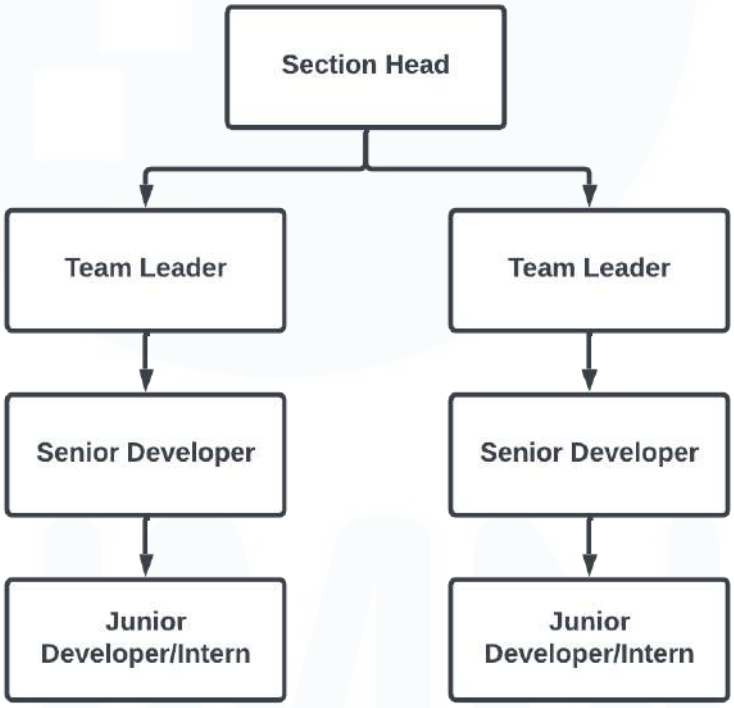
BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Kerja magang dilaksanakan pada BU DOCS yang terdiri atas personel. Kedudukan dan alur koordinasi pada unit bisnis ini dapat dilihat pada Gambar 3.1.



Gambar 3.1 Struktur Internal Tim BU DOCS

Setiap tim pada BU DOCS dipimpin oleh seorang *Section Head*, diikuti dengan *Team Leader*, dan *Supervisor*. Posisi *Intern* berada di bawah naungan seorang *Supervisor* atau bisa juga disebut sebagai *Senior* Developer yang bertanggung jawab dalam membimbing dan mengawasi pengerjaan tugas sehari-hari. Peran yang dipegang pada kerja magang ini adalah sebagai CR *Developer Intern*. Berikut merupakan rincian yang lebih detil terhadap tugas

atau tanggung jawab dari setiap personel dalam satu tim sesuai dengan pekerjaan yang dilakukan secara rutin sehari-hari:

1. *Section Head*

Sebagai *Section Head,* kedudukan ini memiliki peran untuk mengawasi dan mengelola satu bagian atau satu *section* dari departemen BU DOCS. Salah satu dari beberapa tanggung jawab seorang *Section Head* adalah menjalin dan memelihara hubungan yang efektif dan efisien dengan klien, serta dapat menyalurkan dan mengkomunikasikan informasi dari klien secara komprehensif kepada para *Section Head* agar dapat dilanjutkan kepada setiap tim dalam *section* tersebut. Dengan ini, dapat dipastikan bahwa tim memiliki kemampuan untuk menghasilkan solusi yang sesuai dengan tujuan dan visi yang diinginkan oleh klien.

Selain menjadi penghubung antar perusahaan dan klien, seorang *Section Head* juga memiliki tanggung jawab untuk mengawasi dan melakukan *review* akan hasil kerja dari *Section Head* dan *Senior Developer*. Hal ini dilakukan oleh *Section Head* untuk memastikan bahwa hasil kerja yang dilakukan oleh tim memenuhi standar kualitas dan ekspektasi dari klien. Tidak hanya itu, seorang *Section Head* dapat memberikan arahan, *feedback*, dan dukungan kepada para anggota tim agar pekerjaan dapat dilakukan secara efektif dan menghasilkan hasil yang memuaskan.

Terakhir, salah satu tanggung jawab utama dari *Section Head* adalah untuk melakukan manajemen sumber daya secara efisien. Hal ini termasuk penentuan biaya sebuah proyek atau tiket, yang dihitung dalam jumlah *mandays*. Dengan memahami kebutuhan proyek dan kemampuan sumber daya yang tersedia, seorang *Section Head* dapat mengatur jumlah *mandays* yang diperlukan dengan tepat. Selain itu, *Section Head* juga bertanggung jawab untuk memastikan bahwa setiap

anggota tim hadir dan bekerja sesuai dengan jadwal yang telah ditetapkan. Seluruh masalah izin perizinan juga harus di-*approve* oleh *Section Head* sebelumnya.

2. *Team Leader*

Team Leader pada sebuah tim, merupakan sebuah kedudukan yang memiliki peran yang krusial dalam mengelola tim saat melakukan pengembangan sebuah proyek atau tiket. Salah satu tanggung jawab utama dari seorang *Team Leader* merupakan melakukan alokasi tugas secara *detail* kepada *Senior Developer* dan *Intern*. Hal ini melibatkan pemahaman yang tepat akan keahlian dan kemampuan dari setiap anggota tim, seta memastikan tingkat kesulitan setiap tugas yang diberikan kepada para anggota tim merupakan dalam kemampuan anggota tersebut.

Selain melakukan alokasi tugas, seorang *Team Leader* juga bertanggung jawab untuk melakukan *review* terhadap hasil kerja teknis (*coding*) maupun non-teknis (UMT) yang dilakukan oleh *Senior Developer* dan *Intern* dalam satu tim. Hal ini mencakup memastikan bahwa *code* yang ditulis sesuai dengan standar yang telah ditetapkan oleh perusahaan, memastikan bahwa *code* yang ditulis memiliki performa yang efektif dan maksimal, serta memastikan bahwa *code* yang ditulis berhasil mewujudkan ekspektasi dari tugas yang telah diberikan. Tentunya, hal ini hanya dapat dilakukan oleh seorang *Team Leader* yang telah memiliki pemahaman yang mendalam akan standar *code* perusahaan sampai dengan dan pengalaman yang banyak mengenai aktivitas teknis yang dilakukan.

Tidak hanya itu, kedudukan sebagai *Team Leader* juga memiliki peran penting dalam membantu, mengawasi, dan membimbing anggota tim secara teknis jika terdapat suatu kendala atau tantangan dalam proses pengembangan. Hal ini bisa termasuk memberikan saran atau

solusi untuk membantu mengatasi masalah teknis, membantu memberikan dukungan kepada anggota tim untuk mengatasi sebuah hambatan, atau memberikan bimbingan tentang praktik *coding* yang standar pada perusahaan. Seorang *Section Head* juga memiliki tugas untuk menjelaskan sebuah *flow* yang ada dalam sebuah proyek atau tiket kepada anggota tim.

3. *Senior Developer (Supervisor)*

Seorang *Senior Developer* pada sebuah tim memiliki tanggung jawab untuk melakukan pengembangan proyek atau tiket, dan memastikan bahwa fungsionalitas dari hasil yang dihasilkan memenuhi kebutuhan klien. Hal ini tentunya meliputi penerapan solusi teknis yang efisien sesuai dengan kebutuhan proyek atau tiket, membuat *User Manual Testing* (UMT) atas setiap tugas yang dilakukan, dan melakukan *User Acceptance Testing* (UAT) untuk memastikan bahwa solusi yang dikembangkan memenuhi kriteria dan kebutuhan dari klien.

Tidak hanya itu, *Senior Developer* juga memiliki peran dalam membimbing *Junior Developer* atau *Intern* yang berada dalam naungannya. Hal ini bisa termasuk memberikan arahan dalam hal teknik maupun non-teknis, menjelaskan konsep-konsep yang kompleks, serta memberikan bantuan dan dukungan dalam menyelesaikan sebuah tugas. Tidak sebatas membimbing, *Senior Developer* juga harus melakukan *review* atas pekerjaan yang dilakukan oleh *Junior Developer* atau *Intern* agar dapat memberikan evaluasi terhadap *code* yang ditulis, memberikan *feedback* yang konstruktif, dan memastikan standar kualitas dari hasil pekerjaan yang dilakukan.

Selain tugas-tugas teknis tersebut, seorang *Senior Developer* juga dapat berhubungan langsung dengan klien untuk membantu jika terdapat masalah teknis yang ingin ditanyakan oleh klien. Hal ini membuat seorang *Senior Developer* juga wajib memiliki kemampuan

komunikasi yang baik serta pemahaman yang mendalam akan teknologi dan *framework* yang digunakan oleh perusahaan. Dengan ini, dapat dipastikan bahwa jika terdapat masalah teknis yang muncul, dapat diatasi secara cepat dan efisien, hingga hubungan dengan klien dapat berjalan lancar.

