BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Selama proses magang, tugas dan proyek yang diberikan dilaksanakan dengan mencakup pemahaman terhadap sistem yang digunakan, pengembangan fitur, serta pelaksanaan pengujian dan *debugging*. Dalam menjalankan tugas, koordinasi dilakukan secara berkala dengan *team lead* selaku *supervisor*, yang bertanggung jawab dalam memberikan arahan dan bimbingan teknis. Selain itu, setiap aktivitas yang dilakukan terdokumentasi dengan baik sebagai bagian dari proses evaluasi. Selanjutnya dilakukan evaluasi berkala, di mana *team lead* memberikan saran dan masukkan terhadap pekerjaan yang telah diselesaikan. Evaluasi ini bertujuan untuk menilai perkembangan kerja serta memberikan rekomendasi perbaikan untuk meningkatkan kualitas hasil yang dicapai. Pada tahap akhir program magang, laporan dan dokumentasi teknis disusun, berisi ringkasan pengalaman serta kontribusi yang telah diberikan dalam proyek yang dikerjakan.

3.2 Tugas yang Dilakukan

Kontribusi yang diberikan berfokus pada pengembangan aplikasi UFINS, sebuah aplikasi *finance* yang dirancang untuk mendukung berbagai aspek operasional perusahaan melalui sejumlah modul. Beberapa modul utama dalam UFINS meliputi:

- Management Setup
- General Ledger
- Budgeting System
- Procurement System
- Account Payable System
- Tax
- Fixed Asset
- Pre Paid

- Inventory
- Vendor Management

Dalam proyek UFINS, keterlibatan difokuskan pada modul *Procurement System*. Kontribusi yang diberikan mencakup pengembangan dan penyempurnaan fitur untuk mendukung proses pengadaan *(procurement)*. *Procurement System* adalah modul yang mengelola proses pengadaan barang dan jasa secara transparan dan efisien. Proses dimulai dari registrasi *vendor*, pertemuan/penjelasan teknis proses lelang *(aanwijzing)*, hingga *upload quotation*. *Vendor* juga mengikuti klarifikasi dan *beauty contest* untuk memastikan kesesuaian penawaran dengan kebutuhan. Tahapan selanjutnya meliputi *negosiasi*, *awarding* (penentuan pemenang), revisi *quotation*, konfirmasi pengguna, dan *penyanggahan* bagi *vendor* yang tidak terpilih. Proses ini dirancang untuk menjamin keterbukaan dan akuntabilitas dalam pengadaan. Kontribusi yang diberikan meliputi pengembangan fitur *audit trail* untuk merekam seluruh aktivitas dalam sistem. Fitur ini memastikan transparansi dan integritas data, mencakup pencatatan *log*, integrasi dengan sistem, serta pengujian untuk menampilkan data secara akurat sesuai kebutuhan pengguna.

Pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

Tabel 3.1. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang

Minggu Ke -	Pekerjaan yang dilakukan
1	Penyesuaian dan perkenalan tech stack yang digunakan di tempat magang.
2	Memulai pengerjaan service Audit Trail dengan fokus pada API Audit Trail Transaction Flow.
3	Implementasi fitur reporting untuk API Audit Trail Transaction Flow yang melibatkan Excel builder, query builder, dan repository untuk Audit Trail Transaction Flow, untuk memfasilitasi query atau akses ke database, serta membangun struktur tabel dan kolom dalam file Excel.
4	Menyempurnakan struktur file Excel dengan membangun tabel dan kolom secara dinamis berdasarkan data dari Audit Trail Transaction Flow.
5	Melakukan debugging menyeluruh terhadap seluruh implementasi API Audit Trail Transaction Flow, mulai dari controller, DAO, hingga repository dan class lain yang terkait.

Tabel 3.1. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang (lanjutan)

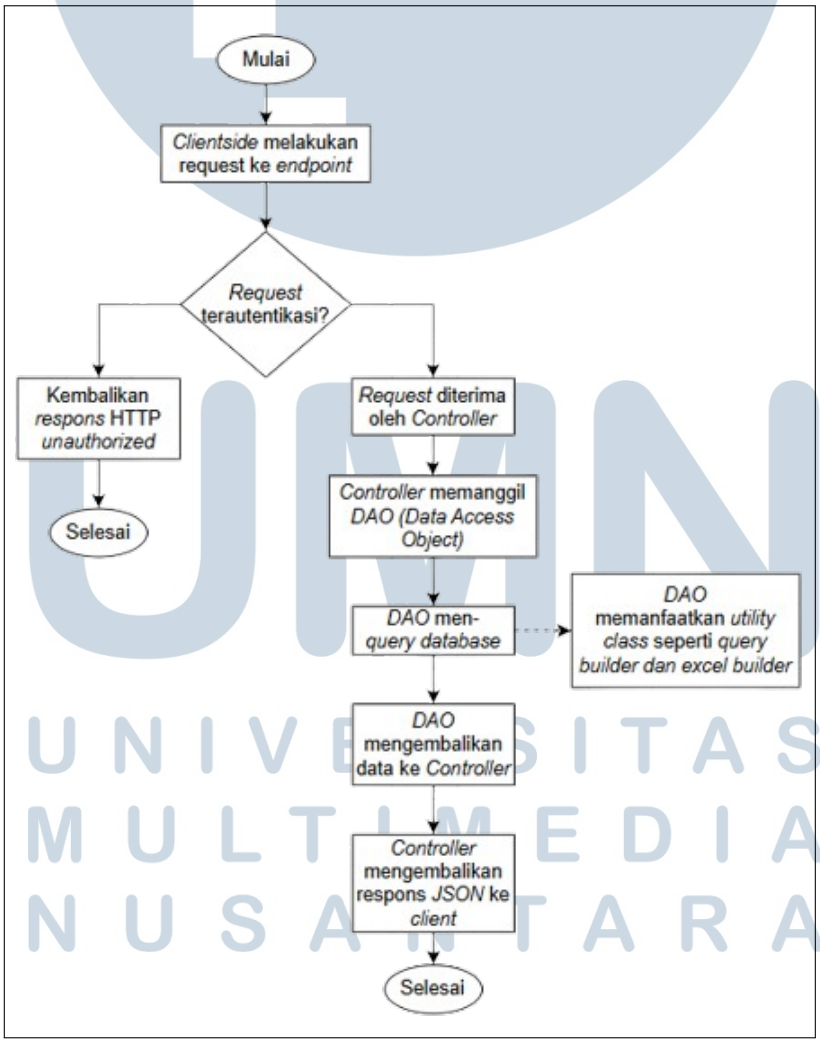
Minggu Ke -	Pekerjaan yang dilakukan
6	Memulai pengerjaan API Audit Trail Step Workflow dalam service Audit Trail.
7	Mengimplementasikan fitur reporting yang mencakup pembuatan Excel builder, query builder, serta repository untuk mengakses database dan membangun struktur tabel serta kolom pada file Excel.
8	Membuat dan menyempurnakan template laporan PDF menggunakan Jaspersoft Studio untuk diimplementasikan ke dalam code backend.
9	Melakukan debugging dan testing menyeluruh terhadap seluruh implementasi API Audit Trail Step Workflow, dan revisi API agar dapat menampilkan field workflowName yang berasal dari master_bidding_workflow_h melalui join tabel dan pengeditan pada query builder dan Excel builder.
10	Memulai pengerjaan API Audit Trail Contract dalam service Audit Trail.
11	Mengimplementasikan fitur reporting dengan menggunakan Excel builder, query builder, dan repository untuk mendukung akses database serta membentuk struktur tabel dan kolom pada file Excel.
12	Membuat dan menyempurnakan template laporan PDF menggunakan Jaspersoft Studio untuk diimplementasikan ke dalam code backend.
13	Melakukan debugging dan testing menyeluruh terhadap seluruh implementasi API Audit Trail Contract.
14	Memulai pengerjaan API Audit Parameter.
15	Mengimplementasikan fitur reporting dengan menggunakan Excel builder, query builder, dan repository untuk mendukung akses database serta membentuk struktur tabel dan kolom pada file Excel.
16	Melakukan debugging dan testing menyeluruh terhadap seluruh implementasi API Audit Parameter.

3.3 Uraian Pelaksanaan Magang

Dalam sistem UFINS, *audit trail* merupakan fitur penting yang berfungsi untuk merekam seluruh aktivitas atau perubahan data yang dilakukan oleh pengguna di berbagai modul sistem. Fitur ini berperan dalam menjaga transparansi, akuntabilitas, serta memudahkan proses pelacakan apabila terjadi kesalahan atau

penyalahgunaan. Pencatatan dilakukan secara otomatis setiap kali terjadi interaksi dengan sistem, baik melalui proses input, perubahan, maupun penghapusan data. Modul-modul yang tercakup dalam *audit trail* mencakup berbagai aspek utama dalam UFINS, seperti pengelolaan anggaran, kontrak, aset, hingga proses persetujuan dan login pengguna. Data hasil pencatatan ini kemudian dapat disajikan dalam bentuk laporan menggunakan *JasperReports*, sehingga mempermudah proses audit internal maupun eksternal. Dengan demikian, *audit trail* menjadi salah satu komponen krusial dalam mendukung keamanan dan integritas data dalam aplikasi.

Untuk memahami proses kerja fitur *audit trail* dalam modul UFINS, dibuat sebuah *flowchart* yang menggambarkan alur komunikasi antara *client* dan *server* saat permintaan data audit dilakukan.



