area unggah yang mendukung metode drag and drop maupun pemilihan file secara langsung. Tombol Upload Bupot File berfungsi untuk menyimpan dan mengaitkan file Bupot ke request yang bersangkutan. Perancangan halaman dan form ini bertujuan untuk memastikan proses unggah Bupot dilakukan secara terkontrol, terdokumentasi dengan baik, serta mendukung transparansi dan ketertelusuran data dalam proses administrasi pembayaran vendor.



Gambar 3.34. Admin Upload Bupot Form

3.3.28 Tampilan Admin Report

Berdasarkan gambar 3.35, ditunjukkan halaman laporan yang hanya dapat diakses oleh admin untuk melihat rekapitulasi data pengajuan DP maupun Payment yang telah diproses di dalam sistem. Halaman ini menyajikan data dalam bentuk tabel yang memuat informasi penting seperti Register Number, Submit Date, Vendor ID, Invoice Number, dan Invoice Amount. Penyajian data secara terstruktur ini bertujuan untuk memudahkan admin dalam melakukan pemantauan, evaluasi, serta dokumentasi terhadap seluruh transaksi yang telah tercatat di sistem.



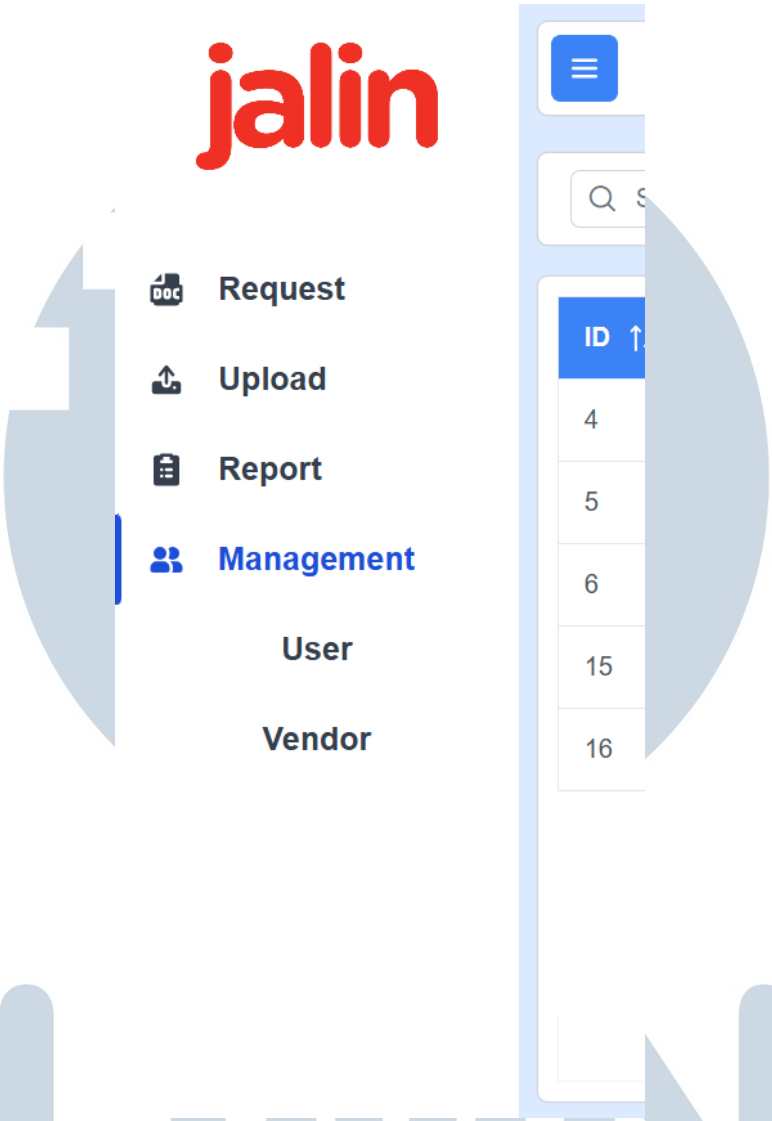
Gambar 3.35. Admin Report

Pada halaman ini tersedia fitur filter berdasarkan tanggal pengajuan (Submit Date) yang disajikan melalui komponen pemilih tanggal (date picker). Admin dapat menentukan rentang waktu tertentu, misalnya dari 10-10-2025 hingga 10-11-2025, sehingga data yang ditampilkan pada tabel disesuaikan dengan periode pengajuan yang dipilih. Fitur ini digunakan untuk membantu proses penyusunan laporan dan penelusuran data berdasarkan periode waktu tertentu secara sistematis.

Selain itu, halaman Admin Report dilengkapi dengan tombol *Download XLSX* yang berada di bagian kanan atas halaman. Setelah menekan tombol *Download XLSX* maka admin dapat mengunduh seluruh data yang ditampilkan pada tabel ke dalam bentuk file Microsoft Excel. Fitur unduhan ini mendukung kebutuhan pelaporan lanjutan, pengarsipan data, serta pengolahan data di luar sistem. Secara keseluruhan, perancangan halaman Admin Report bertujuan untuk menyediakan sarana pelaporan yang lengkap, fleksibel, dan mudah digunakan oleh admin.

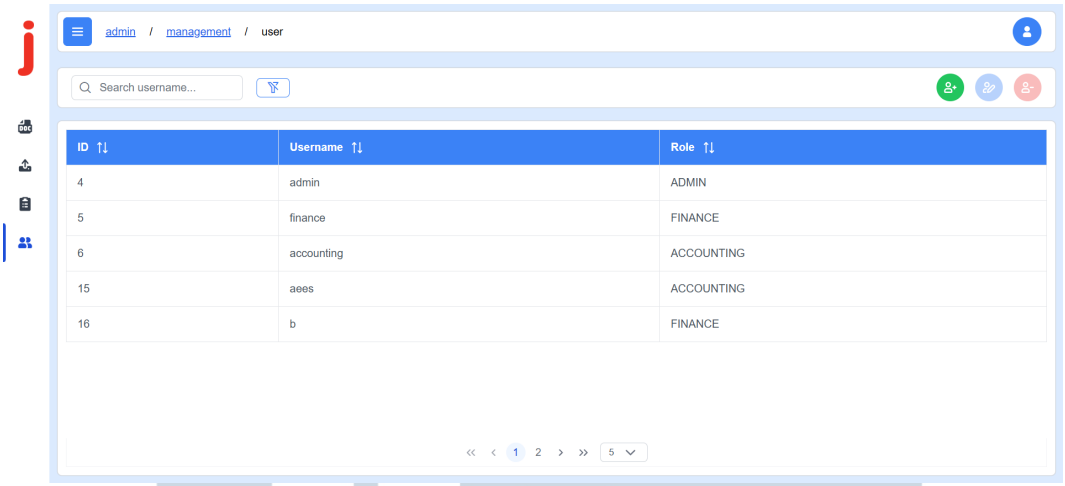
3.3.29 Admin Management

Berdasarkan gambar 3.36, ditunjukkan komponen sidebar yang hanya dapat diakses oleh admin untuk melakukan pengelolaan data pengguna (user) dan data mitra (vendor) yang terdaftar di dalam sistem. Komponen ini berfungsi sebagai pusat administrasi data master yang berkaitan dengan hak akses sistem serta identitas vendor yang terlibat dalam proses pengajuan DP dan Payment.



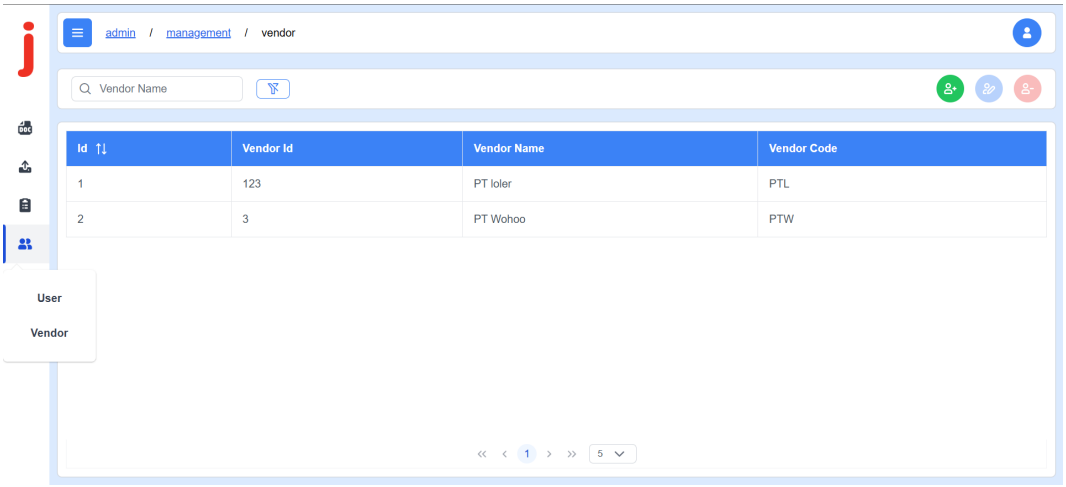
Gambar 3.36. Komponen Sidebar Management

Pada submenu gambar 3.37, ditampilkan daftar pengguna internal sistem dalam bentuk tabel yang berisi informasi ID, Username, dan Role. Role pengguna dibedakan menjadi beberapa kategori, yaitu ADMIN, FINANCE, dan ACCOUNTING, yang masing-masing memiliki hak akses berbeda sesuai dengan alur bisnis sistem. Admin dapat melakukan pencarian data user berdasarkan username serta mengelola data pengguna melalui aksi tambah, ubah, dan hapus. Proses pembaruan data user dilakukan melalui Form Update User, yang menyediakan field Username, Password, dan Role. Pada form ini, password dapat dikosongkan apabila tidak ingin dilakukan perubahan, sehingga menjaga keamanan dan fleksibilitas pengelolaan akun.



Gambar 3.37. Admin Management

Selanjutnya, pada submenu pada gambar 3.38, ditampilkan daftar vendor yang terdaftar di sistem dalam bentuk tabel dengan informasi Id, Vendor ID, Vendor Name, dan Vendor Code. Data vendor ini digunakan sebagai referensi utama dalam seluruh proses pengajuan DP, Payment, serta tracking dokumen oleh pihak vendor. Admin dapat melakukan pencarian vendor berdasarkan nama vendor untuk mempercepat proses penelusuran data.



Gambar 3.38. Admin Vendor Management

Pengelolaan data vendor dilakukan melalui beberapa form, yaitu Create Vendor, Update Vendor, dan Delete Vendor. Pada form Create Vendor, admin mengisikan data Vendor Name, Vendor ID, dan Vendor Code untuk menambahkan vendor baru ke dalam sistem. Form Update Vendor digunakan untuk memperbarui data vendor yang telah ada, sedangkan form Delete Vendor berfungsi sebagai

konfirmasi penghapusan data vendor dengan menampilkan informasi detail vendor yang akan dihapus. Mekanisme konfirmasi ini bertujuan untuk mencegah kesalahan penghapusan data secara tidak disengaja.

Secara keseluruhan, perancangan halaman Admin Management difokuskan pada kemudahan pengelolaan data master, kejelasan struktur informasi, serta kontrol penuh oleh admin terhadap pengguna dan vendor yang terlibat di dalam sistem. Halaman ini mendukung keberlangsungan proses bisnis sistem JEBRET dengan memastikan data pengguna dan vendor selalu terkelola secara konsisten dan terintegrasi.