4. *Junior Developer / Intern*

Terakhir, seorang *Junior Developer* atau *Intern* memiliki tanggung jawab untuk melakukan pengembangan sebuah tugas dalam suatu proyek atau tiket. Tugas sehari-hari seorang yang memiliki kedudukan *Junior Developer* atau *Intern* adalah meliputi melakukan pengembangan fitur atau komponen baru dibawah pengawasan *Senior Developer* dan *Team Leader*, melakukan *debugging* untuk mengatasi segala *error* atau *bugs* yang dapat muncul selama masa *development*. Membuat dokument *User Manual Testing* (UMT), melakukan *User Acceptance Testing* (UAT) serta *System Integration Testing* (SIT) untuk memastikan bahwa solusi yang diberikan sudah sesuai dengan kebutuhan klien.

Selain tugas-tugas pada masa *development*, *Junior Developer* atau *Intern* juga memiliki tanggung jawab untuk memperbaiki *error* atau *bugs* saat fitur atau komponen sudah naik pada *environment production*. Selain itu, melakukan *versioning* juga menjadi salah satu tugas yang harus dilakukan, yang meliputi aktivitas pengelolaan *source code*.

Selain itu, koordinasi pada BU DOCS dilakukan secara "*top-down*", yaitu keputusan, arahan, dan informasi mengalir dari tingkat manajemen yang lebih tinggi ke tingkat yang lebih rendah pada hierarki. Dalam konteks ini, *Head of Business Unit* dan *Departement Head* akan menetapkan tujuan, strategi, dan kebijakan untuk disampaikan kepada *Section Head*, *Team Leader*, dan *Supervisor*. Operasional dan pelaksanaan tugas masing-masing divisi lalu

akan dikoordinasikan oleh tim secara internal dalam satu *section*. Namun, tidak menutup kemungkinan bahwa koordinasi juga dapat dilakukan secara horizontal antar unit atau departemen yang sejajar dalam struktur organisasi.

Meskipun demikian, para *Supervisor*, *Team Leader*, dan *Section Head* dapat melaporkan perkembangan, masalah yang dihadapi, dan saran lain mengenai proyek yang sedang dikerjakan kepada manajemen yang lebih tinggi. Hal ini dilakukan agar manajemen tingkat atas dapat memperoleh pemahaman yang lebih baik mengenai situasi di lapangan. Dengan demikian, pengambilan keputusan menjadi lebih terinformasi.

3.2 Tugas dan Uraian Kerja Magang

Selama periode magang yang dijalankan untuk melakukan pengembangan tiket bernomor ABC yang dilakukan untuk perusahaan *multifinance*, berikut merupakan ringkasan pekerjaan yang telah dilakukan selama periode magang berlangsung.

Tabel 3.1 Tugas dan Uraian Kerja

No.	Pekerjaan / Aktivitas	Minggu
1.	Mengikuti kegiatan *onboarding* dan *training* yang diadakan oleh AdIns untuk para *intern* yang telah diterima, sesuai dengan divisi dan *role* masing-masing.	Minggu 1-3
2.	Perkenalan kepada tim yang telah ditetapkan, lingkungan kerja dalam tim, klien tim, dan rutinitas sehari-hari tim	Minggu 3-4
3.	Tiket ABC (*new*): Melakukan modifikasi *database* sesuai dengan kebutuhan tiket, serta melakukan *update* koneksi dari database kepada ASP.NET	Minggu 4-5
4.	Tiket ABC (*new*): Melakukan pengembangan menu untuk menghasilkan sebuah laporan berbentuk Excel	Minggu 5-6
5.	Tiket ABC (*new*): Melakukan pengembangan menu setting untuk mengatur data (CRUD) kepada *database*	Minggu 6-8

No.	Pekerjaan / Aktivitas	Minggu
6.	Tiket ABC (*new*): Melakukan pengembangan menu untuk mencari dan meng-*update* data berbentuk jurnal kepada *database*	Minggu 8-9
7.	Tiket ABC (*new*): Melakukan koreksi dari hasil *feedback* dan *testing internal* yang diberikan oleh *Team Leader* dan *Supervisor*	Minggu 9-10
8.	Tiket ABC (*new*): Membuat dokumen UMT atas hasil kerja yang telah dilakukan	Minggu 10-11
9.	Tiket ABC (*new*): Mempersiapkan semua *file*, dokumen, dan kebutuhan lainnya agar tiket siap untuk masuk ke tahap *production*	Minggu 11-12

3.2.1 Mengikuti kegiatan *onboarding* dan *training* yang diadakan oleh AdIns untuk para *intern* yang telah diterima, sesuai dengan divisi dan *role* masing-masing.

Dalam rangka menerima dan menyambut para *Intern* yang sudah diterima untuk menjalankan aktivitas magang di AdIns, AdIns telah membuat acara *Pre On-Boarding* dimana peserta magang diminta untuk mengikuti proses sosialisasi yang dirancang khusus untuk memperkenalkan peserta magang terhadap lingkungan magang yang akan dijalankan. Aktivitas ini termasuk familiarisasi dengan fasilitas dan aturan yang berlaku di kantor AdIns. Setelah itu, peserta magang juga diminta untuk melakukan registrasi administratif, seperti pendaftaran *ID-Card* dan registrasi *fingerprint* untuk kepentingan absensi. Tahap ini bertujuan untuk memastikan bahwa para peserta magang memiliki akses dan pengenalan awal atas sistem dan prosedur yang akan diperlukan dan dijalankan selama masa magang berlangsung.

Selanjutnya, para peserta magang akan memasuki tahap resmi *Onboarding*, di mana peserta magang akan diperkenalkan secara resmi dengan perusahaan AdIns dan budaya kerjanya. Aktivitas *Onboarding*

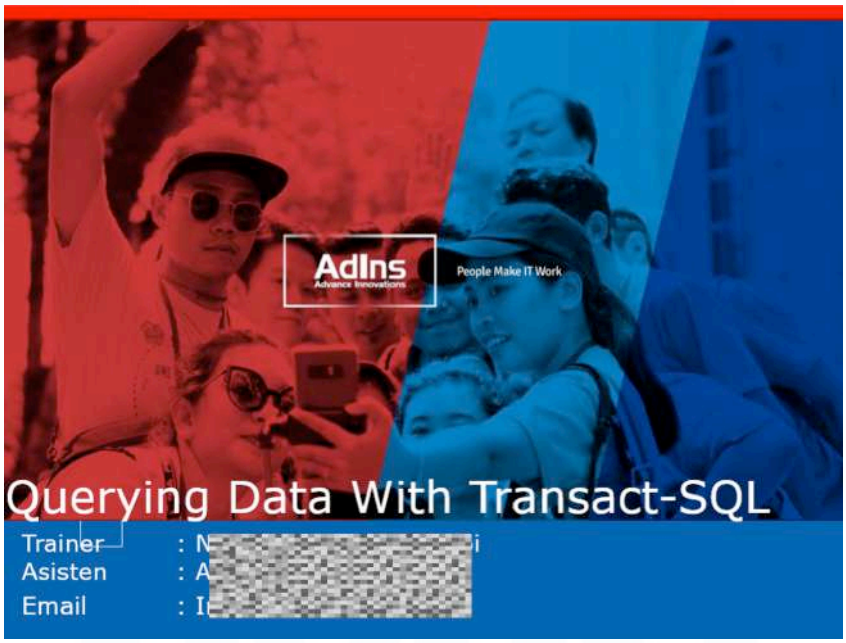
meliputi hal-hal seperti visi-misi perushahaan serta struktur organisasinya. Di mana selama tahap *Onboarding* peserta magang juga diperkenalkan secara langsung dengan para kepala-kepala perusahaan dan *business unit* yang ada. Tentunya, hal ini bertujuan untuk memberikan gambaran yang menyeluruh tentang bagaimana perusahaan beroperasi secara keseharian dan bagaimana setiap unit dapat saling berinteraksi.

Tahap terakhir dari aktivitas *Onboarding* adalah penerimaan laptop yang akan digunakan oleh peserta magang selama periode magang dan melakukan *setup* serta instalasi *framework*, aplikasi, dan teknologi yang diperlukan selama masa aktivitas magang berlangsung. Instalasi akan hal-hal ini dilakukan pada masa *Onboarding* untuk memastikan bahwa masa *training* dapat dilakukan dengan lancar dan efisien di mana tidak lagi ada kendala dengan aplikasi atau *framework* yang dibutuhkan. Dengan demikian, proses *Onboarding* ini membantu peserta magang untuk siap dan termotivasi untuk mengambil peran dalam keseharian proyek atau tiket yang akan dikerjakan selama periode magang.