Gambar 3.1. Bagan Alur Proses Komunikasi API Audit Trail

Alur dimulai saat *client* mengirimkan *request* ke *endpoint* tertentu. Permintaan ini diperiksa untuk memastikan bahwa pengguna telah terautentikasi melalui validasi *token* JWT. Jika token tidak valid atau pengguna tidak memiliki hak akses, sistem akan langsung mengembalikan respons *Unauthorized* dan proses berhenti. Namun jika autentikasi berhasil, *controller* akan meneruskan permintaan ke lapisan *DAO* (Data Access Object), yang bertugas membangun kriteria pencarian secara dinamis dan mengambil data dari *database* sesuai dengan parameter yang diberikan. Data hasil pencarian dikembalikan ke *controller* dan kemudian diformat ke dalam bentuk yang sesuai, seperti *JSON*, PDF, atau Excel, tergantung pada jenis *endpoint* yang dipanggil. Hasil akhir ini dikirim kembali ke *client* sebagai respons.

3.3.1 Audit Trail Transaction Flow

Fokus pekerjaan selama magang diarahkan pada pengembangan fitur *audit trail transaction flow* dalam modul Procurement System UFINS. Tiga *method* baru ditambahkan ke dalam `AudittrailController`, yaitu `viewAudittrailTF`, `downloadAudittrailTF`, dan `downloadAudittrailTFExcel`. Ketiganya bertanggung jawab untuk menampilkan serta mengunduh data audit berdasarkan permintaan pengguna.

- `viewAudittrailTF`:
  Pada Kode 3.1 terdapat metode `viewAudittrailTF`. Metode ini menangani permintaan POST yang ditujukan untuk melihat data audit trail transaction flow. Proses dimulai dengan validasi *JWT token* dari *header* permintaan guna memastikan autentikasi. Setelah itu, dilakukan *query* ke database melalui objek auditDAO berdasarkan kriteria yang diberikan. Bila data ditemukan, maka akan dikembalikan bersama total jumlah entri. Bila tidak, respons yang menunjukkan data kosong dikembalikan.

```
1     @SuppressWarnings("unchecked")
2   @RequestMapping(value = AudittrailConstant.
      PROCUREMENT_AUDITTRAIL_FLOW_TRANSACTION_VIEW, method =
      RequestMethod.POST)
3   @CrossOrigin(origins = "*")
4   @Logging
5   @Authenticate
6   public String viewAudittrailTF (HttpServletRequest request,
      HttpServletResponse response, @RequestBody View View)
      throws Exception {
```

```
7     try
8     {
9       String tokens = request.getHeader(
      AuthenticationConstant.AUTHORIZATION);
10      String nezam = request.getHeader(AuthenticationConstant
      .NEZNAM);
11      authRes = AuthenticationApi.authentication(
      authentication, tokens);
12      if (authRes == null) {
13        return senderAuditTransactionFlow.
      responseNotAuthorized(request, response);
14      }
15      if (!nezam.equalsIgnoreCase(authRes.
      getAuthenticatedMethod())) {
16        return senderAuditTransactionFlow.
      responseNotAuthorized(request, response);
17      }
18      Map<String, Object> data = auditDAO.
      getDataAudittrailTransactionFlowByCriteria(View.getSearch
      (), View.getStatus(), View.getStartDate(),View.getEndDate
      (), View.getBranchid(), authRes.getBranchId());
19      List<AudittrailTransactionFlow> listAuditTF = (List<
      AudittrailTransactionFlow>) data.get("data");
20      long total = (long)data.get("total");
21      if (total == 0) {
22        return senderAuditTransactionFlow.responseEmpty(
      request, response);
23      }
24      return senderAuditTransactionFlow.responseListCount(
      request, response, listAuditTF, total);
25    }
26    catch (Exception e) {
27      e.printStackTrace();
28      return senderAuditTransactionFlow.
      sendErrorInternalServer(request, response, e, logger);
29    }
30  }
31
```

Kode 3.1: Metode viewAudittrailTF

- `downloadAudittrailTF`:

  Pada Kode 3.2 terdapat metode `downloadAudittrailTF`. Metode ini

digunakan untuk menghasilkan dan mengunduh laporan *Audit Trail Transaction Flow* dalam format PDF, berdasarkan parameter pencarian seperti `search`, `status`, `startDate`, `endDate`, dan `branchId`. Setelah menerima permintaan, sistem melakukan *autentikasi* menggunakan token dari header request melalui `AuthenticationApi` untuk memperoleh `userId` dan `branchId`. Selanjutnya, tanggal proses diambil dari database dan diformat, kemudian data audit yang sesuai dicari dengan method `getDataAudittrailTransactionFlowByCriteria()`. Data ini diubah menjadi objek `AudittrailTransactionFlow`, lalu dikombinasikan dengan parameter laporan seperti. Template laporan .jrxml dikompilasi dan diisi menggunakan *JasperReports*, menghasilkan objek JasperPrint yang kemudian dikonversi menjadi file PDF.

```
1    @SuppressWarnings({ "unused", "unchecked" })
2   @RequestMapping(value = AudittrailConstant.
     PROCUREMENT_AUDITTRAIL_FLOW_TRANSACTION_DOWNLOAD_PDF,
     method = RequestMethod.POST)
3   @CrossOrigin(origins = "*")
4   @Logging
5   @Authenticate
6   public void downloadAudittrailTF (HttpServletRequest
     request, HttpServletResponse response, @RequestBody View
     View) throws Exception {
7     String tokens = request.getHeader(AuthenticationConstant.
     AUTHORIZATION);
8     authRes = AuthenticationApi.authentication(authentication
     , tokens);
9     SimpleDateFormat dateFormat3 = new SimpleDateFormat("yyyy
     -MM-dd HH:mm:ss");
10    ProcessDate pros = reportdao.getProcessDate();
11    Date prosdate = SetProcessDate.getprosdate(pros);
12    String datetime = dateFormat3.format(prosdate);
13    String userid = authRes.getUserId();
14    String startDate = View.getStartDate();
15    String endDate = View.getEndDate();
16    List<JasperPrint> jasperPrints = new ArrayList<>();
17    Map<String, Object> data = auditDAO.
     getDataAudittrailTransactionFlowByCriteria(View.getSearch
     (), View.getStatus(), View.getStartDate(),View.getEndDate
     (), View.getBranchid(), authRes.getBranchId());
18    List<AudittrailTransactionFlow> listAuditTF = (List<
     AudittrailTransactionFlow>) data.get("data");
```

```
19
20      Map<String, Object> parameters = new HashMap<String,
        Object>();
21      parameters.put("P_USER", userid);
22      parameters.put("P_PROCESSDATE", datetime);
23      parameters.put("P_STARTDATE", startDate);
24      parameters.put("P_ENDDATE", endDate);
25      String fileName = templateLocation + "
        Audittrail_Transaction_Flow.jrxml";
26      JasperReport jp = JasperCompileManager.compileReport(
        fileName);
27      JasperPrint print = JasperFillManager.fillReport(jp,
        parameters, new JRBeanCollectionDataSource(listAuditTF));
28
29      jasperPrints.add(print);
30
31      PDF pdf = new PDF(response, "REPORT_AUDITTRAIL_TF_",".pdf
        ",print);
32    }
33
```

Kode 3.2: Metode downloadAudittrailTF

- `downloadAudittrailTFExcel`:
  Pada Kode 3.3 terdapat metode `downloadAudittrailTFExcel`. Metode ini menangani pengunduhan data dalam format Excel. Alur prosesnya mirip dengan PDF, namun menggunakan *generator* Excel untuk membuat file dan mengembalikannya dalam bentuk `InputStreamResource`.

```
1     @SuppressWarnings({ "unused", "unchecked" })
2   @RequestMapping(value = AudittrailConstant.
      PROCUREMENT_AUDITTRAIL_FLOW_TRANSACTION_DOWNLOAD_EXCEL,
      method = RequestMethod.POST)
3   @CrossOrigin(origins = "*")
4   @Logging
5   @Authenticate
6   public ResponseEntity<InputStreamResource>
      downloadAudittrailTFExcel(HttpServletRequest request,
      HttpServletResponse response, @RequestBody View View)
      throws Exception {
7     String tokens = request.getHeader(AuthenticationConstant.
      AUTHORIZATION);
8     authRes = AuthenticationApi.authentication(authentication
      , tokens);
```

```
9     Map<String, Object> data = auditDAO.
      getDataAudittrailTransactionFlowByCriteria(View.getSearch
      (), View.getStatus(), View.getStartDate(), View.getEndDate
      (), View.getBranchid(), authRes.getBranchId());
10    List<AudittrailTransactionFlow> listAuditTF = (List<
      AudittrailTransactionFlow>) data.get("data");
11    String filename = "REPORT_AUDITTRAIL_TF_";
12    Date now = new Date();
13    String dateMerge = new SimpleDateFormat(ParameterConstant
      .DATETIME_YMDHMSSSS_MERGE).format(now);
14    String fileExportName = filename + dateMerge + ".xlsx";
15    logger.info("Start Print EXCEL Data Audittrail TF " + new
      Date().getTime());
16    InputStreamResource file = new InputStreamResource(
      auditDAO.loadListDataAudittrailTransactionFlowDownload(
      listAuditTF));
17    logger.info("End Print EXCEL Data Audittrail TF " + new
      Date().getTime());
18    return ResponseEntity.ok().header(HttpHeaders.
      CONTENT_DISPOSITION, "attachment; filename=" +
      fileExportName).contentType(MediaType.parseMediaType("
      application/vnd-ms.excel")).body(file);
19  }
20
```

Kode 3.3: Metode downloadAudittrailTFExcel

Kemudian, dikembangkan *method* baru dalam class `AudittrailDAOImpl` untuk mengambil dan menghitung data berdasarkan kriteria pencarian tertentu yaitu `getDataAudittrailTransactionFlowByCriteria` dan `loadListDataAudittrailTransactionFlowDownload`. Potongan kode dapat dilihat pada Kode 3.4

- `getDataAudittrailTransactionFlowByCriteria`:
  Method ini digunakan untuk mengambil data berdasarkan beberapa kriteria seperti `search`, `status`, `startDate`, `endDate`, `branch`, dan `branchAuth`. Hasil dari *method* ini berupa `Map` yang berisi total data dan daftar hasil pencarian.