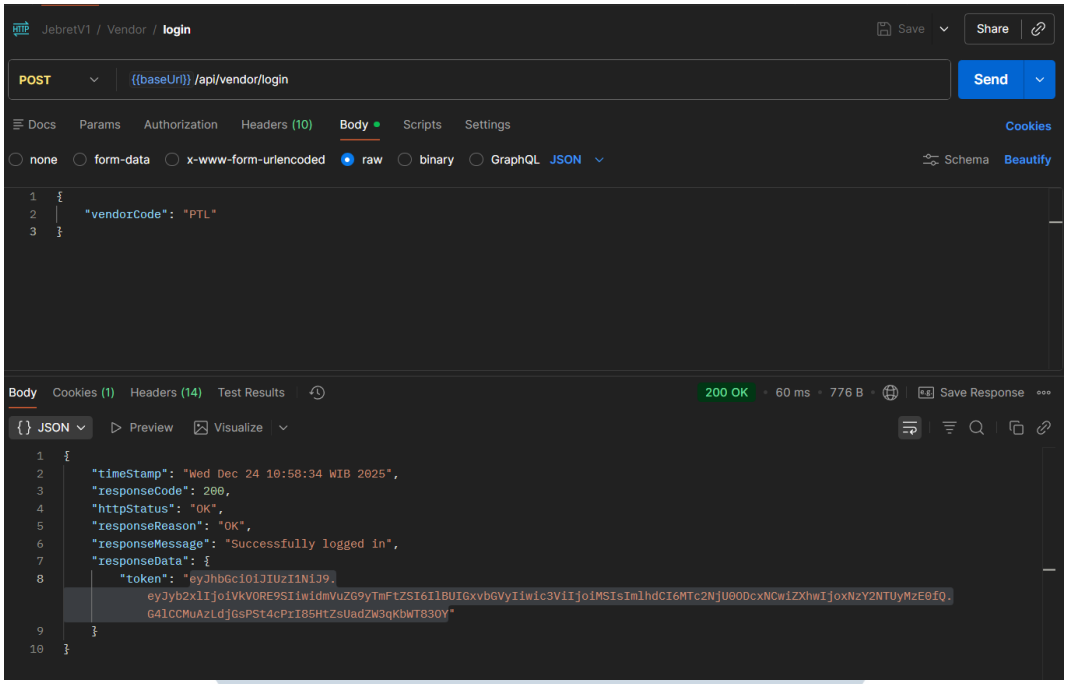
3.3.30 API Autentikasi Vendor

API autentikasi vendor berfungsi untuk mengamankan akses vendor ke dalam sistem serta mengelola siklus sesi login dan logout. Endpoint utama yang diimplementasikan pada modul ini meliputi login vendor dan logout vendor, yang masing-masing menggunakan metode *HTTP POST*. Rincian dari API Autentikasi dapat dilihat pada tabel 3.2.

Tabel 3.2. API Autentikasi Vendor

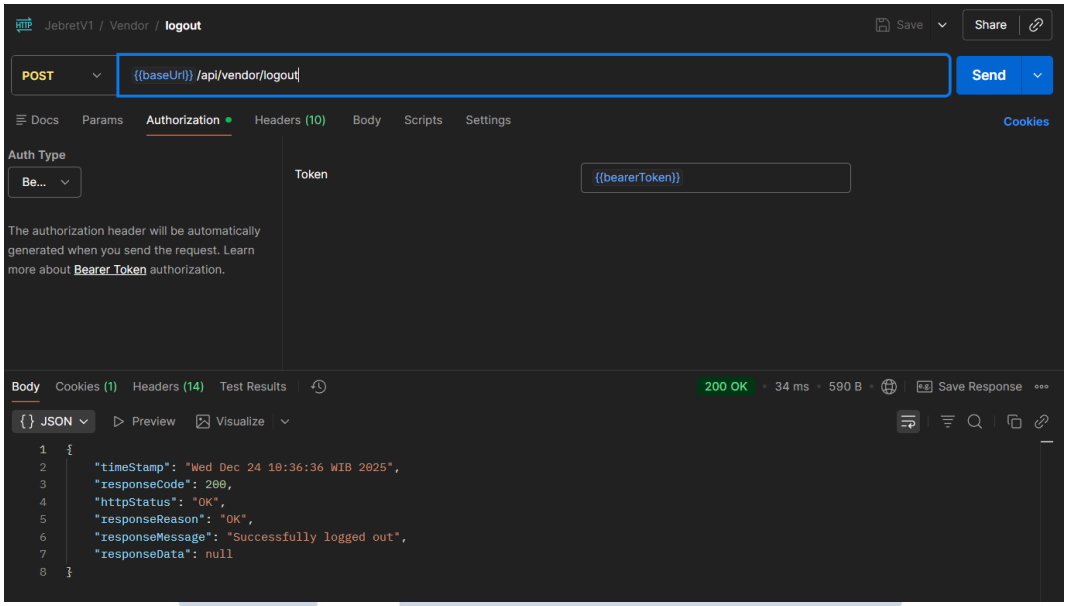
Nomor	Endpoint & Metode HTTP	Ringkasan Endpoint
1	POST `/api/vendor/login`	Mengautentikasi `Vendor Code` lalu mengembalikan `token`
2	POST `/api/vendor/logout`	Menghapus `token` dari sesi backend

Pada gambar 3.39, tampak pengujian dilakukan melalui Postman dengan metode POST ke endpoint `/api/vendor/login`. Pada tab Body, format data yang dikirim adalah raw JSON yang berisi parameter vendorCode (contoh: ”PTL”) sebagai identitas vendor yang digunakan untuk autentikasi. Setelah request dikirim, Postman menampilkan `status 200 OK` yang menunjukkan proses login berhasil. Pada bagian response, struktur balasan menggunakan format standar sistem (misalnya `timeStamp`, `responseCode`, `httpStatus`, `responseReason`, dan `responseMessage`), serta terdapat responseData yang memuat token JWT. Token ini merupakan kredensial akses yang nantinya digunakan vendor untuk mengakses endpoint lain yang membutuhkan autentikasi.



Gambar 3.39. Vendor Login Endpoint

Pada gambar 3.40, tampak pengujian dilakukan melalui Postman dengan metode *POST* ke endpoint `/api/vendor/logout`. Request logout menggunakan tab *Authorization* dengan tipe *Bearer Token*, yang berarti Postman mengirimkan token akses pada header *Authorization* untuk membuktikan bahwa vendor sudah terautentikasi. Setelah request dikirim, hasilnya menunjukkan `status 200 OK` yang menandakan proses logout berhasil dijalankan. Response yang ditampilkan kembali menggunakan format standar yang sama, dengan `responseMessage` berisi informasi bahwa pengguna berhasil logout, sedangkan `responseData` bernilai `null` karena endpoint ini hanya mengakhiri sesi tanpa mengembalikan data tambahan.



Gambar 3.40. Vendor Logout Endpoint

3.3.31 API Vendor DP Request

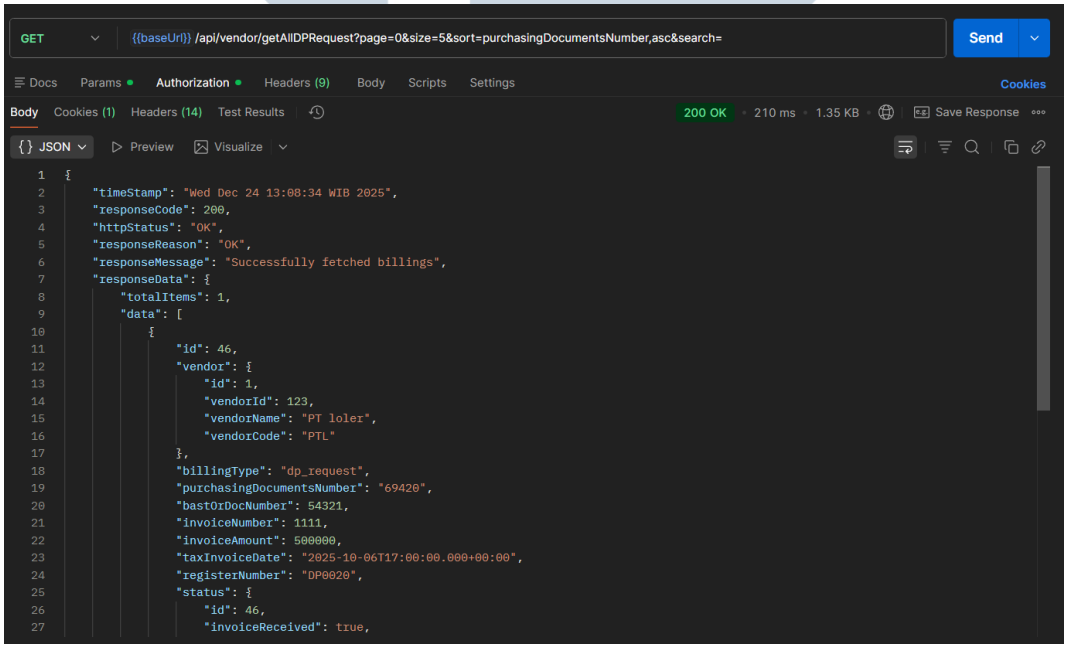
Pada bagian ini menjelaskan implementasi API pada sisi Vendor yang digunakan untuk mengelola proses DP Request, mencakup pengambilan daftar pengajuan DP yang telah dibuat vendor serta pembuatan pengajuan DP baru. API ini membantu vendor memantau status dokumen secara bertahap (misalnya verifikasi Accounting/Finance hingga keputusan reject/paid) melalui data status yang dikembalikan oleh backend. Penjelasan rinci dapat dilihat pada tabel 3.3

Tabel 3.3. API Vendor DP Request

Nomor	Endpoint & Metode HTTP	Ringkasan Endpoint
1	GET `/api/vendor/getAllDPRequest`	Mengambil daftar pengajuan DP milik vendor dengan dukungan paginasi, sorting, dan pencarian.
		Lanjutan di halaman berikutnya...

Nomor	Endpoint & Metode HTTP	Ringkasan Endpoint
2	POST `/api/vendor/createDp`	Membuat pengajuan DP baru beserta lampiran file (`multipart/form-data`) dan menghasilkan nomor registrasi DP.

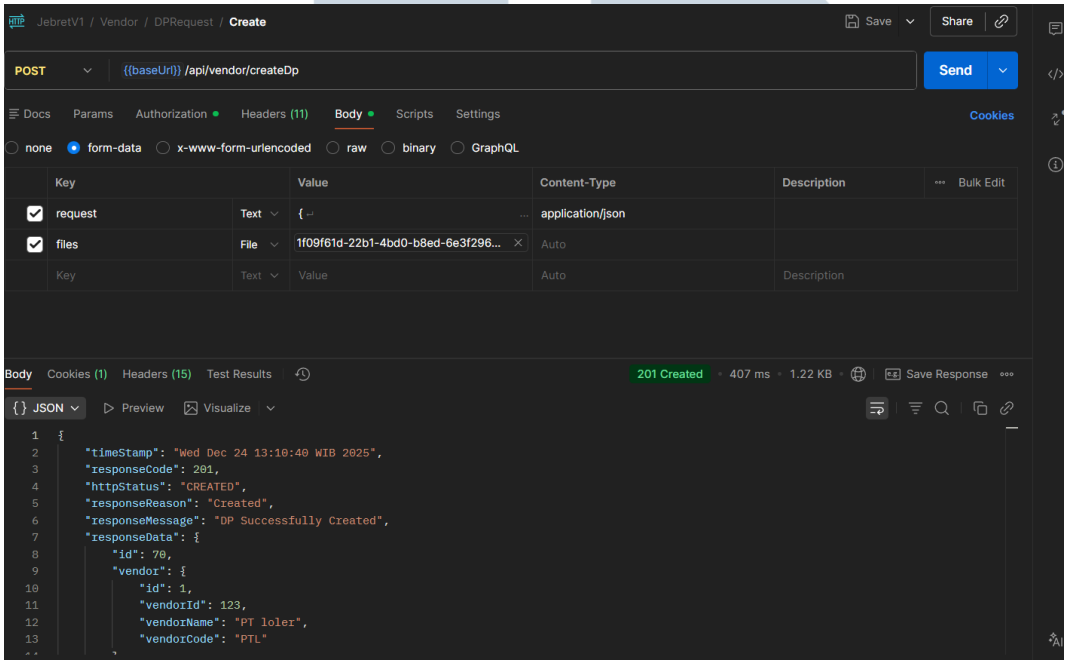
Pada gambar 3.41, terlihat request dilakukan menggunakan metode GET ke endpoint /api/vendor/getAllDPRequest dengan parameter query `page`, `size`, `sort`, dan `search`. Parameter page menunjukkan halaman data yang diminta, size menentukan jumlah data per halaman, sort mengatur kolom dan arah pengurutan (misalnya berdasarkan `purchasingDocumentsNumber` secara ascending), sedangkan search disediakan untuk kebutuhan filter pencarian. Request ini dijalankan dalam kondisi terautentikasi, sehingga vendor yang login hanya akan memperoleh daftar pengajuan yang relevan sesuai identitas vendornya.