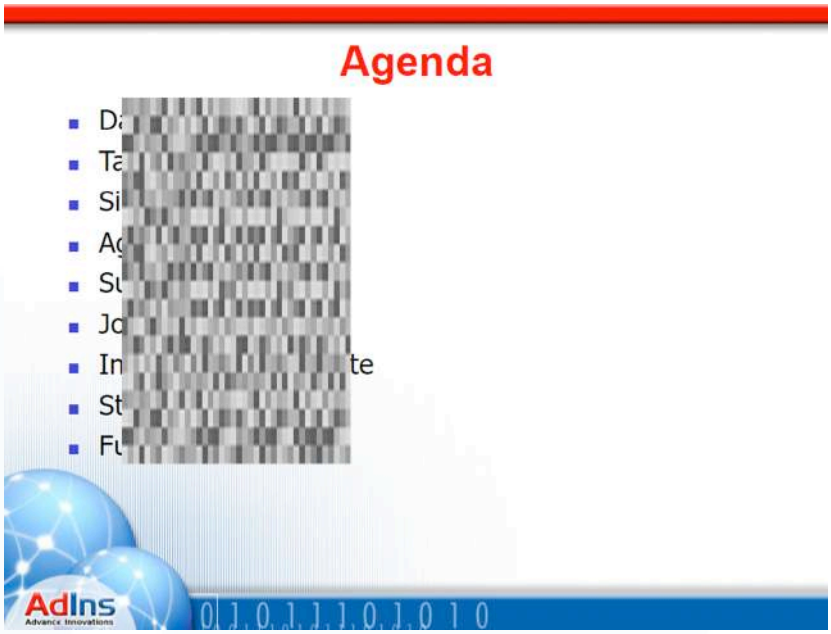
Setelah proses *Onboarding*, terdapat masa *Intensive Training* yang dijalankan oleh peserta magang selama 2 setengah minggu sebagai bagian dari persiapan sebelum terlibat dalam proyek atau tiket secara langsung. Selama periode ini, *training* yang diberikan dapat dibilang intensif yang mencakup berbagai kebutuhan atas pengertian teknis yang digunakan oleh AdIns. Tidak hanya itu, selama periode *Intesive Training*, dapat dijalankan sampai dengan 10-12 jam dalam satu hari untuk memastikan semua materi dan tugas dipahami dan dikerjakan secara komprehensif. Berikut merupakan seluruh materi *training* yang diberikan selama masa *Intensive Training*:

a. SQL Server (Hari ke-1)

Pada hari pertama *Intesive Training*, peserta magang mendapatkan pelatihan komprehensif mengenai SQL Server. Di mana pelatihan ini mencakup semua konsep dasar maupun kompleks dari bahasa SQL Server. Konsep yang dimaksud mencakup pembuatan *query* sederhana, konsep operasi CRUD (*Create, Read, Update, Delete)*, serta memahami pentingnya konsep *transaction*. Dapat dilihat dari kedua gambar dibawah, dimana modul yang digunakan oleh AdIns berada dalam format *Power Point* dan terdapat beberapa materi besar di dalamnya.



Gambar 3.2 Modul *Training* SQL Server



Gambar 3.3 Modul *Training* SQL Server #2

b. *Team Foundation Server*, ASP.NET, dan LINQ (Hari ke-2)

Selanjutnya, terdapat *training* mengenai *Team Foundation Server* atau TFS yang berguna untuk mengelola *Source Code* secara *central*. Lalu ada ASP.NET (C#) sebagai *framework* dasar yang digunakan untuk mengembangkan proyek atau tiket secara sehari-hari. Tidak hanya itu, peserta magang juga mempelajari konsep LINQ sebagai cara untuk mengeksekusi sebuah *query* dari ASP.NET kepada *database*. Gambar 3.4 dan 3.5 memperlihatkan tampilan modul yang digunakan untuk materi di hari kedua *Intensive Training*.

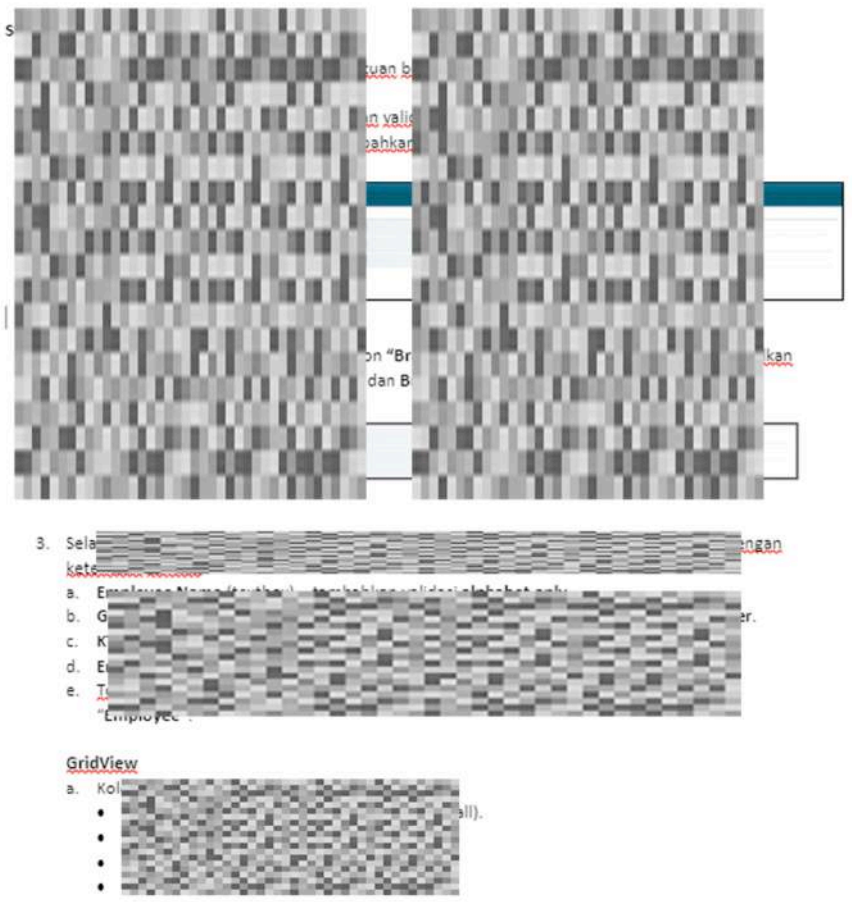
Gambar 3.4 Modul *Training* Hari ke-2

Gambar 3.5 Modul *Training* Hari ke-2 #2

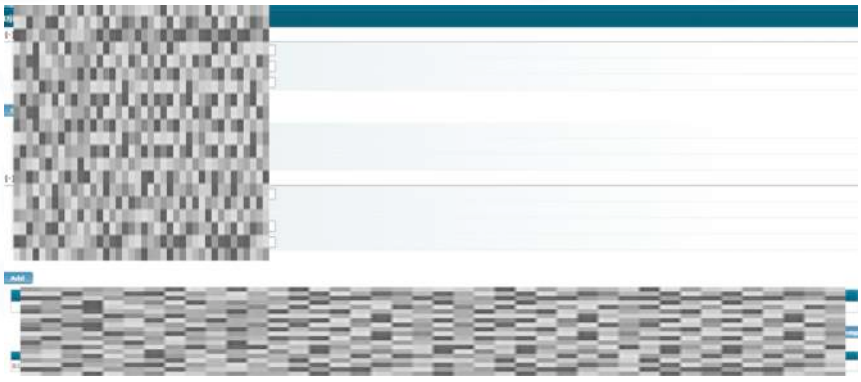
c. ASP.NET – *Part* 2 (Hari ke-3 & 4)

Pada hari ke-3, peserta magang diberikan materi mendalam mengenai konsep-konsep atau fitur yang ada pada *framework* ASP.NET. Konsep-konsep yang dimaksud meliputi fitur-fitur seperti *GridView, Temporary Grid Concept,* dan *Repeater* yang digunakan sehari-hari nantinya. Selain itu, terdapat juga pembuatan *WebForm* yang bertujuan untuk mengenalkan struktur dari pembuatan .aspx dan .cs (file web dari ASP.NET). Pada gambar 3.6 dan 3.7 terdapat contoh tampilan modul dari hari ketiga *Intensive Training* dengan soal ujian yang diberikan pada hari itu.

Pengerjaan ujian dilakukan oleh peserta magang pada hari yang sama dengan training yang dilakukan, dan juga di-*review* oleh penguji yang bertugas pada hari yang sama.