- `loadListDataAudittrailTransactionFlowDownload`:
  Method ini mengubah daftar objek `AudittrailTransactionFlow` menjadi format Excel yang siap diunduh sebagai `InputStream`.

```
1     @Override
2   public Map<String, Object>
      getDataAudittrailTransactionFlowByCriteria(String search,
      String status, String startDate, String endDate, List<
      String> branch, String branchAuth) throws AppException {
3     // TODO Auto-generated method stub
4     Map<String, Object> retValue = new HashMap<String, Object
      >();
5     Map<String, Object> criteria = new HashMap<String, Object
      >();
6     List<AudittrailTransactionFlow> list = new ArrayList<
      AudittrailTransactionFlow>();
7     auditTransactionFlowRepository.findAll(
      builderAuditTransactionFlow.
      queryBuilderAudittrailTransactionFlow(criteria,
8           search,
9           status,
10          startDate,
11          endDate,
12          branch,
13          branchAuth)).forEach(audittrailId -> list.add(
      audittrailId));
14    long t = auditTransactionFlowRepository.count(
      builderAuditTransactionFlow.
      queryBuilderAudittrailTransactionFlow(criteria,
15          search,
16          status,
17          startDate,
18          endDate,
19          branch,
20          branchAuth));
21    retValue.put("total", t);
22    retValue.put("data", list);
23    return retValue;
24  }
25
26  @Override
27  public InputStream
      loadListDataAudittrailTransactionFlowDownload(List<
      AudittrailTransactionFlow> listData) throws AppException {
28    // TODO Auto-generated method stub
29    ByteArrayInputStream in =
      ExcelBuilderAudittrailTransactionFlow.
```

```
        ParameterAudittrailTransactionFlowToExcel(listData);
30      return in;
31    }
32
```

Kode 3.4: Metode getDataAudittrailTransactionFlowByCriteria

Kemudian, diimplementasikan *class* baru bernama `ExcelBuilderAudittrailTransactionFlow`, implementasi *method* `queryBuilderAudittrailTransactionFlow` dalam *class* `QueryBuilderAudittrail`, dan membuat *class* `AudittrailTransactionFlowRepository`.

- `ExcelBuilderAudittrailTransactionFlow`:
  Class ini berfungsi sebagai utilitas untuk membangun file Excel dari data transaksi audit. Disediakan *method* `ParameterAudittrailTransactionFlowToExcel` untuk mengatur isi dan format file, serta `hasExcelFormat` untuk validasi. Hasil atau *screenshot* dari kode dapat dilihat pada Lampiran 7.

- `queryBuilderAudittrailTransactionFlow`:
  Method `queryBuilderAudittrailTransactionFlow` membangun `Specification` dinamis dengan menggabungkan beberapa `Predicate` berdasarkan `search`, `status`, `transactionDate`, serta `branchAuth`. *Field* seperti `title`, `userid`, dan `rfqNo` difilter menggunakan `cb.like`, sementara filter status, cabang, dan tanggal menggunakan `cb.equal` atau `cb.between`. Hasil pencarian diurutkan berdasarkan `transactionDate` secara menurun. Hasil atau *screenshot* dari kode dapat dilihat pada Lampiran 8.

3.3.2 Audit Trail Step Workflow

Pengembangan *API audit trail step workflow* pada modul *Procurement System UFINS* dimulai dengan menambahkan tiga *method* ke dalam `AudittrailController`, yaitu `viewAudittrailSW`, `downloadAudittrailSW`, dan `downloadAudittrailSWExcel`.

- `viewAudittrailSW`:
  Pada Kode 3.5 terdapat metode `viewAudittrailSW`. Metode ini menangani

permintaan POST untuk menampilkan data *Audit Trail Step Workflow* dengan validasi token JWT sebagai autentikasi pengguna. Data diambil melalui metode `getDataAudittrailStepWorkflowByCriteria()` dengan parameter `search`, `status`, `startDate`, `endDate`, dan `branchId`. Data tersebut kemudian dikonversi menjadi daftar `AudittrailStepWorkflow`, dihitung totalnya, dan dikembalikan dalam format *list with count*. Jika tidak ditemukan data yang sesuai, sistem akan mengembalikan respons kosong.

```
1     @SuppressWarnings("unchecked")
2     @RequestMapping(value = AudittrailConstant.
      PROCUREMENT_AUDITTRAIL_STEP_WORKFLOW_VIEW, method =
      RequestMethod.POST)
3     @CrossOrigin(origins = "*")
4     @Logging
5     @Authenticate
6     public String viewAudittrailSW(HttpServletRequest request
      , HttpServletResponse response, @RequestBody View View)
      throws Exception {
7         try {
8             String tokens = request.getHeader(
      AuthenticationConstant.AUTHORIZATION);
9             String nezam = request.getHeader(
      AuthenticationConstant.NEZNAM);
10            authRes = AuthenticationApi.authentication(
      authentication, tokens);
11            if (authRes == null || !nezam.equalsIgnoreCase(
      authRes.getAuthenticatedMethod())) {
12                return senderAuditStepWorkflow.
      responseNotAuthorized(request, response);
13            }
14
15            Map<String, Object> data = auditDAO.
      getDataAudittrailStepWorkflowByCriteria(
16                View.getSearch(), View.getStatus(), View.
      getStartDate(),
17                View.getEndDate(), View.getBranchid(),
      authRes.getBranchId());
18
19            List<AudittrailStepWorkflow> listAuditSW = (List<
      AudittrailStepWorkflow>) data.get("data");
20            long total = (long) data.get("total");
21
22            if (total == 0) {
```

```
23                  return senderAuditStepWorkflow.responseEmpty(
    request, response);
24              }
25
26              return senderAuditStepWorkflow.responseListCount(
    request, response, listAuditSW, total);
27          } catch (Exception e) {
28              e.printStackTrace();
29              return senderAuditStepWorkflow.
    sendErrorInternalServer(request, response, e, logger);
30          }
31      }
32
```

Kode 3.5: Metode viewAudittrailSW

- `downloadAudittrailSW`:

  Pada Kode 3.6 terdapat metode `downloadAudittrailSW`. Metode ini menghasilkan file PDF dari data *Audit Trail Step Workflow* berdasarkan parameter pencarian `search`, `status`, `startDate`, `endDate`, dan `branchId`. Setelah proses autentikasi pengguna, sistem mengambil tanggal proses terkini melalui utilitas `getProcessDate()` dan memformatnya menjadi `processDate`. Informasi ini, bersama `userId`, dimasukkan ke dalam parameter laporan. Data hasil pencarian dikonversi menjadi daftar `AudittrailStepWorkflow`, lalu dikaitkan dengan template JasperReports berformat `.jrxml`. Template ini dikompilasi dan diisi menggunakan `JasperCompileManager` serta `JasperFillManager` sehingga menghasilkan objek `JasperPrint`. File PDF kemudian dihasilkan dari objek ini dan dikirim ke *client* menggunakan utilitas `PDF`.

```
1       @SuppressWarnings({ "unused", "unchecked" })
2       @RequestMapping(value = AudittrailConstant.
    PROCUREMENT_AUDITTRAIL_STEP_WORKFLOW_DOWNLOAD_PDF, method
    = RequestMethod.POST)
3       @CrossOrigin(origins = "*")
4       @Logging
5       @Authenticate
6       public void downloadAudittrailSW(HttpServletRequest
    request, HttpServletResponse response, @RequestBody View
    View) throws Exception {
7           SimpleDateFormat dateFormat3 = new SimpleDateFormat("
    yyyy-MM-dd HH:mm:ss");
```

```
8         ProcessDate pros = reportdao.getProcessDate();
9         Date prosdate = SetProcessDate.getprosdate(pros);
10        String datetime = dateFormat3.format(prosdate);
11        String userid = authRes.getUserId();
12
13        Map<String, Object> data = auditDAO.
    getDataAudittrailStepWorkflowByCriteria(
14                View.getSearch(), View.getStatus(), View.
    getStartDate(),
15                View.getEndDate(), View.getBranchid(), authRes.
    getBranchId());
16
17        List<AudittrailStepWorkflow> listAuditSW = (List<
    AudittrailStepWorkflow>) data.get("data");
18
19        Map<String, Object> parameters = new HashMap<>();
20        parameters.put("P_USER", userid);
21        parameters.put("P_PROCESSDATE", datetime);
22        parameters.put("P_STARTDATE", View.getStartDate());
23        parameters.put("P_ENDDATE", View.getEndDate());
24
25        String fileName = templateLocation + "
    Audittrail_Step_Workflow.jrxml";
26        JasperReport jp = JasperCompileManager.compileReport(
    fileName);
27        JasperPrint print = JasperFillManager.fillReport(jp,
    parameters, new JRBeanCollectionDataSource(listAuditSW));
28
29        PDF pdf = new PDF(response, "REPORT_AUDITTRAIL_SW_",
    ".pdf", print);
30     }
31
```