Gambar 3.41. Vendor Get DP Request

Pada gambar 3.42, request dilakukan menggunakan metode POST ke endpoint /api/vendor/createDp dengan format multipart/form-data. Pada body request terlihat dua bagian utama, yaitu field request yang berisi data pengajuan dalam format JSON (misalnya identitas vendor, tipe billing, nomor dokumen, invoice, jumlah invoice, tanggal tax invoice, dan total waktu), serta field files yang

berisi lampiran dokumen yang diunggah. Pola ini menunjukkan bahwa backend mendukung pembuatan data pengajuan sekaligus upload berkas pendukung dalam satu transaksi request, sehingga data DP Request dan lampiran dapat tercatat dalam satu alur pengajuan.



Gambar 3.42. Vendor Create DP

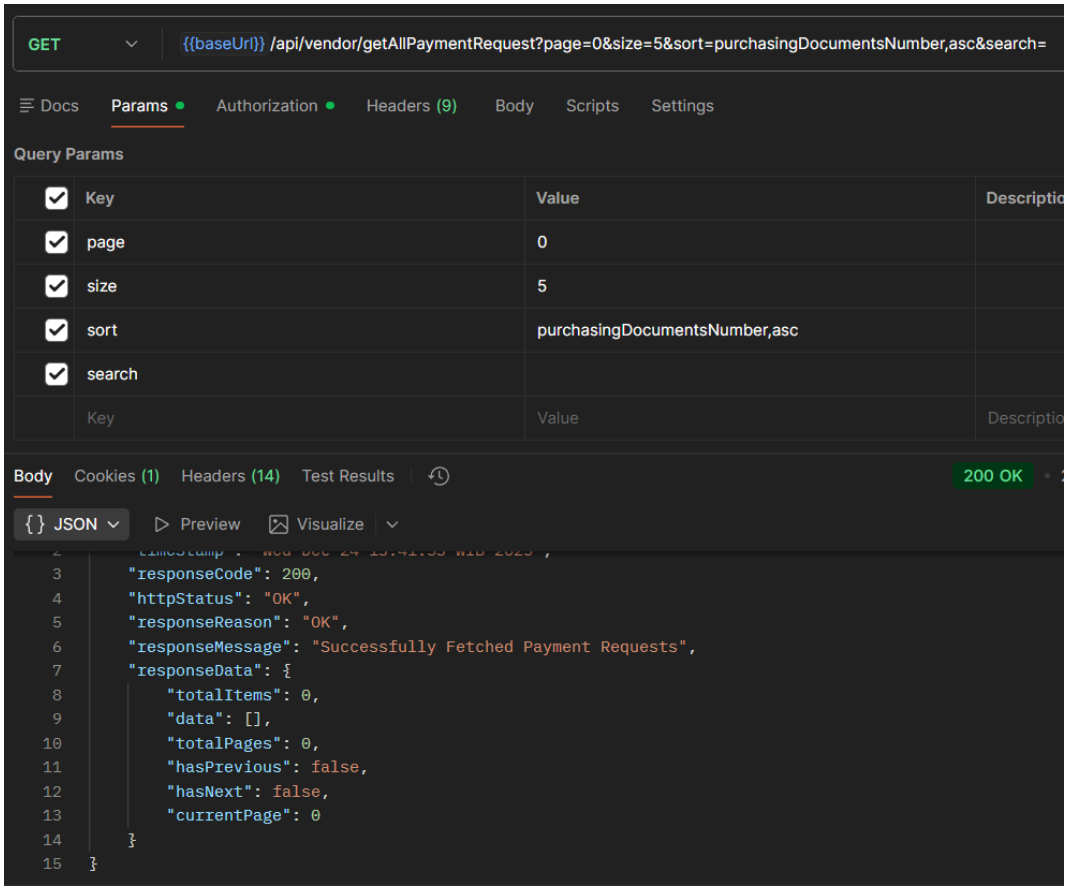
3.3.32 API Vendor Payment

API Payment Vendor digunakan untuk mengelola proses Payment Request, yaitu pengajuan pembayaran yang dibuat oleh vendor serta pemantauan statusnya. Fitur ini mencakup endpoint untuk mengambil daftar payment request yang dimiliki vendor dan endpoint untuk membuat payment request baru disertai lampiran dokumen pendukung. Penjelasan rincian dapat dilihat pada tabel 3.4

Tabel 3.4. API Vendor Payment

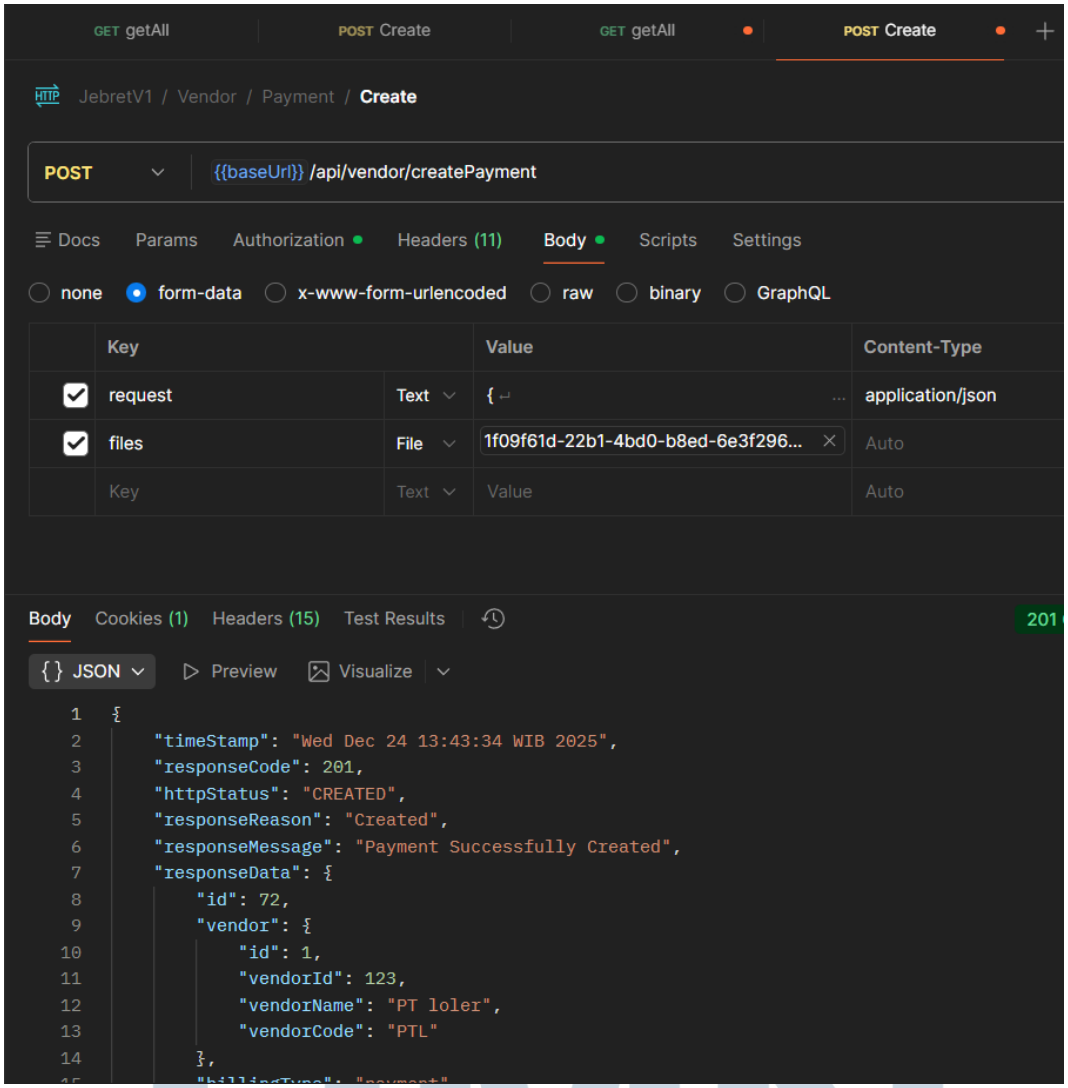
Nomor	Endpoint & Metode HTTP	Ringkasan Endpoint
1	GET `/api/vendor/getAllPaymentRequest`	Mengambil daftar pengajuan payment milik vendor dengan dukungan paginasi, sorting, dan pencarian.
2	POST `/api/vendor/createPayment`	Membuat pengajuan payment baru beserta lampiran file (`multipart/form-data`) dan menghasilkan nomor registrasi payment.

Pada gambar 3.43, terlihat request dilakukan menggunakan metode GET ke endpoint /api/vendor/getAllPaymentRequest dengan query parameter `page`, `size`, `sort`, dan `search`. Parameter `page` menunjukkan nomor halaman data, `size` menentukan jumlah data per halaman, sedangkan `sort` mengatur pengurutan data berdasarkan `purchasingDocumentsNumber` secara *ascending*. Kolom search disediakan untuk pencarian data tertentu apabila diperlukan. Pengujian ini menunjukkan bahwa backend dapat mengembalikan data sesuai parameter yang diberikan, serta mendukung kebutuhan tampilan tabel pada sisi frontend.



Gambar 3.43. Vendor Get All Payment

Pada gambar 3.44, terlihat request dilakukan menggunakan metode POST ke endpoint `/api/vendor/createPayment` dengan format `multipart/form-data`. Pada bagian body terdapat dua komponen utama, yaitu field request yang berisi data pengajuan dalam format JSON (misalnya identitas vendor, nomor dokumen, nomor invoice, jumlah invoice, dan tanggal tax invoice), serta field files yang digunakan untuk mengunggah file dokumen pendukung. Struktur ini menunjukkan bahwa backend dirancang untuk menerima pembuatan data payment request sekaligus unggahan lampiran dalam satu request, sehingga proses pengajuan vendor menjadi lebih efisien.



Gambar 3.44. Vendor Create Payment

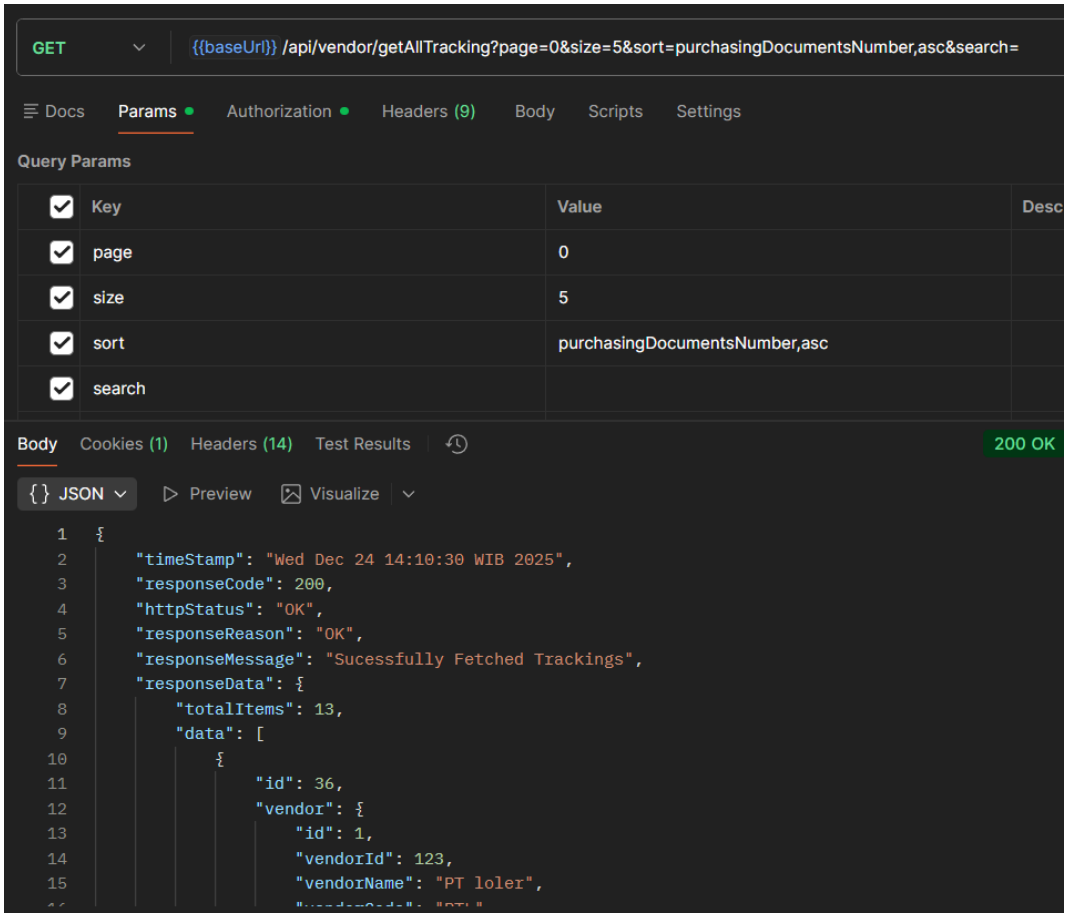
3.3.33 API Vendor Tracking

API Vendor Tracking digunakan untuk menampilkan daftar riwayat dan progres pemrosesan dokumen yang telah diajukan oleh vendor, baik untuk jenis pengajuan DP Request maupun Payment Request. Melalui endpoint ini, vendor dapat memantau tahapan dokumen secara *end-to-end*, mulai dari status penerimaan invoice, verifikasi oleh Accounting dan Finance, hingga kondisi akhir seperti paid atau rejected beserta alasan penolakan apabila terjadi penolakan. Rincian API Tracking dapat dilihat pada tabel 3.5