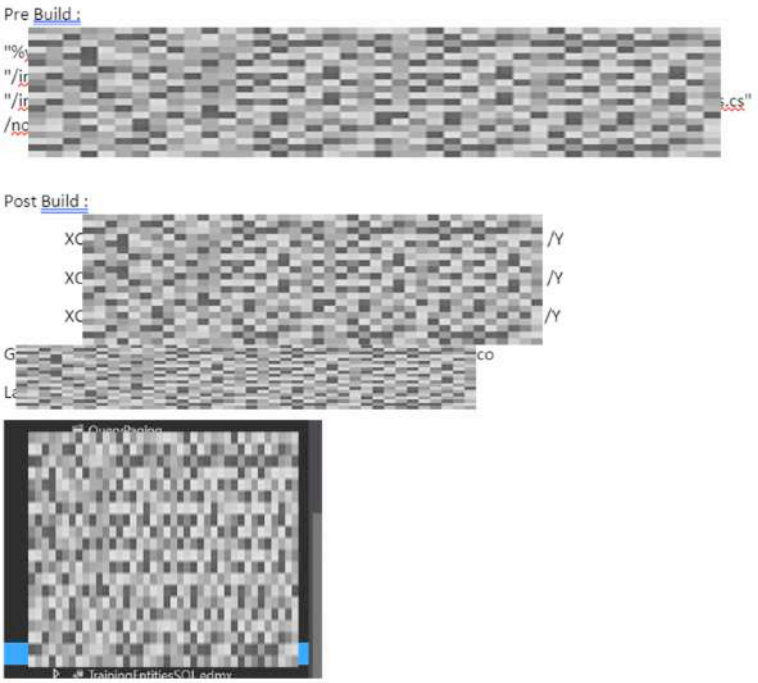
Gambar 3.6 Modul *Training* Hari ke-3



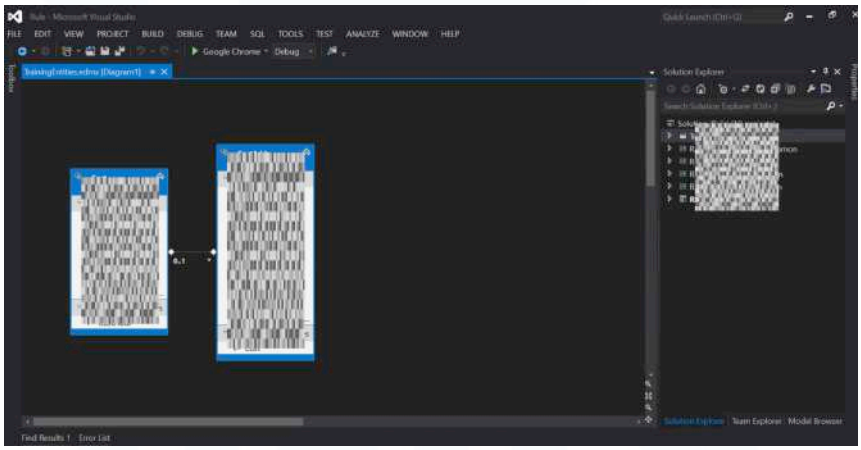
Gambar 3.7 Modul *Training* Hari ke-3 #2

d. *Plain Old CLR Objects* (POCO) (Hari ke-4 & 5)

Pada hari ke-4, dilanjutkan dengan materi *Plain Old CLR Objects* (POCO), yang berguna untuk membuat *object / entity* yang merepresentasikan tabel pada *database*. Selain itu, telah diberi juga pelatihan mengenai cara untuk membuat koneksi dari Microsoft Visual Studio kepada SQL Server untuk melakukan operasi CRUD. Proses POCO sendiri dapat dibilang lumayan kompleks, sehingga dibutuhkan ketelitian agar proses tidak harus diulang berkali-kali. Salah satu tahap paling penting dalam melakukan proses POCO adalah untuk melakukan tahap *Pre-Build* dan *Post-Build*. Pada tahap ini, seluruh *object / entity* yang telah menjalani proses POCO akan dituliskan dan di-*copy* kedalam 4 file berbeda yaitu .mdl, .csdl. .ssdl, dan .Views.cs. Keempat file ini dapat terbuat saat peserta magang melakukan input *code Pre-Build* dan *Post-Build* dimana *code* yang diinput akan dijalankan pada saat sebelum *build* terjadi dan sesudah *build* terjadi. Gambar 3.8 dan 3.9 menunjukkan modul yang digunakan pada hari itu, serta tampilan dari edmx yang dapat digunakan untuk melakukan proses POCO.



Gambar 3.8 Modul *Training* Hari ke-4



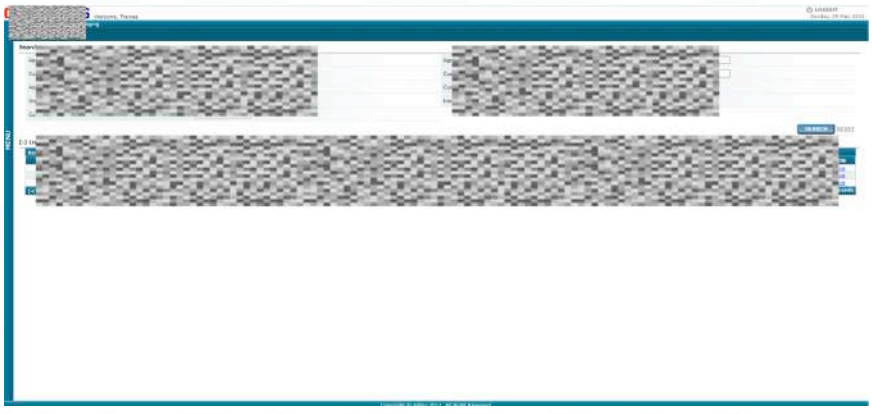
Gambar 3.9 Modul *Training* Hari ke-4 #2

e. *Framework* yang dibuat AdIns, *Review*, dan pengerjaan *Exam* (Hari ke-5 - 14)

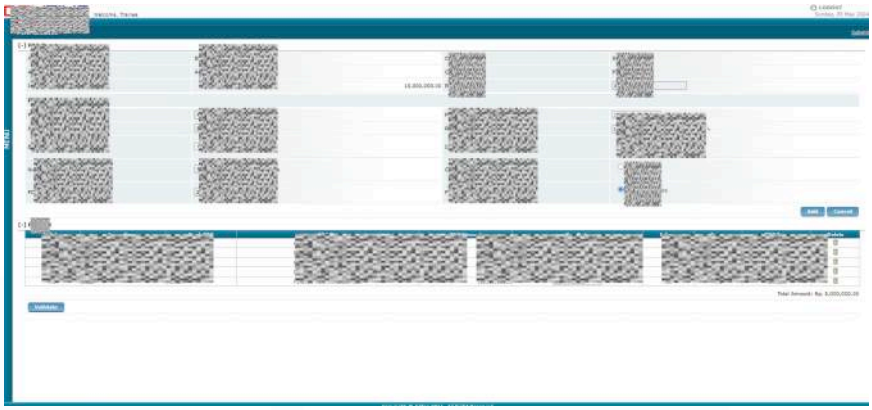
Selanjutnya, pada hari ke-5 sampai dengan hari ke-14, peserta magang diperkenalkan kepada *framework* yang telah dibuat oleh AdIns, yang dapat melakukan berbagai fungsionalitas seperti *searching, lookup, report,* dan masih banyak lagi. Materi yang

diberikan untuk topik ini dapat dibilang lumayan padat sehingga pelatihan dibagi menjadi 2 hari untuk setiap *framework* yang menjadi topik. Setelah semua materi *framework* telah selesai, maka akan dilakukan *review* yang bertujuan untuk mengingatkan ulang seluruh materi yang telah dipelajari dan membantu peserta magang jika terdapat topik yang masih belum sepenuhnya dipahami oleh peserta magang.

Setelah itu, salah satu aktivitas yang dilakukan untuk menutup masa *Intensive Training* adalah pembuatan *final exam* yang mencakup seluruh materi yang telah dipelajari dari hari pertama sampai dengan hari ke-14. Oleh karena soal yang diberikan untuk *final exam* tergolong berat, maka diberikan 4 hari untuk melakukan pengerjaan *final exam* sebelum lanjut ke tahap presentasi kepada penguji dan revisi atau *feedback* dari penguji. Dalam 4 hari tersebut, peserta magang mengerjakan berbagai fitur dan keperluan yang telah dicantumkan oleh soal, dengan menggunakan ilmu yang telah dipelajari dan didapatkan selama *training*. Berikut merupakan hasil akhir dari Web *Final Exam* yang dibuat:



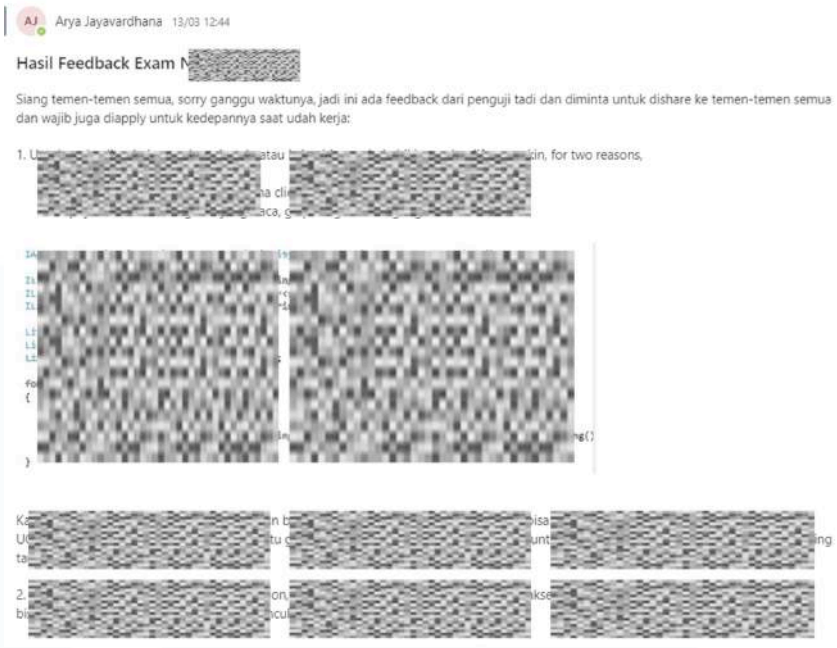
Gambar 3.10 Hasil Akhir *Web Final Exam*