Kode 3.6: Metode downloadAudittrailSW

- `downloadAudittrailSWExcel`:

  Pada Kode 3.7 terdapat metode `downloadAudittrailSWExcel`. Metode ini menangani permintaan pengunduhan laporan *Audit Trail Step Workflow* dalam format Excel. Setelah proses autentikasi, parameter pencarian dari objek `View` seperti `search`, `status`, `startDate`, `endDate`, dan `branchId` digunakan untuk mengambil data dari database melalui metode `getDataAudittrailStepWorkflowByCriteria()`. Data yang diperoleh

dikonversi menjadi daftar objek `AudittrailStepWorkflow` dan diproses oleh utilitas `loadListDataAudittrailStepWorkflowDownload()` untuk menghasilkan file Excel. File tersebut dibungkus dalam objek `InputStreamResource` dan dikirimkan ke *client* sebagai `ResponseEntity`.

```
1   @SuppressWarnings({ "unused", "unchecked" })
2   @RequestMapping(value = AudittrailConstant.
    PROCUREMENT_AUDITTRAIL_STEP_WORKFLOW_DOWNLOAD_EXCEL,
    method = RequestMethod.POST)
3   @CrossOrigin(origins = "*")
4   @Logging
5   @Authenticate
6   public ResponseEntity<InputStreamResource>
    downloadAudittrailSWExcel(HttpServletRequest request,
    HttpServletResponse response, @RequestBody View View)
    throws Exception {
7       String tokens = request.getHeader(
    AuthenticationConstant.AUTHORIZATION);
8       authRes = AuthenticationApi.authentication(
    authentication, tokens);
9
10      Map<String, Object> data = auditDAO.
    getDataAudittrailStepWorkflowByCriteria(
11          View.getSearch(), View.getStatus(), View.
    getStartDate(),
12          View.getEndDate(), View.getBranchid(), authRes.
    getBranchId());
13
14      List<AudittrailStepWorkflow> listAuditSW = (List<
    AudittrailStepWorkflow>) data.get("data");
15
16      String filename = "REPORT_AUDITTRAIL_SW_";
17      String dateMerge = new SimpleDateFormat(
    ParameterConstant.DATETIME_YMDHMSSSS_MERGE).format(new
    Date());
18      String fileExportName = filename + dateMerge + ".xlsx
    ";
19
20      logger.info("Start Print EXCEL Data Audittrail SW " +
     new Date().getTime());
21      InputStreamResource file = new InputStreamResource(
    auditDAO.loadListDataAudittrailStepWorkflowDownload(
    listAuditSW));
```

```
22          logger.info("End Print EXCEL Data Audittrail SW " +
    new Date().getTime());
23
24          return ResponseEntity.ok()
25                  .header(HttpHeaders.CONTENT_DISPOSITION, "
    attachment; filename=" + fileExportName)
26                  .contentType(MediaType.parseMediaType("
    application/vnd-ms.excel"))
27                  .body(file);
28      }
29
```

Kode 3.7: Metode downloadAudittrailSWExcel

Kemudian, dikembangkan *method* baru dalam *class* `AudittrailDAOImpl` untuk mengambil dan menghitung data berdasarkan kriteria pencarian tertentu yaitu `getDataAudittrailStepWorkflowByCriteria` dan `loadListDataAudittrailStepWorkflowDownload`. Potongan kode dapat dilihat pada Kode 3.8.

- `getDataAudittrailStepWorkflowByCriteria`:
  Method ini digunakan untuk mengambil data berdasarkan beberapa kriteria seperti `search`, `status`, `startDate`, `endDate`, `branch`, dan `branchAuth`. Hasil dari *method* ini berupa `Map` yang berisi total data dan daftar hasil pencarian.

- `loadListDataAudittrailStepWorkflowDownload`:
  Method ini mengubah daftar objek `AudittrailTransactionFlow` menjadi format Excel yang siap diunduh sebagai `InputStream`.

```
1       @Override
2   public Map<String, Object>
    getDataAudittrailStepWorkflowByCriteria(String search,
    String status, String startDate
3       , String endDate, List<String> branch, String
    branchAuth) throws AppException
4   {
5       // TODO Auto-generated method stub
6       Map<String, Object> retValue = new HashMap<String, Object
    >();
7       Map<String, Object> criteria = new HashMap<String, Object
    >();
```

```
8       List<AudittrailStepWorkflow> list = new ArrayList<
        AudittrailStepWorkflow>();
9       auditStepWorkflowRepository.findAll(
        builderAuditStepWorkflow.
        queryBuilderAudittrailStepWorkflow(criteria,
10              search,
11              status,
12              startDate,
13              endDate,
14              branch,
15              branchAuth)).forEach(audittrailId -> list.add(
        audittrailId));
16      long t = auditStepWorkflowRepository.count(
        builderAuditStepWorkflow.
        queryBuilderAudittrailStepWorkflow(criteria,
17              search,
18              status,
19              startDate,
20              endDate,
21              branch,
22              branchAuth));
23      retValue.put("total", t);
24      retValue.put("data", list);
25      return retValue;
26  }
27
28  @Override
29  public InputStream
        loadListDataAudittrailStepWorkflowDownload(List<
        AudittrailStepWorkflow> listData) throws AppException {
30      ByteArrayInputStream in =
        ExcelBuilderAudittrailStepWorkflow.
        ParameterAudittrailStepWorkflowToExcel(listData);
31      return in;
32  }
33
```

Kode 3.8: Metode getDataAudittrailStepWorkflowByCriteria

Kemudian, diimplementasikan *class* baru bernama `ExcelBuilderAudittrailStepWorkflow`, implementasi *method* `queryBuilderAudittrailStepWorkflow` dalam *class* `QueryBuilderAudittrail`, dan membuat *class*

AuditTrailStepWorkflowRepository

- ExcelBuilderAuditTrailStepWorkflow:
  Class ini berfungsi sebagai utilitas untuk membangun file Excel dari data *workflow*. Disediakan *method* ParameterAuditTrailStepWorkflowToExcel untuk mengatur isi dan format file, serta hasExcelFormat untuk validasi. Hasil atau *screenshot* dari kode dapat dilihat pada Lampiran 9.

- queryBuilderAuditTrailStepWorkflow:
  Method queryBuilderAuditTrailStepWorkflow membangun Specification dinamis berdasarkan search, status, rentang transactionDate, serta otorisasi cabang (branchAuth atau listBranch). Pencarian dilakukan dengan cb.like pada beberapa *field* seperti branchid, rfqNo, workflowId, dan userid, yang digabungkan dengan cb.or. Filter tambahan diterapkan untuk status, cabang, dan tanggal menggunakan cb.equal atau cb.between. Jika branchAuth bernilai "000", maka pencarian dibatasi ke cabang dalam listBranch; jika tidak, hanya cabang otorisasi yang digunakan. Data juga di-*join* ke entitas workflow dan diurutkan berdasarkan transactionDate secara menurun. Hasil atau *screenshot* dari kode dapat dilihat pada Lampiran 10.

3.3.3 Audit Trail Contract

Pengembangan *API audit trail contract* pada modul *Procurement System UFINS* dimulai dengan menambahkan tiga *method* ke dalam AuditTrailController, yaitu viewAuditTrailContract, downloadAuditTrailContract, dan downloadAuditTrailContractExcel.

- viewAuditTrailContract:
  Pada Kode 3.9 terdapat metode viewAuditTrailContract. Metode tersebut menangani permintaan POST untuk melihat data dengan validasi token JWT. Data difilter berdasarkan search, status, startDate, endDate, dan branchId.

```
1     @SuppressWarnings("unchecked")
2     @RequestMapping(value = AudittrailConstant.
      PROCUREMENT_AUDITTRAIL_CONTRACT_VIEW, method =
      RequestMethod.POST)
```

```
3     @CrossOrigin(origins = "*")
4     @Logging
5     @Authenticate
6     public String viewAudittrailContract (HttpServletRequest
      request, HttpServletResponse response, @RequestBody View
      View) throws Exception
7     {
8       try
9       {
10        String tokens = request.getHeader(
      AuthenticationConstant.AUTHORIZATION);
11        String nezam = request.getHeader(
      AuthenticationConstant.NEZNAM);
12        authRes = AuthenticationApi.authentication(
      authentication, tokens);
13        if (authRes == null) {
14          return senderAuditManagement.responseNotAuthorized(
      request, response);
15        }
16        if (!nezam.equalsIgnoreCase(authRes.
      getAuthenticatedMethod())) {
17          return senderAuditManagement.responseNotAuthorized(
      request, response);
18        }
19        Map<String, Object> data = auditDAO.
      getDataAudittrailContractByCriteria(View.getSearch(), View
      .getStatus(), View.getStartDate(), View.getEndDate(), View
      .getBranchid(), authRes.getBranchId());
20        List<AudittrailContract> listAuditContract = (List<
      AudittrailContract>) data.get("data");
21        long total = (long)data.get("total");
22        if (total == 0) {
23          return senderAuditContract.responseEmpty(request,
      response);
24        }
25        return senderAuditContract.responseListCount(request,
       response, listAuditContract, total);
26      }
27      catch (Exception e) {
28        e.printStackTrace();
29        return senderAuditContract.sendErrorInternalServer(
      request, response, e, logger);
30      }
```

```
31      }
32
```