Tabel 3.5. API Vendor Tracking

Nomor	Endpoint & Metode HTTP	Ringkasan Endpoint
1	GET `/api/vendor/getAllTracking`	Mengambil daftar tracking pengajuan vendor (DP dan Payment) dengan dukungan paginasi, sorting, dan pencarian.

Pada gambar 3.45, terlihat request dilakukan menggunakan metode GET ke endpoint `/api/vendor/getAllTracking` dengan query parameter `page`, `size`, `sort`, dan `search`. Parameter `page` digunakan untuk menentukan halaman data, `size` menentukan jumlah data per halaman, `sort` mengatur urutan berdasarkan `purchasingDocumentsNumber` secara *ascending*, sedangkan search disediakan untuk memfilter data tertentu bila diperlukan. Setelah request dikirim, Postman menampilkan `status 200 OK` yang menandakan backend berhasil mengembalikan data tracking sesuai parameter yang diberikan.



Gambar 3.45. Vendor Tracking

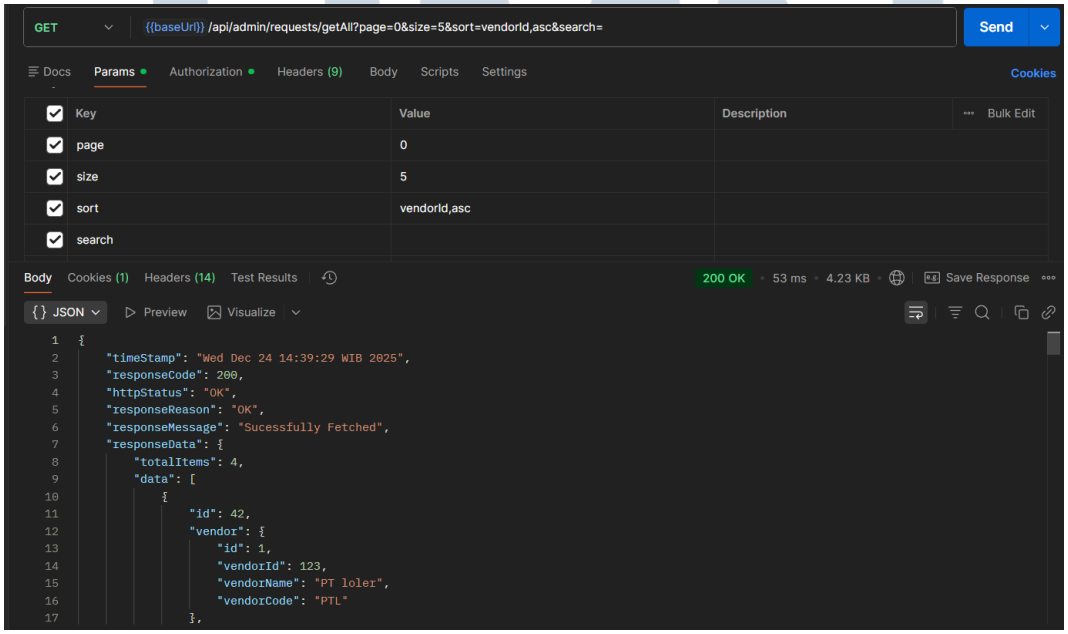
3.3.34 API Admin Request

API Admin Get All Requests digunakan untuk menampilkan seluruh data pengajuan billing yang masuk ke sistem, sehingga pihak internal perusahaan dapat melakukan pemantauan dan tindak lanjut proses dokumen. Endpoint ini dapat diakses oleh role ADMIN, ACCOUNTING, dan FINANCE, karena ketiga peran tersebut terlibat langsung dalam alur validasi dokumen, verifikasi, hingga keputusan akhir terhadap pengajuan vendor. Terdapat rincian Admin Request dalam tabel 3.6

Tabel 3.6. API Admin Request

Nomor	Endpoint & Metode HTTP	Ringkasan Endpoint
1	GET `/api/admin/requests/getAll`	Mengambil daftar seluruh pengajuan (DP/Payment) untuk kebutuhan monitoring dan proses verifikasi oleh Admin, Accounting, dan Finance, dengan dukungan paginasi, sorting, dan pencarian.

Pada gambar 3.46, terlihat request dilakukan menggunakan metode GET ke endpoint /api/admin/requests/getAll dengan query parameter `page`, `size`, `sort`, dan `search`. Parameter `page` menentukan halaman data yang diminta, `size` menentukan jumlah data per halaman, `sort` mengatur pengurutan (contohnya berdasarkan `vendorId` secara *ascending*), sedangkan search disediakan untuk memfilter data tertentu apabila diperlukan. Setelah request dikirim, Postman menampilkan `status 200 OK` yang menandakan bahwa backend berhasil mengembalikan daftar pengajuan sesuai parameter yang diberikan dan sesuai hak akses role yang digunakan (ADMIN/ACCOUNTING/FINANCE).



Gambar 3.46. API Admin Request

3.3.35 API Admin Upload

API Admin Upload Bukti Potong (Bupot) digunakan untuk mendukung kebutuhan administrasi dokumen pada sisi internal, khususnya dalam proses pelengkapan berkas pengajuan yang masuk dari vendor. Melalui fitur ini, admin dapat melihat daftar request yang tersedia untuk proses upload dokumen, mengunggah file bupot ke request tertentu, serta menyediakan endpoint untuk mengunduh kembali file bupot yang telah tersimpan. Tabel 3.7 merupakan rincian dari API Admin Upload.

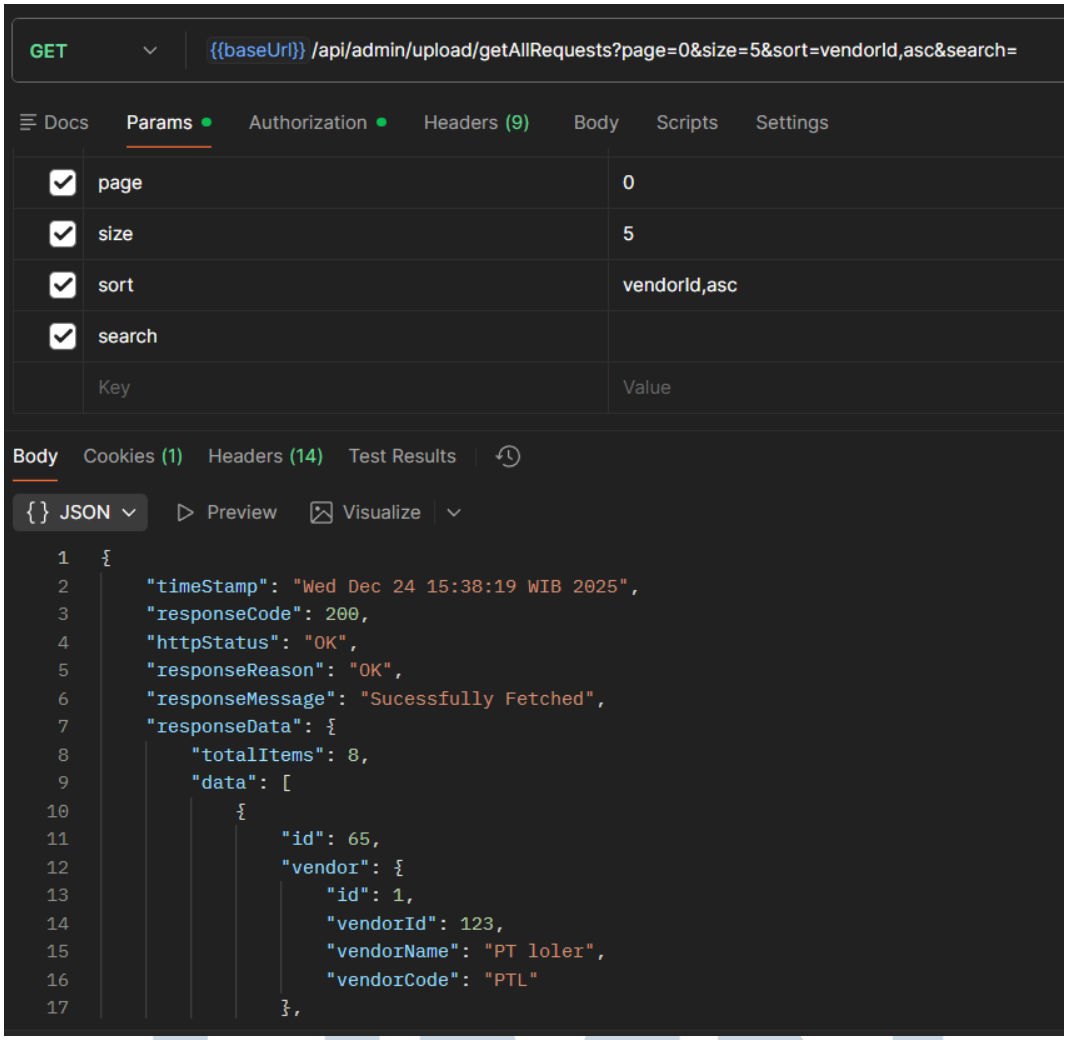
Tabel 3.7. API Admin Upload

Nomor	Endpoint & Metode HTTP	Ringkasan Endpoint
1	GET `/api/admin/upload/getAllRequests`	Mengambil daftar request yang dapat diproses pada modul upload, dengan dukungan paginasi, sorting, dan pencarian.
2	GET `/api/admin/upload/uploadBupots/{requestId}`	Mengunggah file bupot untuk request berdasarkan `requestId` tertentu menggunakan multipart/form-data.

Lanjutan di halaman berikutnya...