Gambar 3.11 Hasil Akhir *Web Final Exam* #2

f. *Final Presentation* dan Revisi (Hari ke-14-17)

Proses terakhir dari masa *Intensive Training* adalah melakukan presentasi baik secara teknis (*coding*) dan non-teknis (fungsionalitas dan pemahaman materi) kepada penguji yang sudah ditetapkan. Hal ini dilakukan agar peserta magang dapat dipastikan sudah memahami seluruh konsep yang telah diajarkan secara mendalam, dan dapat melanjutkan proses untuk terjun ke pengembangan tiket atau proyek secara langsung. Tidak hanya itu, penguji juga memberikan revisi yang dapat dilakukan oleh peserta magang, atau *feedback* yang dapat membantu peserta magang untuk lebih mengoptimalisasikan *code* yang dibuatnya pada masa yang akan datang. Tidak hanya itu, peserta magang juga diminta untuk memberikan hasil *feedback* kepada peserta magang lainnya, agar informasi yang diberikan dapat diketahui oleh semuanya. Pada gambar 3.12, dapat dilihat bahwa peserta magang telah memberikan dan membagikan hasil *feedback* yang diberikan oleh penguji kepada grup internal *Intern R2* pada *platform* Teams.



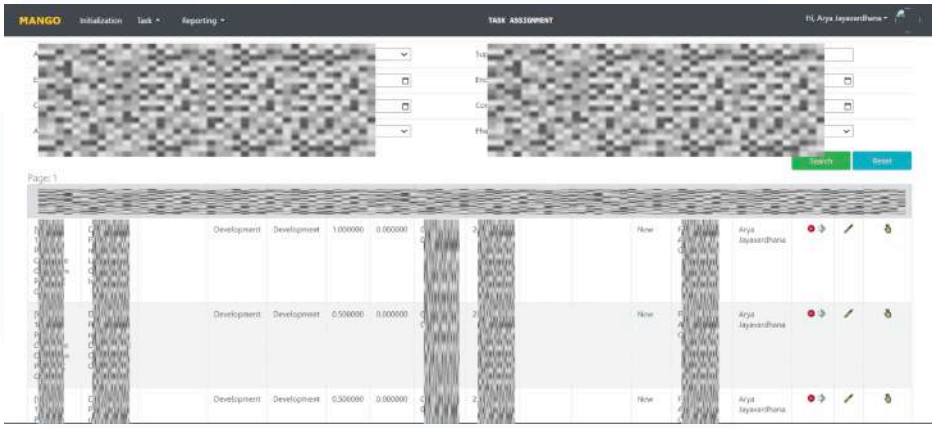
Gambar 3.12 Hasil *feedback Exam*

3.2.2 Perkenalan kepada tim yang telah ditetapkan, lingkungan kerja dalam tim, klien tim, dan rutinitas sehari-hari tim

Saat peserta magang sudah ditetapkan lulus dari masa *Intensive Training*, maka tahap selanjutnya ada pengenalan dan penempatan pada tim, perkenalan kepada klien tim, dan informasi mengenai rutinitas sehari-hari yang akan dilakukan sehari-hari seperti pelaporan tugas, alat komunikasi dengan tim, alat untuk absensi, dan sebagainya. Tentunya hal ini dilakukan agar peserta magang dapat merasa *familiar* dengan lingkungan dan rekan kerjanya sehari-hari nantinya. Tidak hanya itu, hal ini juga dilakukan agar peserta magang dapat mengerti dan memahami *flow* pekerjaan pada saat peserta magang telah mengerjakan proyek atau tiket secara langsung.

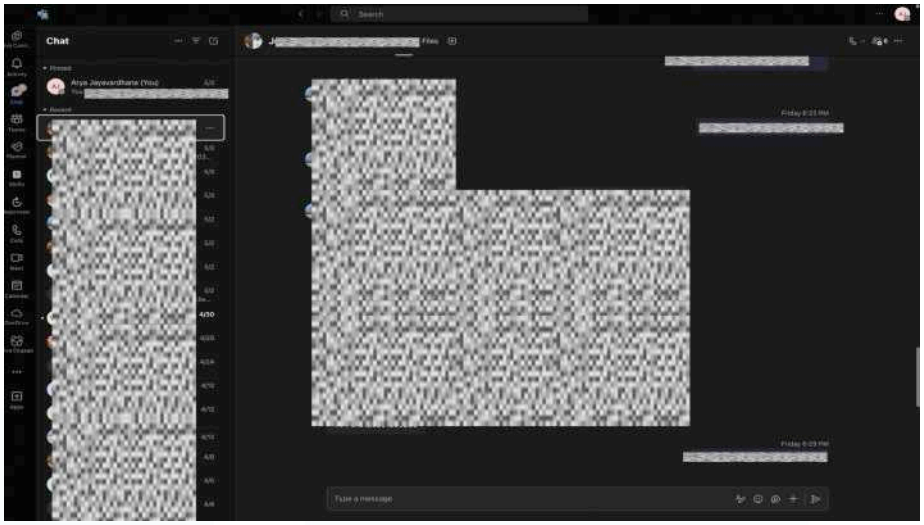
Dalam mengelola progres dari proyek yang sedang berjalan, AdIns menyediakan suatu *website* bernama Mango. Laman ini memfasilitasi para karyawan dan *intern* perusahaan untuk melaporkan aktivitas harian beserta dengan waktu yang dihabiskan untuk setiap

tugas. Mango memiliki *timesheet* yang dapat menghitung jumlah *mandays* secara otomatis. Dengan demikian, Mango juga dinilai memiliki peran yang krusial dalam memberikan informasi penting pada perusahaan sebab dapat digunakan untuk meninjau kinerja para karyawan.



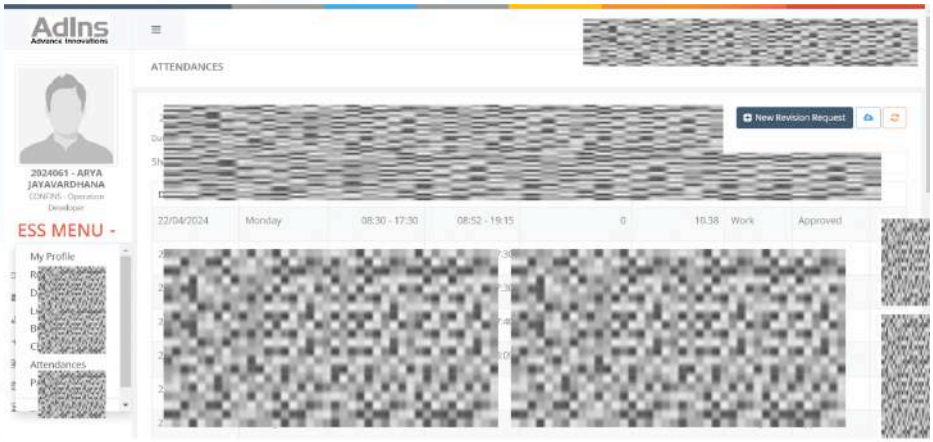
Gambar 3.13 Contoh tampilan pada *website* Mango

Sementara itu, seluruh koordinasi, perbincangan, dan diskusi secara grup dilakukan melalui Microsoft Teams. Terdapat berbagai grup internal sesuai dengan fungsi masing-masing, mulai dari grup besar *production* dan *developer* CR, grup besar *developer*, grup menurut *client*, grup peserta magang dengan *Supervisor* dan *Team Leader*, hingga grup peserta magang dengan *Supervisor* untuk diskusi yang lebih terfokus untuk melakukan supervisi. *Platform* ini juga dilakukan untuk pelaksaan *training* para *intern* pada grup tersendiri.



Gambar 3.14 Contoh komunikasi antar tim pada Microsoft Teams

Tidak hanya itu, AdIns juga menyediakan suatu *website* bernama X (*Dummy Name*), untuk memfasilitasi para karyawan dan *intern* untuk dapat memonitor absensi dan jumlah jam kerja dalam keseharian bekerja. Pada laman ini, para karyawan dan *intern* dapat melakukan revisi jam kerja, revisi *attendance*, lapor atas pelaksanaan WFO, WFH, atau *Out of Office Work*, melihat slip gaji, dan lain-lain. Akses kepada *website* ini diberikan setelah peserta magang lulus masa *Intensive Training* dan sudah masuk kedalam lingkungan pekerjaan.