Kode 3.9: Metode viewAudittrailContract

- `downloadAudittrailContract`:
  Pada Kode 3.10 terdapat metode `downloadAudittrailContract`. *Method* ini menghasilkan file laporan PDF dari data *Audit Trail Contract* yang difilter berdasarkan parameter seperti `search`, `status`, `startDate`, `endDate`, dan `branchId`. Setelah proses *autentikasi* melalui `AuthenticationApi`, sistem mengambil `userId`, `branchId`, serta tanggal proses terkini. Data kemudian diambil dari *database* dan dikonversi menjadi daftar objek `AudittrailContract`. Parameter laporan disiapkan dan disisipkan ke dalam *template JasperReports* berformat `.jrxml`, yang kemudian dikompilasi dan diisi menggunakan `JasperFillManager`. Hasilnya dikonversi menjadi file PDF.

```
1     @SuppressWarnings("unchecked")
2     @RequestMapping(value = AudittrailConstant.
      PROCUREMENT_AUDITTRAIL_CONTRACT_VIEW, method =
      RequestMethod.POST)
3     @CrossOrigin(origins = "*")
4     @Logging
5     @Authenticate
6     public String viewAudittrailContract (HttpServletRequest
      request, HttpServletResponse response, @RequestBody View
      View) throws Exception
7     {
8       try
9       {
10        String tokens = request.getHeader(
      AuthenticationConstant.AUTHORIZATION);
11        String nezam = request.getHeader(
      AuthenticationConstant.NEZNAM);
12        authRes = AuthenticationApi.authentication(
      authentication, tokens);
13        if (authRes == null) {
14          return senderAuditManagement.responseNotAuthorized(
      request, response);
15        }
16        if (!nezam.equalsIgnoreCase(authRes.
      getAuthenticatedMethod())) {
```

```
17          return senderAuditManagement.responseNotAuthorized(
    request, response);
18        }
19        Map<String, Object> data = auditDAO.
    getDataAudittrailContractByCriteria(View.getSearch(), View
    .getStatus(), View.getStartDate(), View.getEndDate(), View
    .getBranchid(), authRes.getBranchId());
20        List<AudittrailContract> listAuditContract = (List<
    AudittrailContract>) data.get("data");
21        long total = (long)data.get("total");
22        if (total == 0) {
23          return senderAuditContract.responseEmpty(request,
    response);
24        }
25        return senderAuditContract.responseListCount(request,
     response, listAuditContract, total);
26      }
27      catch (Exception e) {
28        e.printStackTrace();
29        return senderAuditContract.sendErrorInternalServer(
    request, response, e, logger);
30      }
31    }
32
```

Kode 3.10: Metode downloadAudittrailContract

- `downloadAudittrailContractExcel`:
  Pada Kode 3.11 terdapat metode `downloadAudittrailContractExcel`. *Method* ini menangani pembuatan dan pengunduhan laporan *Audit Trail Contract* dalam format Excel, berdasarkan parameter seperti `search`, `status`, `startDate`, `endDate`, dan `branchId`. Setelah proses *autentikasi* melalui `AuthenticationApi`, data diambil dari *database* menggunakan method `getDataAudittrailContractByCriteria()`. Data tersebut diubah menjadi stream Excel melalui method `loadListDataAudittrailContractDownload()` dan dibungkus dalam `InputStreamResource`. Nama file dibentuk secara dinamis dengan *timestamp* dan dikembalikan ke *client* sebagai file Excel.

```
1   @SuppressWarnings({ "unused", "unchecked" })
2   @RequestMapping(value = AudittrailConstant.
    PROCUREMENT_AUDITTRAIL_CONTRACT_DOWNLOAD_EXCEL, method =
    RequestMethod.POST)
```

```
3     @CrossOrigin(origins = "*")
4     @Logging
5     @Authenticate
6     public ResponseEntity<InputStreamResource>
      downloadAudittrailContractExcel(HttpServletRequest request
      , HttpServletResponse response, @RequestBody View View)
      throws Exception
7     {
8        String tokens = request.getHeader(
      AuthenticationConstant.AUTHORIZATION);
9        authRes = AuthenticationApi.authentication(
      authentication, tokens);
10
11       Map<String, Object> data = auditDAO.
      getDataAudittrailContractByCriteria(View.getSearch(), View
      .getStatus(), View.getStartDate(), View.getEndDate(), View
      .getBranchid(), authRes.getBranchId());
12
13       List<AudittrailContract> listAuditContract = (List<
      AudittrailContract>) data.get("data");
14       String filename = "REPORT_AUDITTRAIL_CONTRACT_";
15       Date now = new Date();
16       String dateMerge = new SimpleDateFormat(
      ParameterConstant.DATETIME_YMDHMSSSS_MERGE).format(now);
17       String fileExportName = filename + dateMerge + ".xlsx";
18       logger.info("Start Print EXCEL Data Audittrail Contract
       " + new Date().getTime());
19       InputStreamResource file = new InputStreamResource(
      auditDAO.loadListDataAudittrailContractDownload(
      listAuditContract));
20       logger.info("End Print EXCEL Data Audittrail Contract "
       + new Date().getTime());
21       return ResponseEntity.ok().header(HttpHeaders.
      CONTENT_DISPOSITION, "attachment; filename=" +
      fileExportName).contentType(MediaType.parseMediaType("
      application/vnd.ms-excel")).body(file);
22    }
23
```

Kode 3.11: Metode downloadAudittrailContractExcel

Kemudian, dikembangkan *method* baru dalam class `AudittrailDAOImpl` untuk mengambil dan menghitung data berdasarkan kriteria pencarian

tertentu, yaitu `getDataAudittrailContractByCriteria` dan `loadListDataAudittrailContractDownload`.

- `getDataAudittrailContractByCriteria`
  Method ini digunakan untuk mengambil data berdasarkan beberapa kriteria seperti `search`, `status`, `startDate`, `endDate`, `branch`, dan `branchAuth`. Hasil dari *method* ini berupa `Map` yang berisi total data dan daftar hasil pencarian.

- `loadListDataAudittrailContractDownload`
  Method ini mengubah daftar objek `AudittrailContract` menjadi format Excel yang siap diunduh sebagai `InputStream`.

```
1     @Override
2   public Map<String, Object>
      getDataAudittrailContractByCriteria(String search, String
      status, String startDate
3         , String endDate, List<String> branch, String
      branchAuth) throws AppException
4   {
5     // TODO Auto-generated method stub
6     Map<String, Object> retValue = new HashMap<String, Object
      >();
7     Map<String, Object> criteria = new HashMap<String, Object
      >();
8     List<AudittrailContract> list = new ArrayList<
      AudittrailContract>();
9     auditContractRepository.findAll(builderAuditContract.
      queryBuilderAudittrailContract(criteria,
10          search,
11          status,
12          startDate,
13          endDate,
14          branch,
15          branchAuth)).forEach(audittrailId -> list.add(
      audittrailId));
16    long t = auditContractRepository.count(
      builderAuditContract.queryBuilderAudittrailContract(
      criteria,
17          search,
18          status,
19          startDate,
20          endDate,
```

```
21          branch,
22          branchAuth));
23      retValue.put("total", t);
24      retValue.put("data", list);
25      return retValue;
26    }
27
28    @Override
29    public InputStream loadListDataAudittrailContractDownload(
        List<AudittrailContract> listData) throws AppException {
30      // TODO Auto-generated method stub
31      ByteArrayInputStream in = ExcelBuilderAudittrailContract.
        ParameterAudittrailContractToExcel(listData);
32      return in;
33    }
34
```

Kode 3.12: Metode getDataAudittrailContractByCriteria

Kemudian, diimplementasikan *class* baru bernama `ExcelBuilderAudittrailContract`, implementasi *method* `queryBuilderAudittrailContract` dalam *class* `QueryBuilderAudittrail`, dan membuat *class* `AudittrailContractRepository`.

- `ExcelBuilderAudittrailContract`:
  Class ini berfungsi sebagai utilitas untuk membangun file Excel dari data kontrak. Disediakan *method* `ParameterAudittrailContractToExcel` untuk mengatur isi dan format file, serta `hasExcelFormat` untuk validasi. Hasil atau *screenshot* dari kode dapat dilihat pada Lampiran 11.

- `queryBuilderAudittrailContract`:
  Metode *queryBuilderAudittrailContract* membangun *Specification* dinamis berdasarkan parameter pencarian (`search`), `status`, `transactionDate`, dan `branchAuth`. Pencarian menggunakan `cb.like` pada beberapa *field* dan digabung dengan `cb.or`. Filter tambahan diterapkan berdasarkan `status`, `branchAuth`, serta rentang tanggal (`cb.equal` atau `cb.between`). Hasil diurutkan menurun berdasarkan `transactionDate`. Cuplikan kode terdapat pada Lampiran 12.

3.3.4 Audit Trail Parameter

Pengembangan *API audit trail parameter* dimulai dengan menambahkan tiga *method* ke dalam `AudittrailController`, yaitu `reportAuditParameterView`, `reportAuditParameterDownload`, dan `downloadAudittrailParameterExcel`.