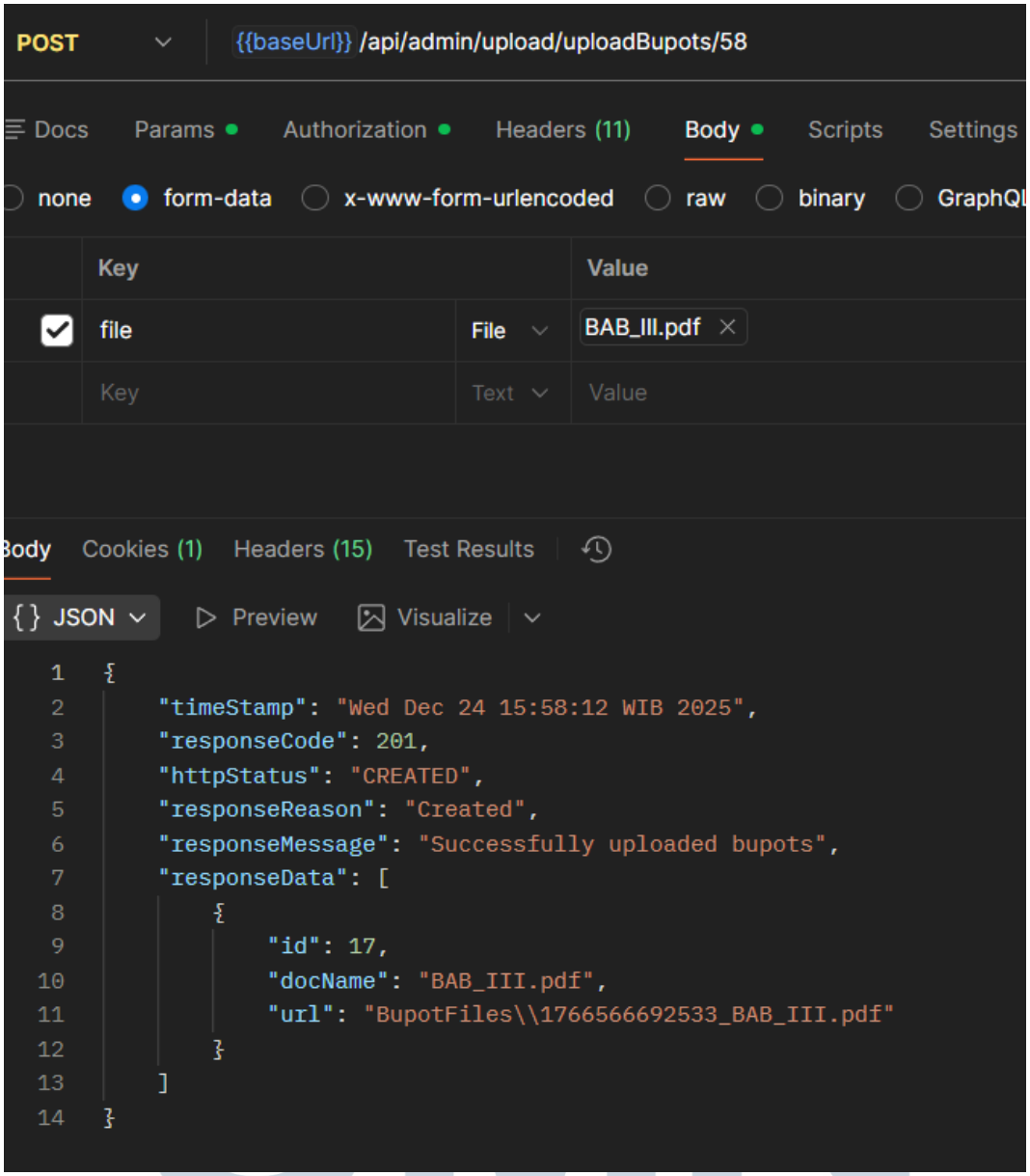
Nomor	Endpoint & Metode HTTP	Ringkasan Endpoint
3	GET `/api/admin/upload/getFile/{fileId}`	Mengunduh file bupot berdasarkan `fileId` yang tersimpan.

Pada gambar 3.47, terlihat request dilakukan menggunakan metode GET menuju /api/admin/upload/getAllRequests dengan query parameter `page`, `size`, `sort`, dan `search`. Parameter `page` menentukan halaman data, `size` menentukan jumlah data per halaman, sedangkan `sort` mengatur pengurutan data (contohnya berdasarkan `vendorId` secara *ascending*). Setelah request dijalankan, Postman menampilkan `status 200 OK` yang menunjukkan backend berhasil mengembalikan daftar request sesuai parameter yang diberikan.



Gambar 3.47. Admin Get Request (Upload)

Pada gambar 3.48, request dilakukan menggunakan metode POST ke endpoint `/api/admin/upload/uploadBupots/58`. Pada bagian body, format yang digunakan adalah `multipart/form-data` dengan field file bertipe File, yang menandakan bahwa backend menerima unggahan dokumen bupot sebagai file. Setelah request dikirim, Postman menampilkan `status 201 Created` yang menunjukkan file berhasil diunggah dan disimpan oleh sistem.



Gambar 3.48. Upload Bupot

Endpoint `/api/admin/upload/getFile/17` digunakan untuk mengunduh file bupot yang telah diunggah sebelumnya. Proses download dilakukan dengan mengirimkan `fileId` sesuai *metadata* yang diperoleh dari response upload bupot. Ketika endpoint dipanggil, backend akan mengambil file berdasarkan `fileId` tersebut dan mengembalikannya sebagai file download sehingga dokumen bupot dapat diakses kembali oleh pihak internal untuk kebutuhan verifikasi, arsip, maupun audit dokumen.

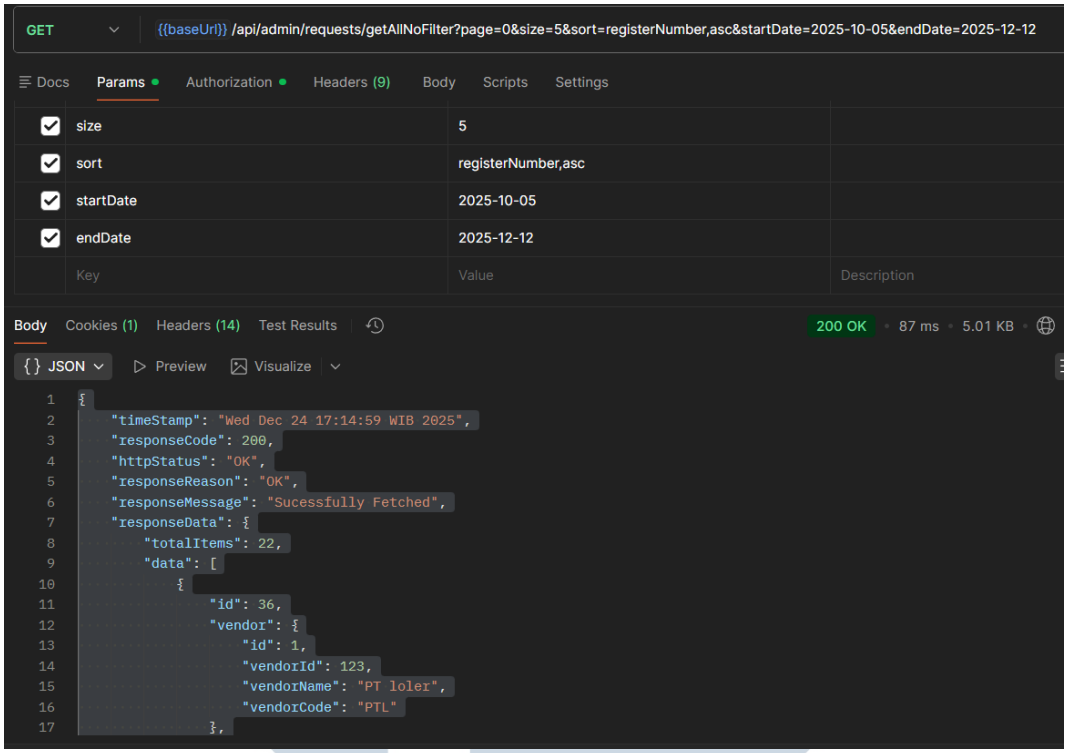
3.3.36 API Admin Report

Admin Report digunakan untuk meringkas seluruh data yang ada di Billing Request menjadi tabel. Fitur ini menggunakan `startDate` dan `endDate` untuk melakukan filter terhadap `Submit Date`. Rincian dari endpoint ini terdapat pada tabel 3.8

Tabel 3.8. API Admin Report

Nomor	Endpoint & Metode HTTP	Ringkasan Endpoint
1	GET `/api/admin/requests/getAllNoFilter?page=0&size=5&sort=registerNumber,asc&startDate=&endDate=`	Mengambil data Billing Request berdasarkan `startDate` sampai dengan `endDate`.

Pada gambar 3.49, terlihat request dilakukan melalui Postman menggunakan metode `GET` ke endpoint `/api/admin/requests/getAllNoFilter` dengan beberapa query parameter yang aktif, yaitu `page=0` dan `size=5` untuk mengatur paginasi, `sort=registerNumber,asc` untuk mengurutkan data berdasarkan nomor registrasi secara menaik, serta `startDate=2025-10-05` dan `endDate=2025-12-12` untuk memfilter data laporan berdasarkan rentang tanggal. Pada tab `Authorization` digunakan tipe `Bearer Token` yang menunjukkan endpoint hanya dapat diakses oleh user internal yang telah login. Setelah tombol `Send` ditekan, Postman menampilkan status `200 OK` yang menandakan request berhasil, lalu pada panel response terlihat struktur balasan standar sistem berisi `timeStamp`, `responseCode`, `httpStatus`, `responseReason`, `responseMessage`, dan `responseData`, di mana `responseData` memuat `totalItems` sebagai jumlah data hasil filter serta daftar `data` yang menampilkan item laporan pada halaman pertama.



Gambar 3.49. Admin Get Report

3.3.37 Admin Management

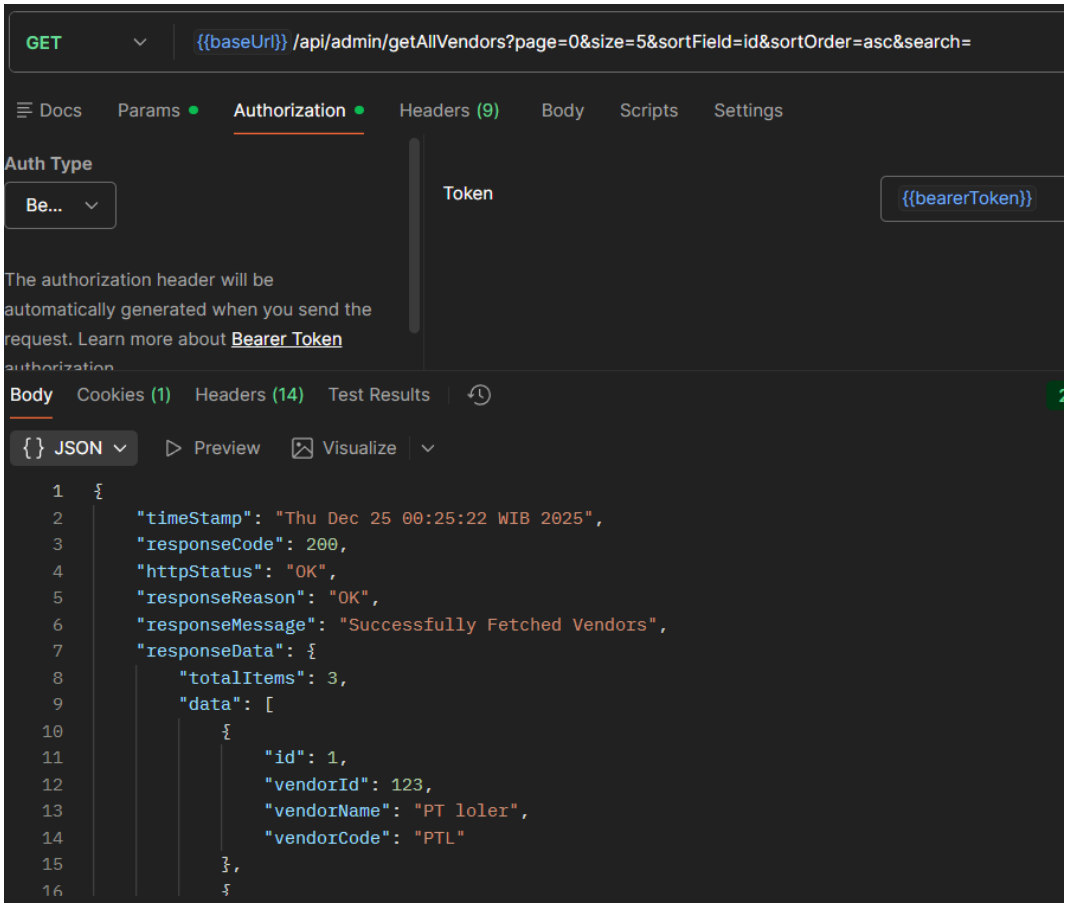
Admin Management digunakan untuk mengelola User dan Vendor yang terdaftar dalam basis data di dalam sistem JEBRET. Dengan fitur ini, admin dapat melakukan aksi CRUD (Create, Read, Update, Delete) data Vendor dan User dengan mudah. Rincian dari endpoint ini terdapat pada tabel 3.9

Tabel 3.9. API Admin Upload

Nomor	Endpoint & Metode HTTP	Ringkasan Endpoint
1	GET `/api/admin/getAllVendors`	Mengambil daftar Vendor yang terdapat pada sistem basis data JEBRET, dengan dukungan paginasi, sorting, dan pencarian.
2	POST `/api/admin/createVendor`	Membuat Vendor baru dan menyimpan data user ke dalam basis data JEBRET.
3	PUT `/api/admin/updateVendor/{id}`	Memperbarui data Vendor berdasarkan dari `id` yang dimasukkan ke dalam query.
4	DELETE `/api/admin/deleteVendor/{id}`	Menghapus Vendor dari basis data berdasarkan dari `id` yang dicantumkan ke dalam query.
5	GET `/api/admin/getAllUsers`	Mengambil daftar User yang terdapat pada sistem basis data JEBRET, dengan dukungan paginasi, sorting, dan pencarian.
		Lanjutan di halaman berikutnya...