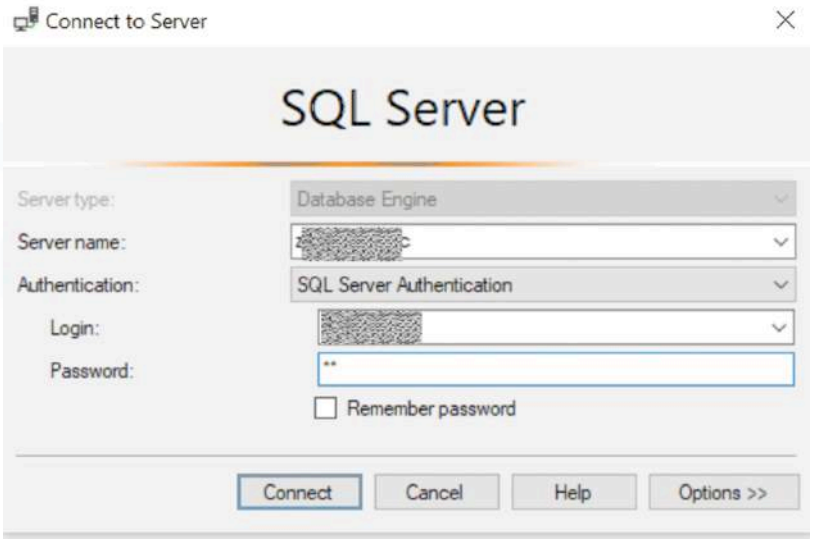


Gambar 3.15 Contoh tampilan pada *website* X (*Dummy Name*)

3.2.3 Tiket ABC (*new*): Melakukan modifikasi *database* sesuai dengan kebutuhan tiket, serta melakukan *update* koneksi dari database kepada ASP.NET

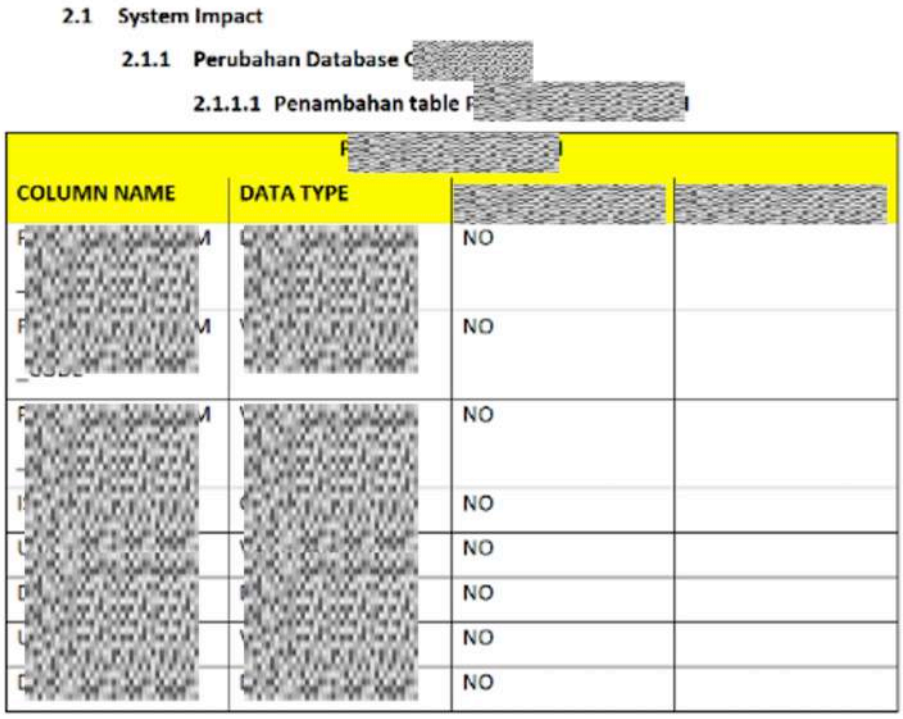
Setelah menjalani seluruh proses *Intensive Training* dan sudah dikenalkan kepada tim, klien, dan seluruh proses atau rangkaian baik dari sisi tim atau klien, maka peserta magang telah diberikan tugas untuk melakukan pengembangan untuk tiket bernomor ABC. Tugas yang pertama yang diberikan kepada peserta magang untuk tiket ini adalah untuk memodifikasi *database* dan melakukan penambahan tabel sesuai dengan kebutuhan yang telah tertera pada dokumen *Functional Specification Document* (FSD). Terdapat penambahan 4 tabel dan 8 kolom pada tabel *existing*.

Untuk melakukan hal ini, langkah pertama yang lakukan oleh peserta magang adalah melakukan proses koneksi ke *server database* bernama Z (*dummy name*) seperti yang dapat dilihat pada gambar 3.16. Proses koneksi akan dilakukan menggunakan *SQL Server Authentication* dengan menggunakan nama *server* tertentu dan memberikan *Login* dan *Password* yang sesuai. Tentunya, terdapat beberap *server* berbeda yang digunakan, dan *server-server* tersebut memiliki fungsi-funsi yang berebda.



Gambar 3.16 Proses *connect* ke *database*

Selanjutnya dibutuhkan untuk membuat halaman *query* baru agar nanti *script* SQL dapat disimpan. Terakhir, akan dibuat *query create table* dengan penjagaan untuk memastikan bahwa nama dari tabel yang akan dibuat tersebut belum terdapat pada *database* internal maupun *production*. Pembuatan tabel baru akan dilakukan sesuai dengan dokumen FSD yang telah diberikan oleh *supervisor*. Seperti yang dapat dilihat pada gambar 3.17, dokumen FSD menyediakan informasi nama tabel baru, kolom yang terdapat pada tabel tersebut, dan tipe data dari kolom tersebut. Dengan informasi ini, tugas dari peserta magang adalah membuat *query create table* yang dapat dilihat pada gambar 3.18.