- `reportAuditParameterView`:
  Menangani permintaan *POST* untuk melihat data dengan validasi token *JWT*. Data difilter berdasarkan `search`, `status`, `startDate`, `endDate`, dan `branchId`.

```
1    @SuppressWarnings("unchecked")
2    @RequestMapping(value = AudittrailConstant.
     PROCUREMENT_AUDITTRAIL_PARAMETER_VIEW, method =
     RequestMethod.POST)
3    @CrossOrigin(origins = "*")
4    @Logging
5    @Authenticate
6    public String reportAuditParameterView (
     HttpServletRequest request, HttpServletResponse response,
     @RequestBody View View) throws Exception {
7       try {
8          String tokens = request.getHeader(
     AuthenticationConstant.AUTHORIZATION);
9          String nezam = request.getHeader(
     AuthenticationConstant.NEZNAM);
10         uthRes = AuthenticationApi.authentication(
     authentication, tokens);
11         if (authRes == null) return senderAuditParameter.
     responseNotAuthorized(request, response);
12         if(!nezam.equalsIgnoreCase(authRes.
     getAuthenticatedMethod()))
13         return senderAuditParameter.responseNotAuthorized(
     request, response);
14
15         Map<String, Object> data = auditDAO.
     getDataAuditParameter(View.getSearch(), View.getStatus(),
     View.getStartDate(),View.getEndDate(),View.getBranchid(),
     authRes.getBranchId());
16
17         List<AudittrailParameter> listAuditParameter = (List<
     AudittrailParameter>) data.get("data");
```

```
18          long total = (long)data.get("total");
19          if (total == 0) return senderAuditParameter.
    responseEmpty(request, response);
20          return senderAuditParameter.responseListCount(request
    , response, listAuditParameter, total);
21        }
22        catch (Exception e) {
23          e.printStackTrace();
24          return senderAuditParameter.sendErrorInternalServer(
    request, response, e, logger);
25        }
26      }
27
```

Kode 3.13: Metode reportAuditParameterView

- `reportAuditParameterDownload`:
  Method ini menghasilkan file laporan PDF dari data *Audit Trail Parameter* yang difilter berdasarkan parameter seperti `search`, `status`, `startDate`, `endDate`, dan `branchId`. Setelah proses *autentikasi* melalui `AuthenticationApi`, sistem mengambil `userId`, `branchId`, serta tanggal proses terkini. Data kemudian diambil dari *database* dan dikonversi menjadi daftar objek `AudittrailParameter`. Parameter laporan disiapkan dan disisipkan ke dalam *template JasperReports* berformat .jrxml, yang kemudian dikompilasi dan diisi menggunakan `JasperFillManager`. Hasilnya dikonversi menjadi file PDF.

```
1     @SuppressWarnings({ "unchecked", "unused" })
2     @RequestMapping(AudittrailConstant.
    PROCUREMENT_AUDITTRAIL_PARAMETER_DOWNLOAD_PDF)
3     @CrossOrigin(origins = "*")
4     @Logging
5     @Authenticate
6     public void reportAuditParameterDownload(
    HttpServletRequest request, HttpServletResponse response,
    @RequestBody View View) throws Exception {
7
8       String tokens = request.getHeader(
    AuthenticationConstant.AUTHORIZATION);
9       authRes = AuthenticationApi.authentication(
    authentication, tokens);
10
11      SimpleDateFormat dateFormat3 = new SimpleDateFormat("
    yyyy-MM-dd HH:mm:ss");
12      String user = authRes.getUserId();
13      ProcessDate pros = reportdao.getProcessDate();
14      Date prosdate = SetProcessDate.getprosdate(pros);
15      String datetime = dateFormat3.format(prosdate);
16      List<JasperPrint> jasperPrints = new ArrayList<>();
17      String startDate = View.getStartDate();
18      String endDate = View.getEndDate();
```

```
19
20      Map<String, Object> data = auditDAO.
    getDataAuditParameter(View.getSearch(), View.getStatus(),
    View.getStartDate(), View.getEndDate(), View.getBranchid()
    , authRes.getBranchId());
21
22      List<AudittrailParameter> listAuditParameter = (List<
    AudittrailParameter>) data.get("data");
23
24      String fileName = templateLocation + "
    Audittrail_Parameter.jrxml";
25      Map<String, Object> parameters = new HashMap<String,
    Object>();
26      parameters.put("P_REPORT_LOCATION", templateLocation +
    "logo.png");
27      parameters.put("P_USER", user);
28      parameters.put("P_PROCESSDATE", datetime);
29      JasperReport jp = JasperCompileManager.compileReport(
    fileName);
30      JasperPrint print = JasperFillManager.fillReport(jp,
    parameters, new JRBeanCollectionDataSource(
    listAuditParameter));
31      jasperPrints.add(print);
32
33      PDF pdf = new PDF(response, "
    REPORT_AUDITTRAIL_PARAMETER_", ".pdf", print);
34    }
35
```

Kode 3.14: Metode reportAuditParameterDownload

- `downloadAudittrailParameterExcel`:
  Method ini menangani pembuatan dan pengunduhan laporan *Audit Trail Parameter* dalam format Excel, berdasarkan parameter seperti `search`, `status`, `startDate`, `endDate`, dan `branchId`. Setelah proses *autentikasi* melalui `AuthenticationApi`, data diambil dari *database* menggunakan *method* `getDataAudittrailParameterByCriteria()`. Data tersebut diubah menjadi *stream* Excel melalui *method* `loadListDataAudittrailParameterDownload()` dan dibungkus dalam `InputStreamResource`. Nama file dibentuk secara dinamis dengan *timestamp* dan dikembalikan ke *client* sebagai file Excel.

```
1     @SuppressWarnings({ "unchecked", "unused" })
2     @RequestMapping(AudittrailConstant.
    PROCUREMENT_AUDITTRAIL_PARAMETER_DOWNLOAD_PDF)
3     @SuppressWarnings("unchecked")
4     @RequestMapping(AudittrailConstant.
    PROCUREMENT_AUDITTRAIL_PARAMETER_DOWNLOAD_EXCEL)
5     @CrossOrigin(origins = "*")
6     @Authenticate
7     @Logging
8     public ResponseEntity<InputStreamResource>
    downloadAudittrailParameterExcel(HttpServletRequest
    request, HttpServletResponse response, @RequestBody View
    View) throws Exception {
9
10      String tokens = request.getHeader(
    AuthenticationConstant.AUTHORIZATION);
```

```
11        authRes = AuthenticationApi.authentication(
      authentication, tokens);
12        Map<String, Object> data = auditDAO.
      getDataAuditParameter(View.getSearch(), View.getStatus(),
      View.getStartDate(), View.getEndDate(), View.getBranchid()
      , authRes.getBranchId());
13        List<AudittrailParameter> listAuditParameter = (List<
      AudittrailParameter>) data.get("data");
14        String filename = "REPORT_AUDITTRAIL_PARAMETER_";
15        Date now = new Date();
16        String dateMerge = new SimpleDateFormat(
      ParameterConstant.DATETIME_YMDHMSSSS_MERGE).format(now);
17        String fileExportName = filename + dateMerge + ".xlsx";
18        logger.info("Start Print EXCEL Data Audittrail
      Parameter " + new Date().getTime());
19        InputStreamResource file = new InputStreamResource(
      auditDAO.loadListDataAudittrailParameterDownload(
      listAuditParameter));
20        logger.info("End Print EXCEL Data Audittrail Parameter"
       + new Date().getTime());
21        return ResponseEntity.ok().header(HttpHeaders.
      CONTENT_DISPOSITION, "attachment; filename=" +
      fileExportName).contentType(MediaType.parseMediaType("
      application/vnd.ms-excel")).body(file);
22     }
23
```

Kode 3.15: Metode downloadAudittrailParameterExcel

Kemudian, diimplementasikan *class* baru bernama `ExcelBuilderAudittrailParameter`, implementasi *method* `queryBuilderAudittrailParameter` dalam *class* `QueryBuilderAudittrail`, dan membuat *class* `AudittrailParameterRepository`.

- `ExcelBuilderAudittrailParameter`:
  Class ini berfungsi sebagai utilitas untuk membangun file Excel dari data parameter. Disediakan *method* `ParameterAudittrailParameterToExcel` untuk mengatur isi dan format file, serta `hasExcelFormat` untuk validasi. Hasil atau *screenshot* dari kode dapat dilihat pada Lampiran 13.

- `queryBuilderAudittrailParameter`:
  Metode `queryBuilderAudittrailParameter` membangun *Specification* dinamis berdasarkan parameter pencarian (`search`), `status`, `transactionDate`, dan `branchAuth`. Pencarian menggunakan `cb.like` pada beberapa field dan digabung dengan `cb.or`. Filter tambahan diterapkan berdasarkan `status`, `branchAuth`, serta rentang tanggal (`cb.equal` atau `cb.between`). Hasil diurutkan menurun berdasarkan `transactionDate`. Cuplikan kode terdapat pada Lampiran 14.

3.3.5 Hasil

Setiap *API endpoint* akan melalui proses *debugging* dan uji fungsionalitas. Tiga *method* utama yang diuji pada setiap *endpoint* adalah *view*, *download PDF* dan *download Excel*. Berikut adalah contoh pengujian yang dilakukan menggunakan *Postman*.