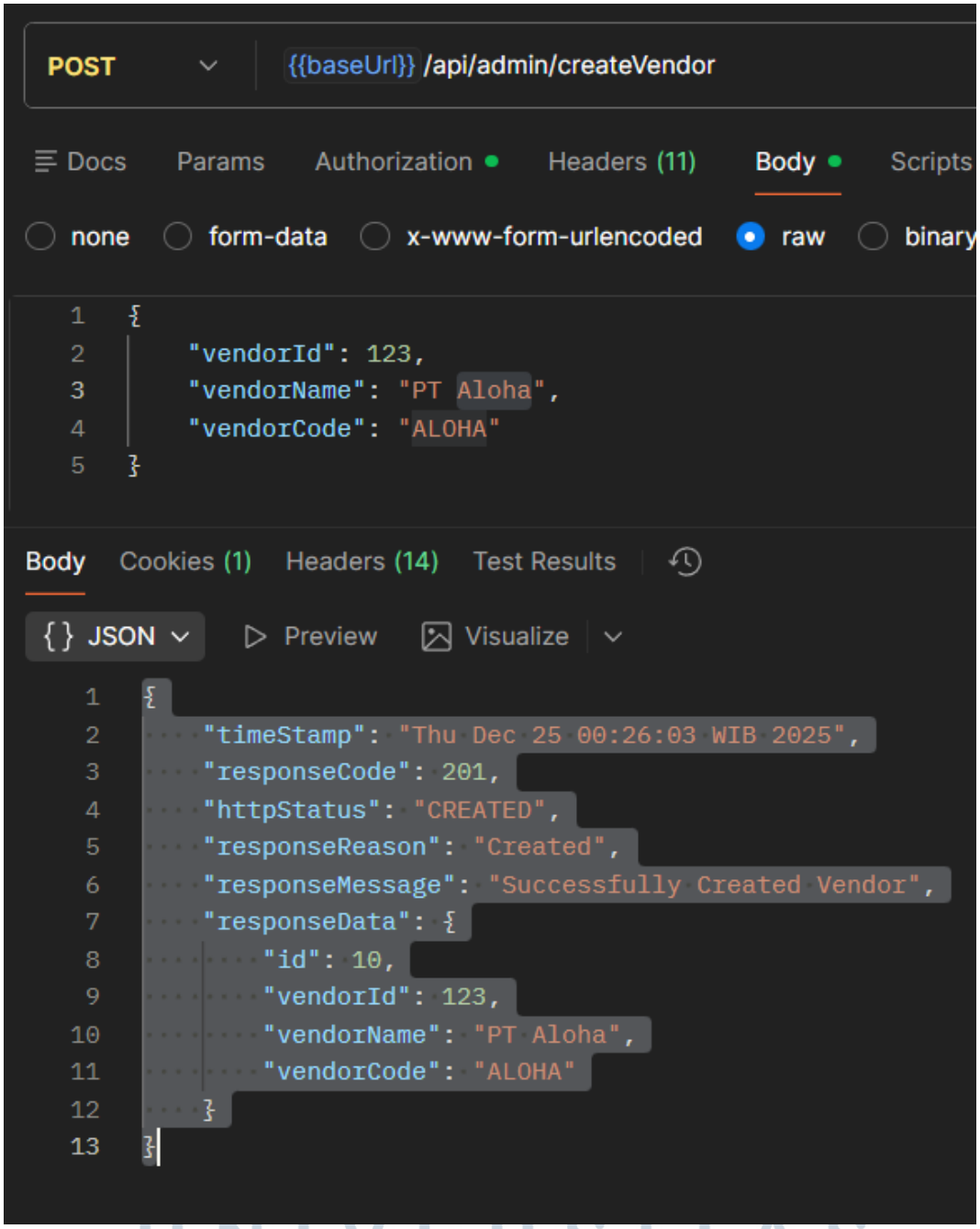
Nomor	Endpoint & Metode HTTP	Ringkasan Endpoint
6	POST `/api/admin/createUser`	Membuat User baru dan menyimpan data user ke dalam basis data JEBRET.
7	PUT `/api/admin/updateUser/{id}`	Memperbarui data User berdasarkan dari `id` yang dimasukkan ke dalam query.
8	DELETE `/api/admin/deleteUser/{id}`	Menghapus User dari basis data berdasarkan dari `id` yang dicantumkan ke dalam query.

Pada gambar 3.50, request dilakukan menggunakan metode GET ke endpoint `/api/admin/getAllVendors` dengan parameter `page=0` dan `size=5` untuk mengatur halaman serta jumlah data per halaman, `sortField=id` dan `sortOrder=asc` untuk mengurutkan data vendor berdasarkan kolom id secara menaik, serta search untuk kebutuhan pencarian jika diperlukan. Pengujian juga menggunakan *Bearer Token* pada tab *Authorization*, yang menandakan endpoint ini dilindungi dan hanya dapat diakses oleh user internal yang telah login.



Gambar 3.50. Admin Get All Vendors

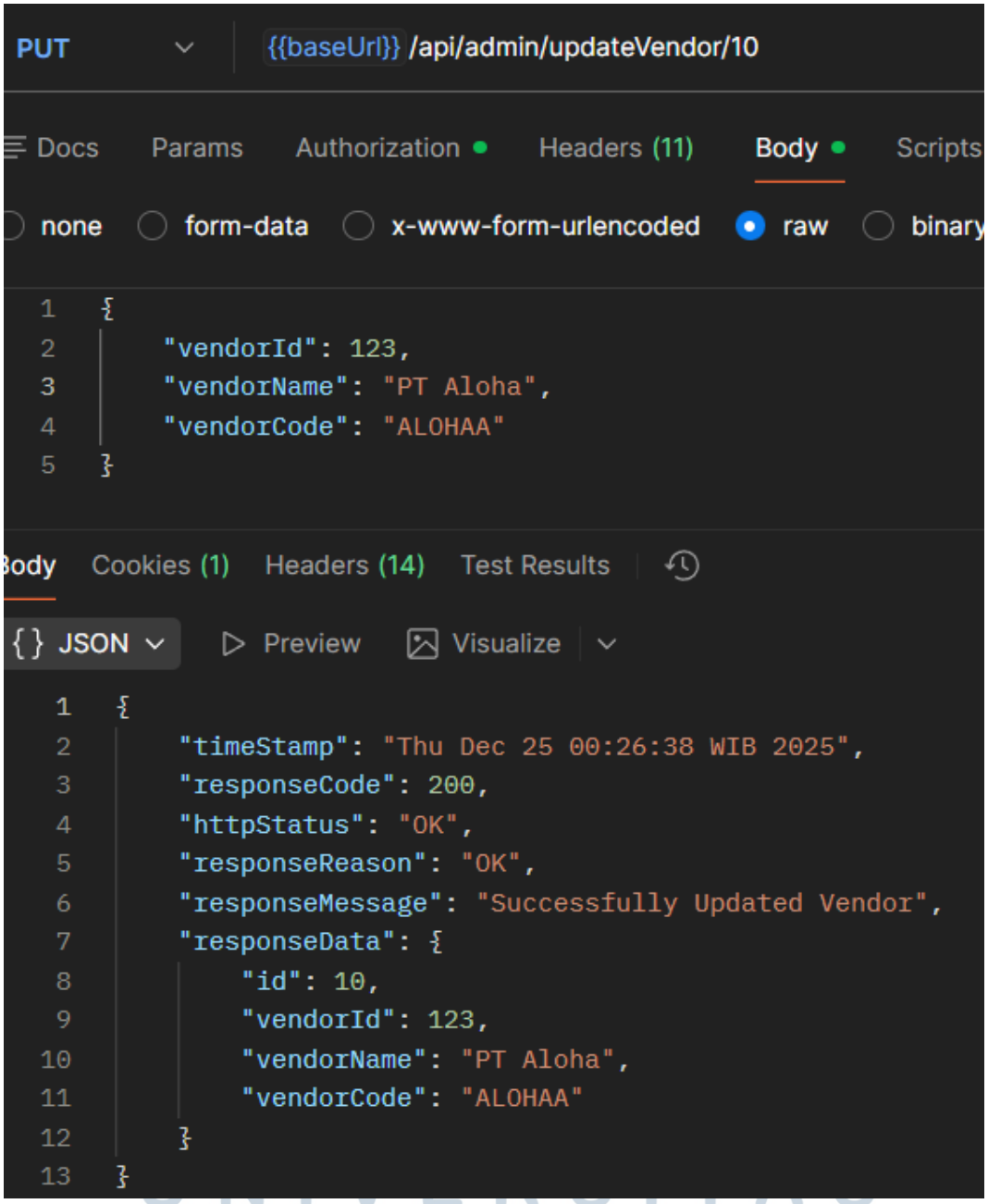
Pada gambar 3.51, request dilakukan menggunakan metode POST ke endpoint `/api/admin/createVendor` dengan body bertipe *raw* JSON yang memuat data vendor yang akan ditambahkan, seperti `vendorId`, `vendorName`, dan `vendorCode`. Setelah request dikirim, Postman menampilkan status `201 Created`, yang menunjukkan proses penambahan data vendor berhasil dilakukan oleh backend.



Gambar 3.51. Admin Create Vendor

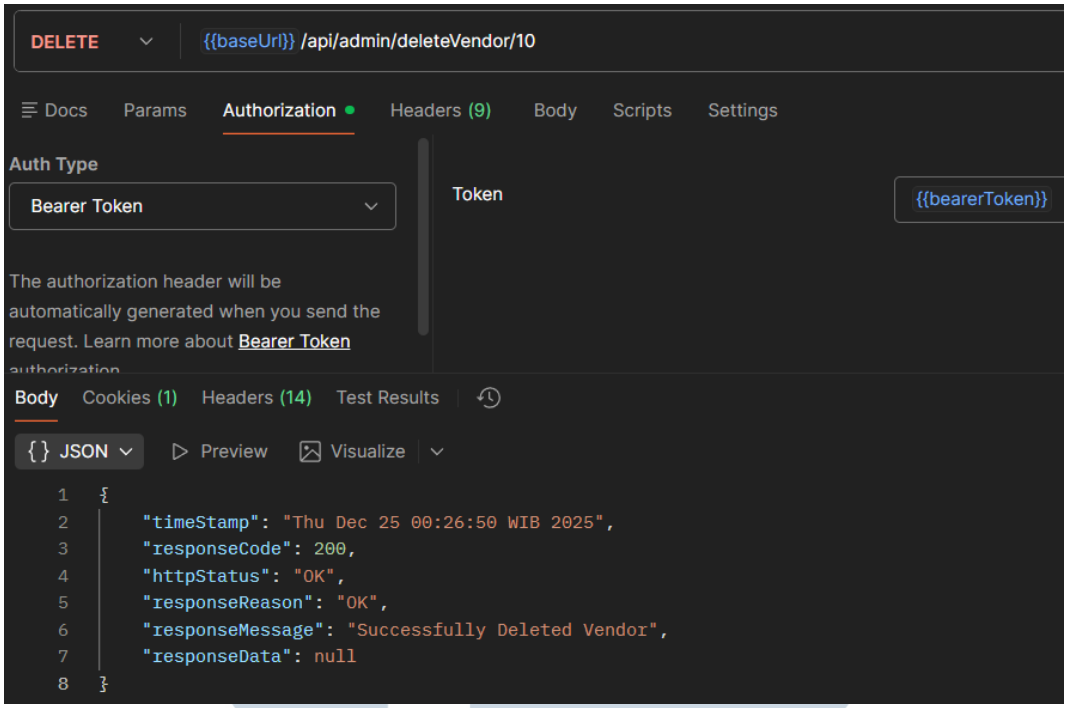
Pada gambar 3.52, request dilakukan menggunakan metode PUT ke endpoint `/api/admin/updateVendor/{id}`, yang menunjukkan bahwa parameter path `id` digunakan untuk menentukan vendor mana yang akan diperbarui. Body request dikirim dalam format *raw* JSON berisi nilai terbaru seperti `vendorId`, `vendorName`, dan `vendorCode`. Pengujian ini juga menggunakan autentikasi *Bearer Token* agar hanya user internal yang berwenang dapat melakukan perubahan

data.