2.1 System Impact

2.1.1 Perubahan Database

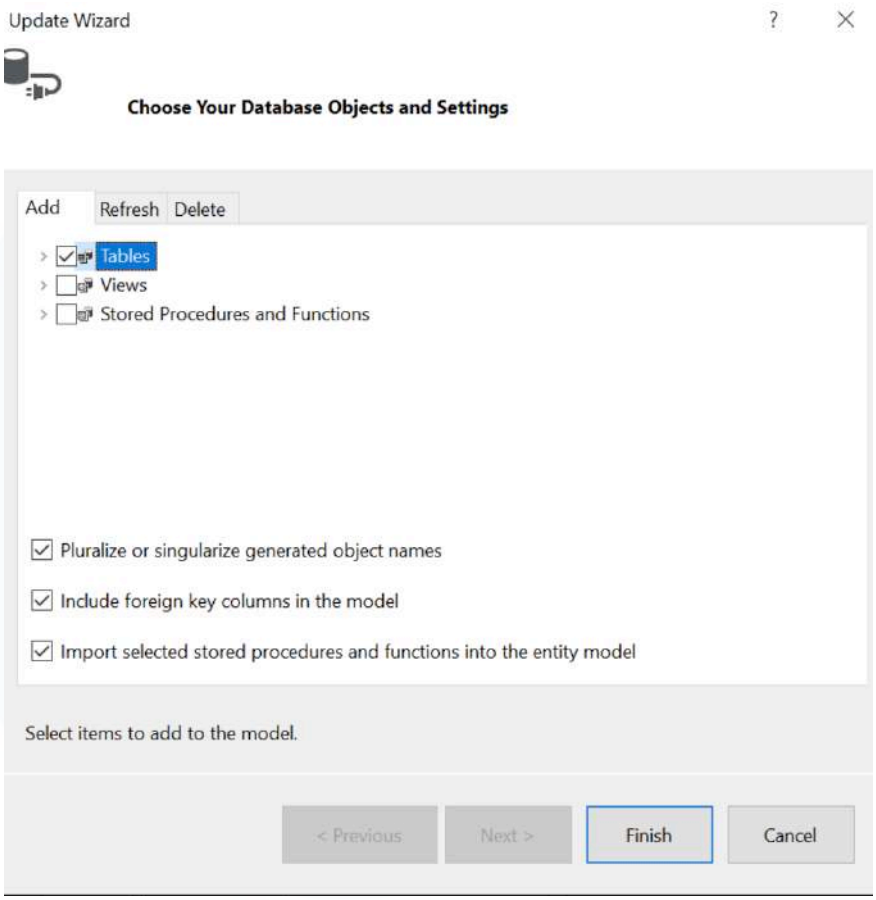
2.1.1.1 Penambahan table

COLUMN NAME	DATA TYPE		
		NO	
		NO	
		NO	
		NO	
		NO	
		NO	
		NO	
		NO	

Gambar 3.17 *Contoh* spesifikasi tabel pada FSD

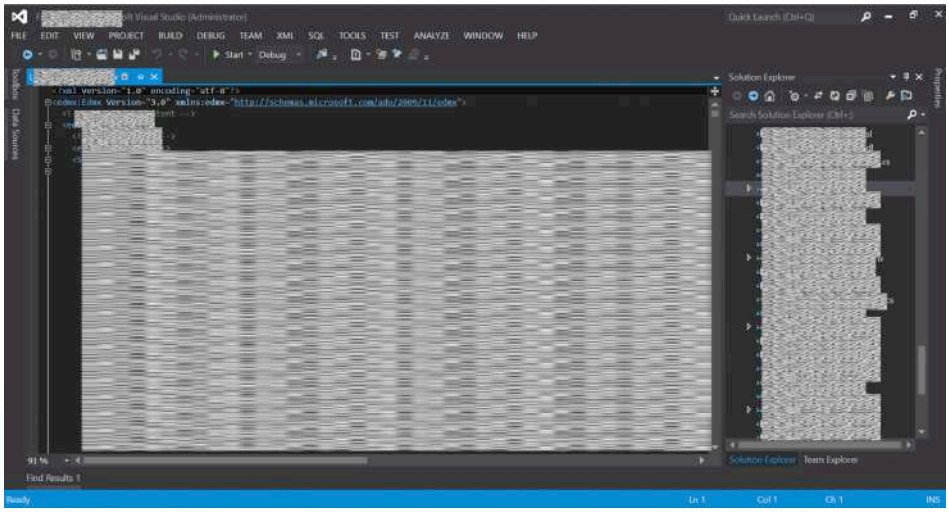
Gambar 3.18 *Script create table*

Setelah proses pembuatan tabel sudah selesai, peserta magang diberi tugas untuk melakukan *update* pada *source code* agar *object / entity* tabel yang baru ditambahkan kepada *database* dapat digunakan dalam *framework*. Untuk melakukan hal itu, Fitur *Update Wizard* dapat digunakan untuk hal ini, dengan cara menggunakan fitur *Add* atau *Refresh* yang berfungsi untuk menambahkan *object / entitiy* baru ke dalam *framework* atau memuat ulang *object / entity* yang sudah terdapat pada *framework*.



Gambar 3.19 Tampilan *Upadate Wizard* untuk menarik tabel dari *database*

Selain itu, peserta magang melakukan penarikan data dari *database* lalu melakukan konfigurasi ulang menggunakan XML Editor. Di mana XML Editor, berfungsi untuk mengubah *layout* .edmx yang berupa *graph table* menjadi *source code* yang dapat diubah secara manual oleh peserta magang, dan membuat proses modifikasi *database* dan koneksi antar *framework* dan *database* menjadi lebih mudah.



Gambar 3.20 Tampilan XML Editor

3.2.4 Tiket ABC (*new*): Melakukan pengembangan menu untuk menghasilkan sebuah laporan berbentuk Excel

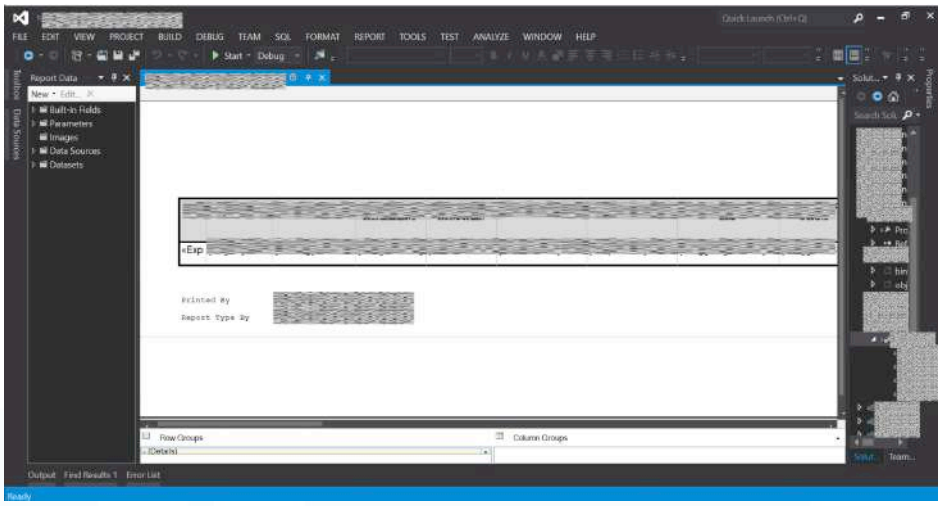
Setelah tahap modifikasi *database* telah selesai, maka selanjutnya tugas yang diberikan oleh supervisi kepada peserta magang adalah untuk melakukan pengembangan sebuah menu yang dapat menghasilkan sebuah laporan berbentuk Excel. Menu ini bertujuan untuk memberikan karyawan klien sebuah wadah untuk melihat data dari jangka waktu tertentu yang telah ditentukan oleh *user* (Maks. 1 bulan). Selain itu, terdapat 33 kolom yang akan ditampikan pada saat *user* melakukan *print* dari *report* pada menu tersebut. Berikut merupakan *pseudo code* untuk melakukan pengecekan jangka hari yang di-input oleh *user*.

```
DECLARE DateTime dtFrom = DateTime.Now
DECLARE DateTime dtTo = DateTime.Now

FOR EACH KeyValuePair<string, object> entry IN criteria DO
    IF entry.Key EQUALS "StartDtFrom" THEN
        SET dtFrom = Convert.ToDateTime(entry.Value.ToString())
    END IF

    IF entry.Key EQUALS "StartDtTo" THEN
        SET dtTo = Convert.ToDateTime(entry.Value.ToString())
    END IF
END FOR EACH

DECLARE int diffDate = (dtTo.Date - dtFrom.Date).Days

IF diffDate GREATER THAN 30 THEN
    THROW new Exception("Date range exceeds 30 days.")
END IF
```

Gambar 3.21 *Pseudo code* penjagaan *range* hari

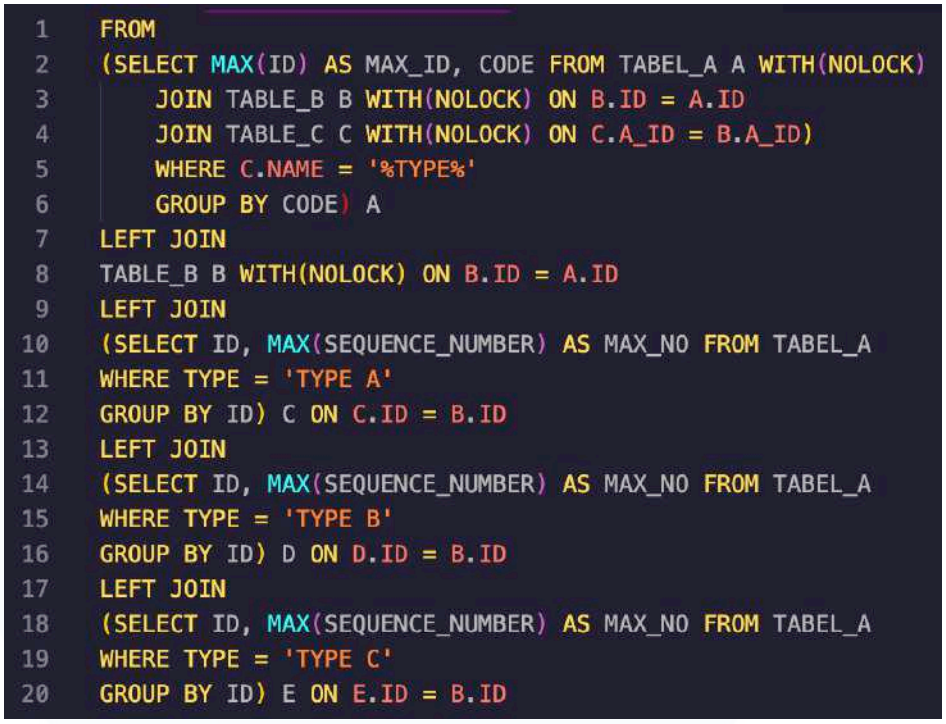
Dari gambar 3.21, dapat dilihat bahwa sistem akan mengambil filter yang di-input oleh *user* dan dari itu akan masukkan ke dalam sebuah variabel dtFrom dan dtTo. Seteah itu akan dilakukan pengurangan di mana hasilnya akan dalam bentuk hari. Kemudia akan terdapat *logic* IF ELSE, di mana jika hasil pengurangan lebih besar dari 30 maka akan dilemparkan sebuah *exception* dan sistem akan berhenti.