A API Audit Trail Step Workflow

API Audit Trail Step Workflow memiliki 3 *endpoint* utama yaitu:

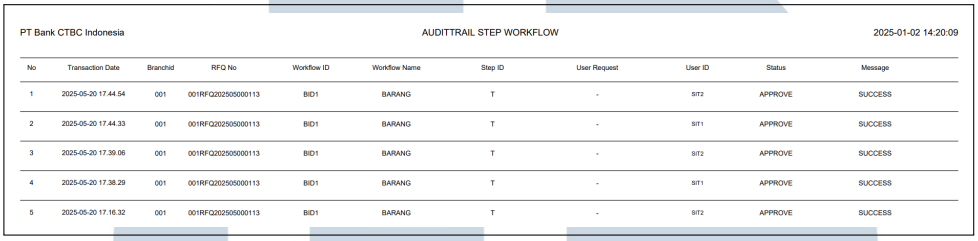
- procurement/audittrail/step/workflow/view
- procurement/audittrail/step/workflow/download/pdf
- procurement/audittrail/step/workflow/download/excel

Method `view` digunakan untuk mengambil dan menampilkan data berdasarkan kriteria pencarian. *Hit API* menggunakan metode **POST** ke alamat **http://localhost:3023/procurement/audittrail/step/workflow/view**, yang akan memanggil *method* `viewAudittrailSW` dan mengembalikan data dalam format **JSON**, seperti contoh berikut:

```
{
    "auditId": 127,
    "transactionDate": 1747737894959,
    "branchid": "001",
    "date": 1747674000000,
    "rfqNo": "001RFQ202505000113",
    "workflowId": "BID1",
    "stepId": "T",
    "userid": "SIT2",
    "status": "APPROVE",
    "message": "SUCCESS",
    "workflowName": "BARANG"
}
```

Method download PDF digunakan untuk menghasilkan dan mengunduh laporan audit trail dalam format PDF, dengan *template* yang dibuat menggunakan *Jaspersoft Studio*. API dipanggil melalui metode **POST** ke alamat **http://localhost:3023/procurement/audittrail/step/workflow/download/pdf**,

yang memanggil *method* `downloadAudittrailSW` dan mengembalikan file PDF. Jika diuji melalui Postman, file ditampilkan dalam bentuk mentah (*raw*) dan harus diunduh terlebih dahulu agar dapat dibuka dengan PDF *reader*. Output dapat dilihat pada Gambar 3.1.

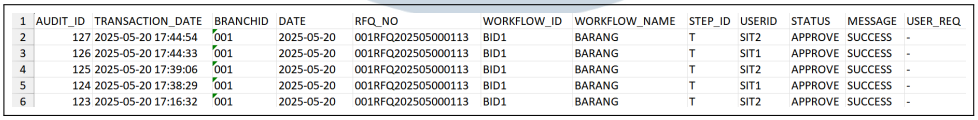


PT Bank CTBC Indonesia — AUDITTRAIL STEP WORKFLOW — 2025-01-02 14:20:09

No	Transaction Date	Branchid	RFQ No	Workflow ID	Workflow Name	Step ID	User Request	User ID	Status	Message
1	2025-05-20 17.44.54	001	001RFQ202505000113	BID1	BARANG	T	-	SIT2	APPROVE	SUCCESS
2	2025-05-20 17.44.33	001	001RFQ202505000113	BID1	BARANG	T	-	SIT1	APPROVE	SUCCESS
3	2025-05-20 17.39.06	001	001RFQ202505000113	BID1	BARANG	T	-	SIT2	APPROVE	SUCCESS
4	2025-05-20 17.38.29	001	001RFQ202505000113	BID1	BARANG	T	-	SIT1	APPROVE	SUCCESS
5	2025-05-20 17.16.32	001	001RFQ202505000113	BID1	BARANG	T	-	SIT2	APPROVE	SUCCESS

Gambar 3.2. Contoh Output Hasil PDF API Step Workflow

Method download Excel dengan alamat **http://localhost:3023/procurement/audittrail/step/workflow/download/excel** akan memanggil *method* `downloadAudittrailSWExcel` dan mengembalikan file Excel. Saat diuji di Postman, file akan muncul dalam format mentah dan perlu diunduh terlebih dahulu untuk dapat dibuka, seperti pada Gambar 3.2 dibawah.



	AUDIT_ID	TRANSACTION_DATE	BRANCHID	DATE	RFQ_NO	WORKFLOW_ID	WORKFLOW_NAME	STEP_ID	USERID	STATUS	MESSAGE	USER_REQ
2	127	2025-05-20 17:44:54	001	2025-05-20	001RFQ202505000113	BID1	BARANG	T	SIT2	APPROVE	SUCCESS	-
3	126	2025-05-20 17:44:33	001	2025-05-20	001RFQ202505000113	BID1	BARANG	T	SIT1	APPROVE	SUCCESS	-
4	125	2025-05-20 17:39:06	001	2025-05-20	001RFQ202505000113	BID1	BARANG	T	SIT2	APPROVE	SUCCESS	-
5	124	2025-05-20 17:38:29	001	2025-05-20	001RFQ202505000113	BID1	BARANG	T	SIT1	APPROVE	SUCCESS	-
6	123	2025-05-20 17:16:32	001	2025-05-20	001RFQ202505000113	BID1	BARANG	T	SIT2	APPROVE	SUCCESS	-

Gambar 3.3. Contoh Output Hasil Excel API Step Workflow

B API Audit Trail Contract

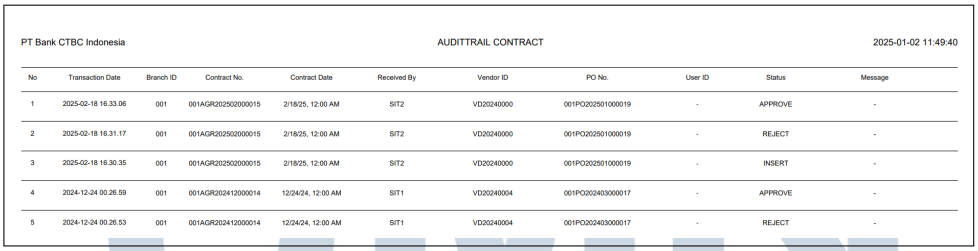
API Audit Trail Contract memiliki 3 *endpoint* utama yaitu:

- procurement/audittrail/contract/view
- procurement/audittrail/contract/download/pdf
- procurement/audittrail/contract/download/excel

Method `view` digunakan untuk mengambil dan menampilkan data berdasarkan kriteria pencarian. *Hit API* menggunakan metode **POST** ke alamat **http://localhost:3023/procurement/audittrail/contract/view**, yang akan memanggil *method* `viewAudittrailContract` dan mengembalikan data dalam format **JSON**, seperti contoh berikut:

```
{
    "auditId": 23,
    "transactionDate": 1739871186160,
    "branchid": "001",
    "contractNo": "001AGR202502000015",
    "contractDate": 1739811600000,
    "receivedBy": "SIT2",
    "vendorId": "VD20240000",
    "poNo": "001PO202501000019",
    "status": "APPROVE"
},
```


Method download PDF digunakan untuk menghasilkan dan mengunduh laporan audit trail dalam format PDF, dengan *template* yang dibuat menggunakan *Jaspersoft Studio*. API dipanggil melalui metode **POST** ke alamat **http://localhost:3023/procurement/audittrail/contract/download/pdf**, yang memanggil *method* `downloadAudittrailSW` dan mengembalikan file PDF. Jika diuji melalui Postman, file ditampilkan dalam bentuk mentah (*raw*) dan harus diunduh terlebih dahulu agar dapat dibuka dengan PDF reader. Output dapat dilihat pada Gambar



PT Bank CTBC Indonesia — AUDITTRAIL CONTRACT — 2025-01-02 11:49:40

No	Transaction Date	Branch ID	Contract No.	Contract Date	Received By	Vendor ID	PO No.	User ID	Status	Message
1	2025-02-18 16.33.06	001	001AGR20250200015	2/18/25, 12:00 AM	SIT2	VD20240000	001PO202501000019	-	APPROVE	-
2	2025-02-18 16.31.17	001	001AGR20250200015	2/18/25, 12:00 AM	SIT2	VD20240000	001PO202501000019	-	REJECT	-
3	2025-02-18 16.30.35	001	001AGR20250200015	2/18/25, 12:00 AM	SIT2	VD20240000	001PO202501000019	-	INSERT	-
4	2024-12-24 00.26.59	001	001AGR20241200014	12/24/24, 12:00 AM	SIT1	VD20240004	001PO202403000017	-	APPROVE	-
5	2024-12-24 00.26.53	001	001AGR20241200014	12/24/24, 12:00 AM	SIT1	VD20240004	001PO202403000017	-	REJECT	-

Gambar 3.4. Contoh Output Hasil PDF API Contract

Method download Excel dengan alamat http://localhost:3023/procurement/audittrail/contract/download/excel akan memanggil *method* `downloadAudittrailContractExcel` dan mengembalikan file Excel. Saat diuji di Postman, file akan muncul dalam format mentah dan perlu diunduh terlebih dahulu untuk dapat dibuka seperti pada Gambar 3.4 di bawah.