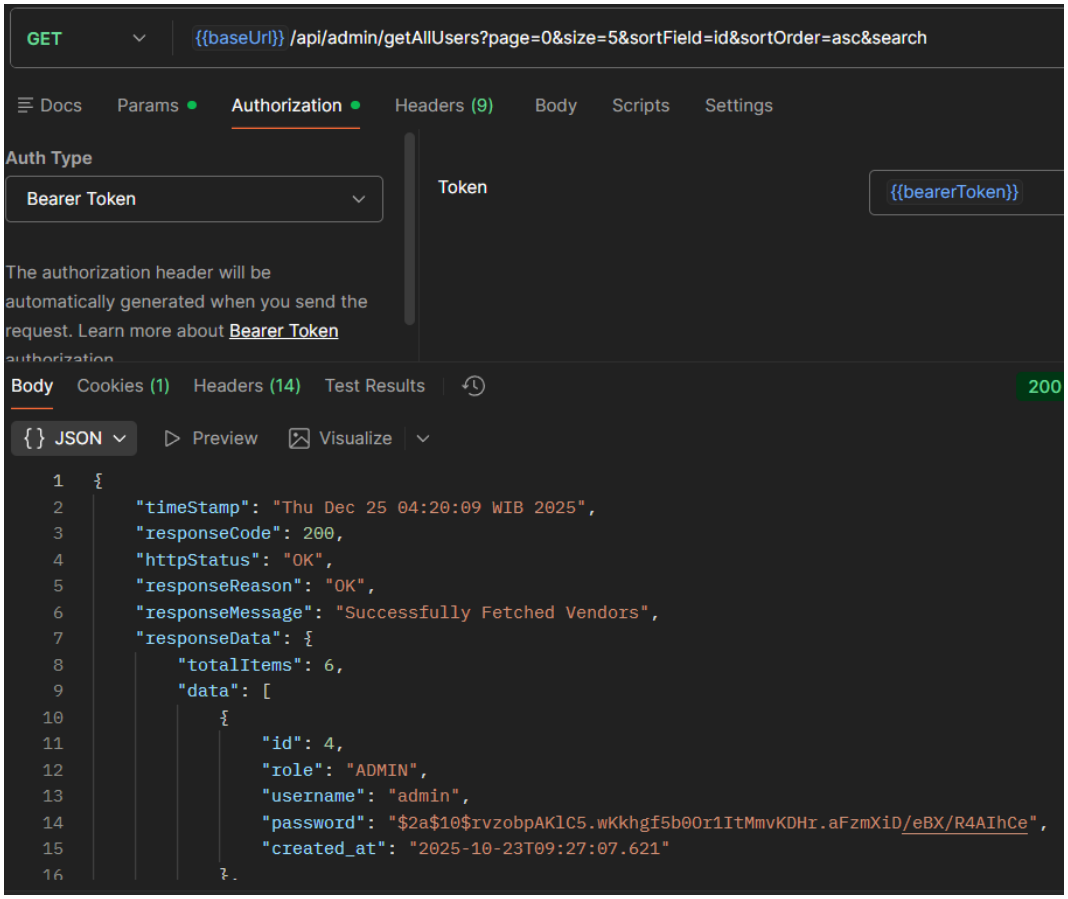
Gambar 3.52. Admin Update Vendor

Pada gambar 3.53, request dilakukan menggunakan metode DELETE ke endpoint `/api/admin/deleteVendor/{id}` dengan Bearer Token pada tab Authorization. Hal ini menunjukkan proses penghapusan vendor dilakukan berdasarkan id tertentu dan hanya dapat dieksekusi oleh user internal yang telah terautentikasi.



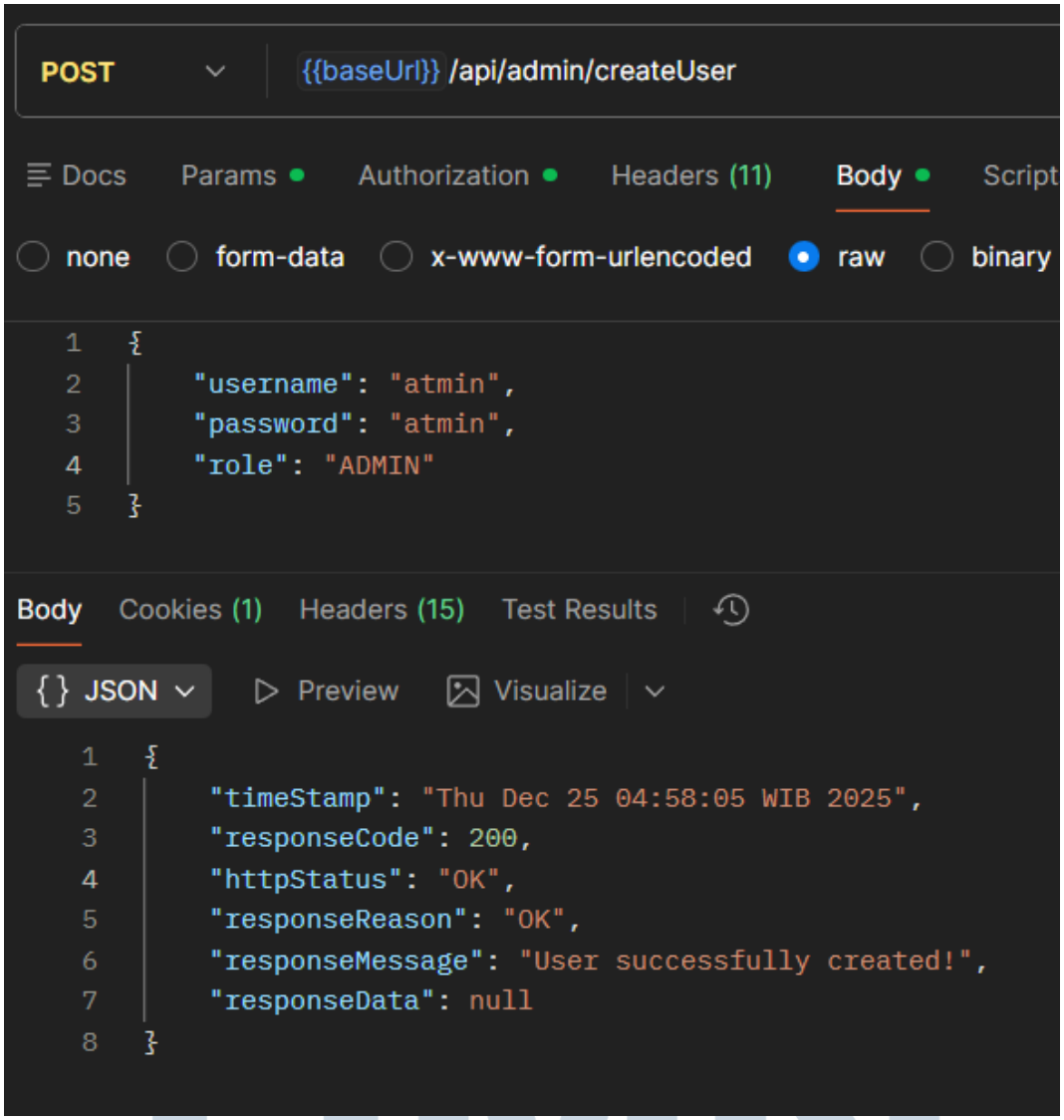
Gambar 3.53. Admin Delete Vendor

Pada gambar 3.54, request dilakukan menggunakan metode `GET` ke endpoint `/api/admin/getAllUsers` dengan parameter `page=0` dan `size=5` untuk menentukan halaman serta jumlah data per halaman, `sortField=id` dan `sortOrder=asc` untuk mengurutkan data berdasarkan `id` secara menaik, serta `search` untuk pencarian jika diperlukan. Pengujian menggunakan `Bearer Token` pada tab `Authorization`, yang menunjukkan endpoint ini dilindungi dan hanya dapat diakses oleh user internal yang telah login.



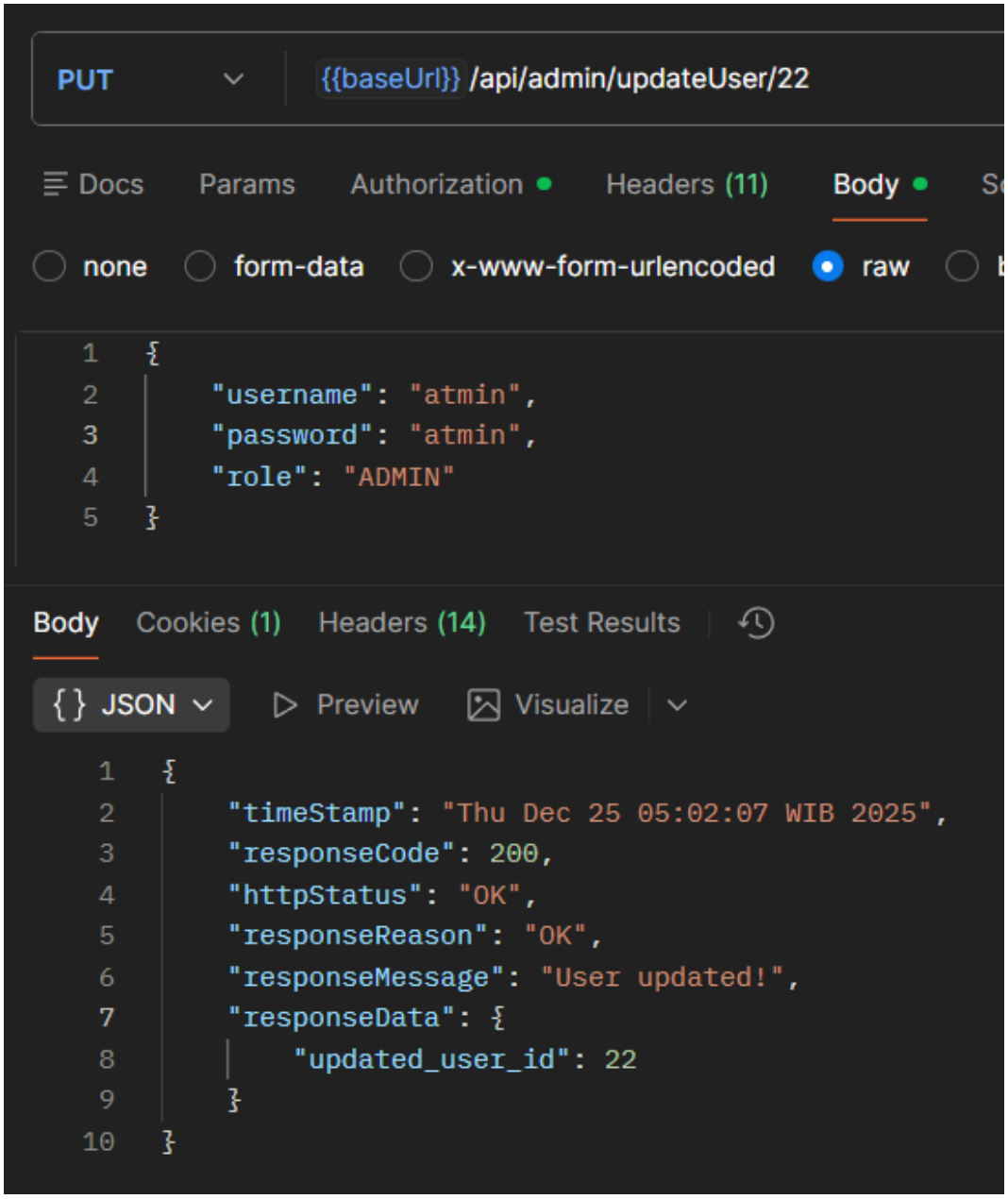
Gambar 3.54. Admin Get All Users

Pada gambar 3.55, request dilakukan menggunakan metode `POST` ke endpoint `/api/admin/createUser` dengan body bertipe `raw JSON` yang memuat `username`, `password`, dan `role`. Setelah request dikirim, Postman menampilkan status `200 OK` yang menandakan proses pembuatan user berhasil dijalankan.