C API Audit Trail Parameter

API Audit Trail Parameter memiliki 3 *endpoint* utama, yaitu:

	AUDIT_ID	TRANSACTION_DATE	BRANCHID	CONTRACT_NO	CONTRACT_DATE	RECEIVED_BY	VENDORID	PO_NO	USERID	STATUS	MESSAGE
2	23	2025-02-18 16:33:06	001	001AGR2025020(	2025-02-18	SIT2	VD20240000	001PO202501000019	-	APPROVE	-
3	22	2025-02-18 16:31:17	001	001AGR2025020(	2025-02-18	SIT2	VD20240000	001PO202501000019	-	REJECT	-
4	21	2025-02-18 16:30:35	001	001AGR2025020(	2025-02-18	SIT2	VD20240000	001PO202501000019	-	INSERT	-
5	20	2024-12-24 00:26:59	001	001AGR2024120(	2024-12-24	SIT1	VD20240004	001PO202403000017	-	APPROVE	-
6	19	2024-12-24 00:26:53	001	001AGR2024120(	2024-12-24	SIT1	VD20240004	001PO202403000017	-	REJECT	-
7	18	2024-12-24 00:26:19	001	001AGR2024120(	2024-12-24	SIT1	VD20240004	001PO202403000017	-	INSERT	-
8	17	2024-12-23 14:51:34	001	001AGR2024120(	2024-12-23	SIT1	VD20240009	001PO202403000016	-	APPROVE	-
9	16	2024-12-23 14:49:14	001	001AGR2024120(	2024-12-23	SIT1	VD20240009	001PO202403000016	-	REJECT	-
10	15	2024-12-23 14:47:38	001	001AGR2024120(	2024-12-23	SIT1	VD20240009	001PO202403000016	-	INSERT	-

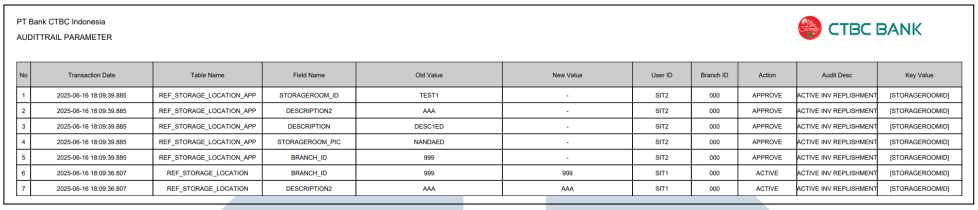
Gambar 3.5. Contoh Output Hasil Excel API Contract

- procurement/audittrail/parameter/view
- procurement/audittrail/parameter/download/pdf
- procurement/audittrail/parameter/download/excel

Method `view` digunakan untuk mengambil dan menampilkan data berdasarkan kriteria pencarian. *Hit API* menggunakan metode **POST** ke alamat **http://localhost:3023/procurement/audittrail/parameter/view**, yang akan memanggil *method* `reportAuditParameterView` dan mengembalikan data dalam format **JSON**, seperti contoh berikut:

```
{
    "audittrailId": 1566,
    "action": "APPROVE",
    "transactionDate": 1750072179885,
    "tableName": "REF_STORAGE_LOCATION_APP",
    "userId": "SIT2",
    "auditDesc": "ACTIVE INV REPLISHMENT GROUP",
    "fieldName": "STORAGEROOM_ID",
    "oldValue": "TEST1",
    "newValue": "-",
    "keyValue": "[STORAGEROOMID]",
    "branchid": "000"
},
```

Method ini menghasilkan dan mengunduh laporan *audit trail* dalam format PDF menggunakan *template* dari *Jaspersoft Studio*. API dipanggil dengan metode **POST** ke **http://localhost:3023/procurement/audittrail/parameter/download/pdf**, yang memicu *method* `reportAuditParameterDownload` dan mengembalikan file PDF. Saat diuji di Postman, file ditampilkan dalam format *raw* dan perlu diunduh terlebih dahulu untuk dibuka di PDF reader.



PT Bank CTBC Indonesia

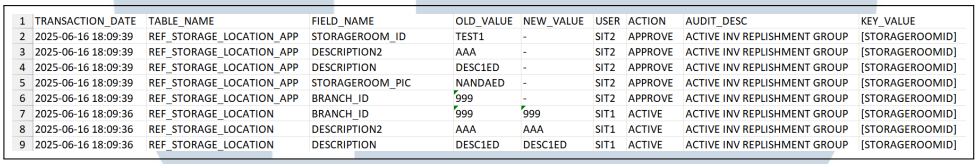
AUDITTRAIL PARAMETER

CTBC BANK

No	Transaction Date	Table Name	Field Name	Old Value	New Value	User ID	Branch ID	Action	Audit Desc	Key Value
1	2025-06-16 18:09:39.885	REF_STORAGE_LOCATION_APP	STORAGEROOM_ID	TEST1	-	SIT2	000	APPROVE	ACTIVE INV REPLISHMENT	[STORAGEROOMID]
2	2025-06-16 18:09:39.885	REF_STORAGE_LOCATION_APP	DESCRIPTION2	AAA	-	SIT2	000	APPROVE	ACTIVE INV REPLISHMENT	[STORAGEROOMID]
3	2025-06-16 18:09:39.885	REF_STORAGE_LOCATION_APP	DESCRIPTION	DESC1ED	-	SIT2	000	APPROVE	ACTIVE INV REPLISHMENT	[STORAGEROOMID]
4	2025-06-16 18:09:39.885	REF_STORAGE_LOCATION_APP	STORAGEROOM_PIC	NANDAED	-	SIT2	000	APPROVE	ACTIVE INV REPLISHMENT	[STORAGEROOMID]
5	2025-06-16 18:09:39.885	REF_STORAGE_LOCATION_APP	BRANCH_ID	999	-	SIT2	000	APPROVE	ACTIVE INV REPLISHMENT	[STORAGEROOMID]
6	2025-06-16 18:09:36.807	REF_STORAGE_LOCATION	BRANCH_ID	999	999	SIT1	000	ACTIVE	ACTIVE INV REPLISHMENT	[STORAGEROOMID]
7	2025-06-16 18:09:36.807	REF_STORAGE_LOCATION	DESCRIPTION2	AAA	AAA	SIT1	000	ACTIVE	ACTIVE INV REPLISHMENT	[STORAGEROOMID]

Gambar 3.6. Contoh Output Hasil PDF API Parameter

Method ini mengunduh laporan audit trail dalam format Excel melalui alamat **http://localhost:3023/procurement/audittrail/parameter/download/excel**, yang memanggil *method* `downloadAudittrailParameterExcel`.



	TRANSACTION_DATE	TABLE_NAME	FIELD_NAME	OLD_VALUE	NEW_VALUE	USER	ACTION	AUDIT_DESC	KEY_VALUE
2	2025-06-16 18:09:39	REF_STORAGE_LOCATION_APP	STORAGEROOM_ID	TEST1	-	SIT2	APPROVE	ACTIVE INV REPLISHMENT GROUP	[STORAGEROOMID]
3	2025-06-16 18:09:39	REF_STORAGE_LOCATION_APP	DESCRIPTION2	AAA	-	SIT2	APPROVE	ACTIVE INV REPLISHMENT GROUP	[STORAGEROOMID]
4	2025-06-16 18:09:39	REF_STORAGE_LOCATION_APP	DESCRIPTION	DESC1ED	-	SIT2	APPROVE	ACTIVE INV REPLISHMENT GROUP	[STORAGEROOMID]
5	2025-06-16 18:09:39	REF_STORAGE_LOCATION_APP	STORAGEROOM_PIC	NANDAED	-	SIT2	APPROVE	ACTIVE INV REPLISHMENT GROUP	[STORAGEROOMID]
6	2025-06-16 18:09:39	REF_STORAGE_LOCATION_APP	BRANCH_ID	999	-	SIT2	APPROVE	ACTIVE INV REPLISHMENT GROUP	[STORAGEROOMID]
7	2025-06-16 18:09:36	REF_STORAGE_LOCATION	BRANCH_ID	999	999	SIT1	ACTIVE	ACTIVE INV REPLISHMENT GROUP	[STORAGEROOMID]
8	2025-06-16 18:09:36	REF_STORAGE_LOCATION	DESCRIPTION2	AAA	AAA	SIT1	ACTIVE	ACTIVE INV REPLISHMENT GROUP	[STORAGEROOMID]
9	2025-06-16 18:09:36	REF_STORAGE_LOCATION	DESCRIPTION	DESC1ED	DESC1ED	SIT1	ACTIVE	ACTIVE INV REPLISHMENT GROUP	[STORAGEROOMID]

Gambar 3.7. Contoh Output Hasil Excel API Parameter

3.4 Kendala dan Solusi yang Ditemukan

- **Kendala:** Data tidak muncul pada tampilan frontend meskipun proses pengambilan data dari *database* berhasil dilakukan.

  Solusi: Data yang dikirim dari backend belum sesuai dengan struktur atau format yang diharapkan oleh frontend. Penyesuaian dilakukan pada response *controller* agar struktur data dan nama *field* sesuai dengan kebutuhan tampilan.

- **Kendala:** Nama variabel yang tidak sinkron atau tidak konsisten sehingga menyebabkan *server* mengembalikan kode error 500 (*Internal Server Error*) saat *request*. Kode error 500 merupakan indikasi bahwa telah terjadi kesalahan *serverside*, biasanya disebabkan oleh *exceptions*, kesalahan logika, atau masalah konfigurasi pada *backend*.

  Solusi: Dilakukan penelusuran kesalahan dengan membaca *stack trace* untuk mengidentifikasi bagian kode yang bermasalah. Selanjutnya, dilakukan penyesuaian agar nama-nama variabel yang digunakan di seluruh komponen menjadi konsisten. Proses ini diulangi sampai *server* memberikan respons dengan kode 200 (*OK*).

- **Kendala:** Ditemukan masalah kompatibilitas versi pada *Jaspersoft Studio* yang menyebabkan beberapa elemen laporan tidak dapat dimuat atau dijalankan dengan benar. Selain itu, diperlukan penyesuaian nama-nama *field* di *Jaspersoft* agar sesuai dengan nama variabel yang terdapat pada kelas *entity* di *backend*.

  Solusi: Masalah kompatibilitas diatasi dengan menjalankan *Jaspersoft Studio* dalam mode *compatibility*, yang memungkinkan akses dan penggunaan format atau fitur dari versi *Jaspersoft Studio* yang lama. Selain itu, nama-nama *field* dalam file `.jrxml` disesuaikan dengan atribut pada kelas *entity* agar data dapat ditampilkan dengan benar saat proses *rendering* laporan.