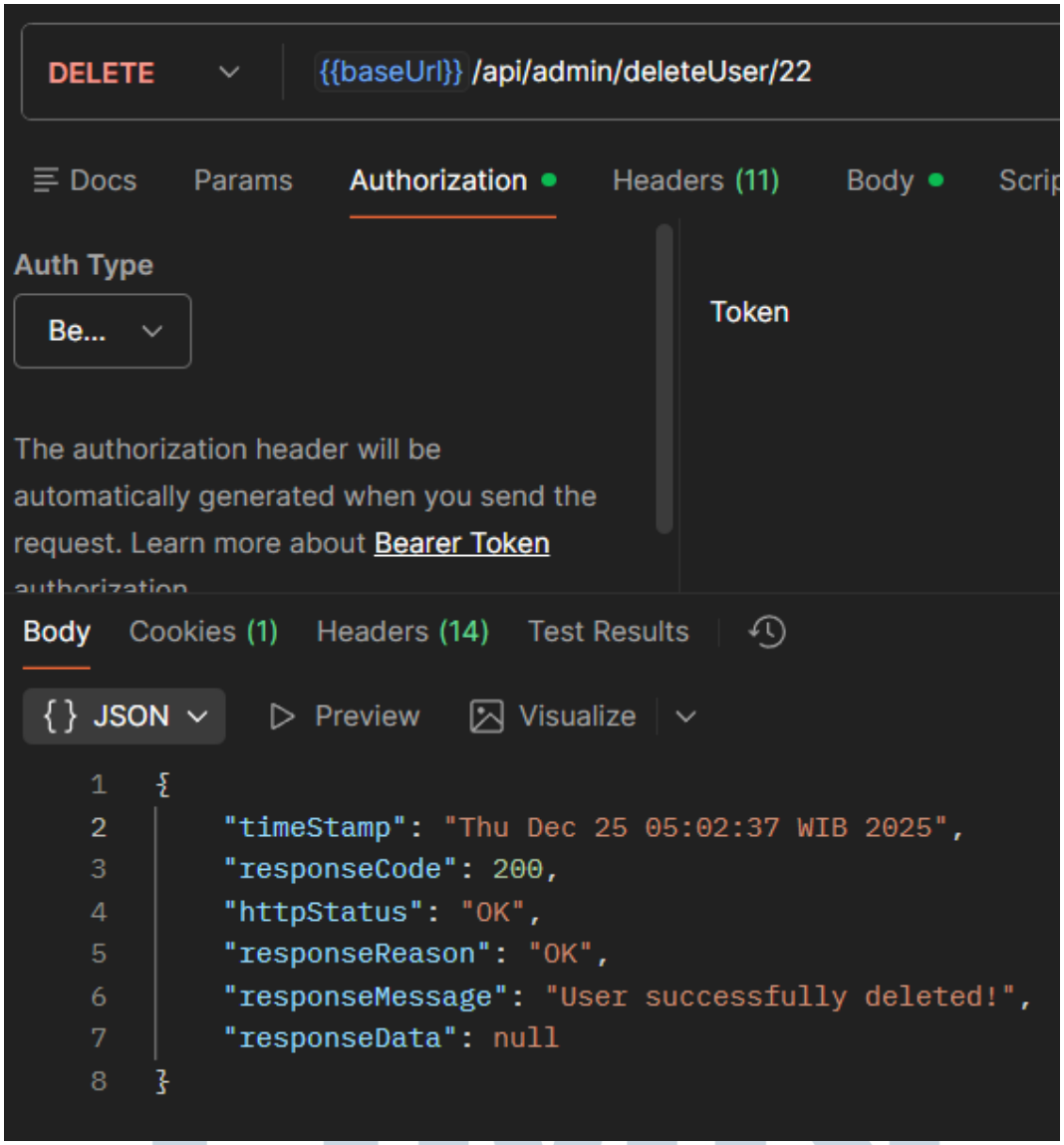
Gambar 3.55. adminCreateUser

Pada gambar 3.56, request dilakukan menggunakan metode `PUT` ke endpoint `/api/admin/updateUser/{id}`, yang menunjukkan bahwa path parameter `id` digunakan sebagai target user yang akan diperbarui. Body request dikirim dalam format `raw JSON` yang berisi `username`, `password`, dan `role` terbaru. Pengujian dilakukan dalam kondisi terautentikasi agar perubahan data hanya dapat dilakukan oleh user yang berwenang.



Gambar 3.56. Admin Update User

Pada gambar 3.57, request dilakukan menggunakan metode `DELETE` ke endpoint `/api/admin/deleteUser/{id}` dengan `Bearer Token` pada tab `Authorization`. Hal ini menunjukkan proses penghapusan dilakukan berdasarkan `id` tertentu dan hanya dapat dijalankan oleh user internal terautentikasi.



Gambar 3.57. Admin Delete User

3.4 Pengujian Sistem

Website JEBRET diuji menggunakan metode pengujian *Black Box Testing*. Pengujian ini dilakukan dalam tujuan untuk memastikan bahwa sistem yang telah dirancang dan dibangun berjalan sesuai dengan ekspektasi. Komponen yang diuji yaitu Vendor dan Admin. Dapat dilihat rincian dari pengujian *Black Box* pada tabel 3.10

Tabel 3.10. Tabel Pengujian Sistem JEBRET

Pengujian	**Skenario Yang Dilakukan**	**Hasil yang Diharapkan**	**Simpulan**
Vendor (DP Request).	Membuat Request Down Payment di halaman DP Request.	Berhasil membuat Billing Request Down Payment.	Sesuai dengan yang diharapkan.
Vendor (Payment Request).	Membuat Request Payment di halaman Payment Request.	Berhasil membuat Billing Request Payment.	Sesuai dengan yang diharapkan.
Vendor (Tracking).	Melihat progress Billing Request.	Dapat tracking progress Billing Request.	Sesuai dengan yang diharapkan.
Admin (Request)	Melakukan Approve/Rejection terhadap Billing yang di request oleh vendor.	Dapat tracking progress Billing Request.	Sesuai dengan yang diharapkan.
Admin (Upload).	Mengunggah file Bupot (Bukti Potongan) ke Billing Request.	Berhasil mengunggah file Bupot ke Billing Request.	Sesuai dengan yang diharapkan.
Admin (Report).	Mengunduh tabel dalam halaman Report menjadi file xlsx.	Berhasil mengubah dan mengunduh tabel dalam halaman report menjadi file xlsx.	Sesuai dengan yang diharapkan.
Admin (Management).	Mengelola kredensial Vendor dan User	Berhasil melakukan pembuatan, pengubahan, dan penghapusan Vendor dan User.	Sesuai dengan yang diharapkan.

3.5 Kendala dan Solusi yang Ditemukan

Selama proses perancangan dan pengembangan sistem JEBRET di PT Hexaon Business Mitrasindo, terdapat beberapa kendala yang muncul, baik pada sisi antarmuka pengguna maupun pada implementasi logika bisnis di *backend*. Kendala-kendala tersebut sempat memengaruhi kelancaran pengembangan, namun secara bertahap dapat diatasi melalui diskusi dengan pembimbing serta penyesuaian pada rancangan teknis yang digunakan.

3.5.1 Kendala yang Dihadapi

Dalam pelaksanaan kegiatan magang, beberapa kendala utama yang dihadapi antara lain sebagai berikut:

1. **Ketidakpastian desain website**
   Pada tahap awal pengembangan, rancangan tampilan website JEBRET belum memiliki acuan desain yang benar-benar jelas. Hal ini menyebabkan kebingungan dalam menentukan struktur *layout*, pemanfaatan komponen antarmuka, pemilihan skema warna, serta penyelarasan tampilan antara halaman vendor dan admin. Perbedaan sudut pandang terkait desain juga mengakibatkan perlunya beberapa kali revisi sebelum diperoleh bentuk antarmuka yang dianggap sesuai oleh pihak terkait.

2. **Potensi ketidakamanan endpoint vendor**
   Implementasi awal *endpoint* yang digunakan oleh vendor masih menyisakan potensi celah keamanan. Identitas vendor yang mengakses sistem belum divalidasi secara ketat, karena *endpoint* hanya mengandalkan `vendorId` yang dikirim melalui parameter tanpa diverifikasi terhadap identitas vendor yang sedang login. Kondisi tersebut berpotensi dimanfaatkan untuk mengakses data vendor lain apabila parameter dimanipulasi.

3. **Ketidakjelasan struktur kode backend**
   Pada saat menyusun *backend*, sempat terjadi kebingungan dalam menentukan pola pengorganisasian kode dan batas tanggung jawab tiap komponen. Beberapa logika bisnis masih tersebar dan bercampur antara *controller*, *service*, dan *repository*, sehingga struktur proyek menjadi kurang rapi. Hal ini berdampak pada berkurangnya kejelasan alur program dan menyulitkan proses pengembangan fitur lanjutan maupun pemeliharaan kode.

3.5.2 Solusi yang Diterapkan

Untuk mengatasi kendala-kendala tersebut, beberapa langkah solusi telah diterapkan selama pengembangan sistem JEBRET, yaitu sebagai berikut:

1. **Konsultasi dan penyelarasan desain website**
   Untuk mengurangi ketidakpastian desain, dilakukan sesi konsultasi berkala dengan pembimbing dan tim pengembang guna memperoleh gambaran yang lebih pasti mengenai standar tampilan yang diinginkan. Berdasarkan masukan yang diterima, dibuat kembali rancangan antarmuka dengan struktur *layout*, komponen tabel, dialog, dan elemen navigasi yang lebih konsisten. Setelah pola desain disepakati, proses implementasi *frontend* menjadi lebih terarah dan jumlah revisi dapat diminimalkan.

2. **Penerapan validasi `vendorId` berbasis JWT pada setiap request**
   Permasalahan keamanan pada *endpoint* vendor diatasi dengan menambahkan mekanisme validasi identitas vendor menggunakan token JWT [10]. Setiap permintaan dari sisi vendor akan mengambil `vendorId` dari token yang tersimpan pada sesi login, kemudian dicocokkan dengan `vendorId` yang terdapat pada *path* atau *body request*. Jika nilai tersebut tidak sesuai, *backend* akan menolak permintaan dan mengembalikan respon *error*. Dengan cara ini, akses data menjadi terbatas hanya pada vendor yang berhak, sehingga keamanan sistem meningkat.

3. **Konsultasi penataan struktur dan perbaikan organisasi kode backend**
   Ketidakjelasan struktur kode *backend* diselesaikan dengan berkonsultasi kepada pembimbing dan tim *backend* mengenai pola arsitektur yang umum digunakan di perusahaan. Berdasarkan arahan tersebut, kode *backend* disusun ulang dengan pemisahan yang lebih tegas antara lapisan *controller*, *service*, dan *repository*, serta pengelompokan kelas berdasarkan domain atau fitur (misalnya *billing*, *vendor*, dan *user*). Penataan ulang ini membuat struktur proyek menjadi lebih sistematis, memudahkan proses pembacaan kode, dan mendukung pengembangan serta pemeliharaan sistem di masa mendatang.