Untuk melakukan hal ini, pertama-tama akan dibuat terlebih dahulu *template* laporan yang akan diprint oleh *user.* Hal ini dapat dilakukan dengan membuat *file* .rdlc pada ASP.NET, yang di mana pada *file* tersebut peserta magang dapat menetapkan kolom yang akan diambil dari *database* dan juga tampilan Excel yang akan di-*generate* oleh sistem saat *user* melakukan *print*. Berikut merupakan tampilan dari file .rdlc yang telah dibuat:



Gambar 3.22 Tampilan RDLC pada ASP.NET

Tidak hanya itu, untuk menu ini, dibutuh juga sebuah *stored procedure* dari *database*, untuk dapat menyediakan data untuk *report* ini. *Stored Procedure* yang telah dibuat akan menjalankan *query* dengan *where condition* yang didapatkan dari *filter* pada menu, yang di mana hasil dari *query* tersebut akan dikembalikan sebagai tabel kepada RDLC sebelum ditampilkan pada tabel di laporan tersebut. *Query* yang digunakan untuk menghasilkan data yang diperlukan dapat dibilang kompleks, dikarenakan terdapat beberapa *sub-query* yang digunakan agar dapat menghasilkan data yang diperlukan. Sebagai contoh, dapat dilihat bahwa dalam gambar 3.23, peserta magang salah satu contoh penggunaan *sub-query* adalah untuk mendapatkan data paling terakhir dalam satu *sequence* untuk satu data.



```
FROM
(SELECT MAX(ID) AS MAX_ID, CODE FROM TABEL_A A WITH(NOLOCK)
    JOIN TABLE_B B WITH(NOLOCK) ON B.ID = A.ID
    JOIN TABLE_C C WITH(NOLOCK) ON C.A_ID = B.A_ID)
    WHERE C.NAME = '%TYPE%'
    GROUP BY CODE) A
LEFT JOIN
TABLE_B B WITH(NOLOCK) ON B.ID = A.ID
LEFT JOIN
(SELECT ID, MAX(SEQUENCE_NUMBER) AS MAX_NO FROM TABEL_A
WHERE TYPE = 'TYPE A'
GROUP BY ID) C ON C.ID = B.ID
LEFT JOIN
(SELECT ID, MAX(SEQUENCE_NUMBER) AS MAX_NO FROM TABEL_A
WHERE TYPE = 'TYPE B'
GROUP BY ID) D ON D.ID = B.ID
LEFT JOIN
(SELECT ID, MAX(SEQUENCE_NUMBER) AS MAX_NO FROM TABEL_A
WHERE TYPE = 'TYPE C'
GROUP BY ID) E ON E.ID = B.ID
```

Gambar 3.23 Contoh *query from* dari SP menu pembuatan laporan

Seperti yang dapat dilihat dari gambar 3.23, terdapat potongan kecil dari *query select from* yang dibuat oleh peserta magang. Dapat dilihat dari *query* tersebut, salah sastu data yang diambil dari *database* merupakan data yang telah difilter agar hanya mengambil data dengan tipe tertentu dan memiliki *sequence number* paling tinggi. Hal ini dapat dilakukan dengan cara menggunakan beberapa *sub-query* berbeda dimana setiap *sub-query* memiliki fungsi untuk mengambil satu tipe spesifik dengan *sequence number* tertinggi. Dapat dilihat, *sub-query* pertama dengan alias A memiliki fungsi untuk mengambil data dengan tipe "*TYPE*" saja, dimana "*TYPE*" merupakan *dummy name* dari tipe induk yang ada pada *database*. Selanjutnya akan dilakukan *LEFT JOIN* kepada *Table_B* yang bertujuan untuk mengambil seluruh data yang ada pada master *table*. Setelah itu, beberapa *sub-query* selanjutnya yang memiliki alias C, D, dan E berfungsi untuk mengambil sebuah tipe spesifik dengan *sequence number* tertinggi dari *Table_B*.

Selain itu, terdapat sebuah *logic* pada *query* tersebut yang berguna untuk mengolah data waktu yang memiliki format detik, dan harus diubah menjadi format hh Jam mm Menit ss Detik. Berikut merupakan *pseudo code* yang telah dibuat untuk merepresentasikan *logic source code* yang telah dibuat peserta magang.

```
IF (DATEDIFF(SECOND, [tanggal_selesai], [tanggal_kerja]) IS NULL
    OR DATEDIFF(SECOND, [tanggal_selesai], [tanggal_kerja]) = 0) THEN
    RETURN ''-''
ELSE
    DECLARE jam INT = DATEDIFF(SECOND, [tanggal_selesai], [tanggal_kerja]) / 3600
    DECLARE menit INT = (DATEDIFF(SECOND, [tanggal_selesai], [tanggal_kerja]) % 3600) / 60
    DECLARE detik INT = DATEDIFF(SECOND, [tanggal_selesai], [tanggal_kerja]) % 60

    DECLARE hasil VARCHAR(100) = ''

    IF jam > 0 THEN
        hasil = hasil + CONVERT(VARCHAR, jam) + ' Jam '
    END IF

    IF menit > 0 THEN
        hasil = hasil + CONVERT(VARCHAR, menit) + ' Menit '
    END IF

    hasil = hasil + CONVERT(VARCHAR, detik) + ' Detik'

    RETURN hasil
END IF
```

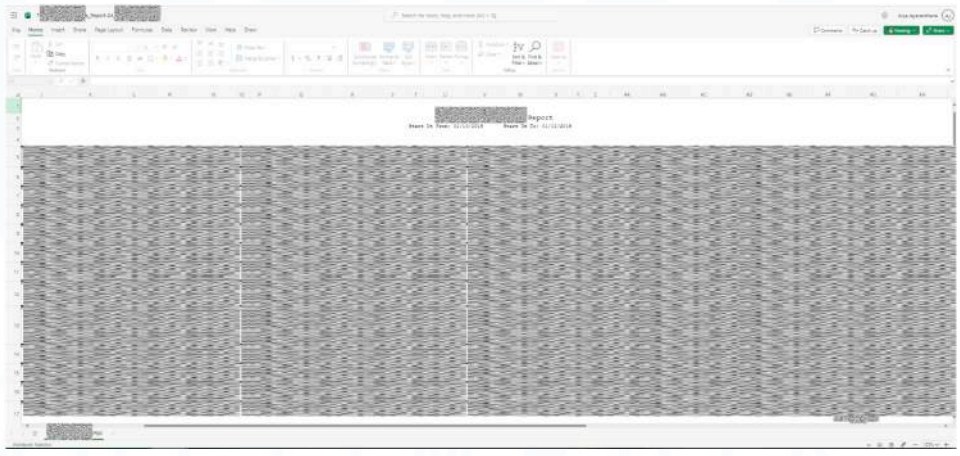
Gambar 3.24 *Pseudo code* untuk melakukan konversi detik

Pada gambar 3.24, dapat dilihat bahwa sistem akan mengecek terlebih dahulu apa kah hasil dari *function* DATEDIFF menghasilkan hasil *null* atau 0, jika iya maka sistem akan mengembalikan value ‘-‘, jika tidak maka hasil DATEDIFF akan dimasukkan kedalam variabel jam, menit, dan detik, di mana pada setiap variabel terdapat perhitungan untuk mengubah data detik menjadi jam dengan membagi data detik dengan 3600, karena 1 jam terdapat 3600 detik, menjadi menit dengan menggunakan fungsi *mod* atau *modulus* untuk mendapatkan sisa dari pembagian data detik dengan 3600, baru kemudian di bagi dengan 60, karena satu menit terdapat 60 detik, lalu terakhir, perhitungan detik akan dilakukan dengan menghidtung data detik *mod* 60 untuk mendapatkan sisa detik. Sebagai contoh, jika data detik 28,962 detik, maka dengan

logic ini akan diubah menjadi, 28,962 / 3600 yaitu 8, (28,962 % 3600) / 60 yaitu 2, lalu 28,962 % 60 yaitu 42, maka menjadi 8 Jam 2 Menit 42 Detik.

Langkah terakhir dari pengembangan ini adalah untuk membuat menu di mana para *user* dapat mengakses dan memberikan *filter* untuk di-*passing* kepada *Stored Procedure* yang telah dibuat. Berikut merupakan tampilan akhir dari menu tersebut dan hasil *print* dalam bentuk Excel yang dihasilkan.

Gambar 3.25 Tampilan *menu* yang *report* dibuat



Gambar 3.26 Hasil *print* menjadi *excel*

3.2.5 Tiket ABC (*new*): Melakukan pengembangan menu *setting* untuk mengatur data (CRUD) kepada *database*

Setelah pengembangan menu laporan selesai, maka peserta magang diberikan tugas selanjutnya yaitu membuat menu untuk melakukan *setting* data, yang dapat melakukan operasi CRUD (*Create, Read, Update, Delete*) kepada *database.* Menu ini merupakan salah satu fitur inti yang dikembangkan pada Tiket ABC. Pada menu ini, *user* dapat membuat data baru atau melakukan *edit* pada data baru yang berupa *Header* dan dari menu yang sama, *user* pun dapat memilih properti-properti yang dapat terkait dengan *header* tersebut.

Proses dalam menu ini dimulai dari pencarian data, jika *user* ingin melakukan *edit* pada data *existing*, lalu *user* dapat menekan tombol *edit*, dan akan dibawa ke halaman utama untuk menetapkan nama dan kode, sama jikanya kalau *user* ingin menambahkan data baru, namun perbedaannya adalah jika *edit* maka *textbox* akan secara otomatis terisi, jika menambahkan data baru maka *textbox* harus diisi secara manual. Pada halaman utama, terdapat penjagaan bahwa kode tidak boleh sama dengan yang di *database*, untuk melakukan hal ini, peserta magang harus berhati-hati karena validasi terdapat pada tombol *next* maka *logic* yang dibuat harus dapat mengetahui apakah *user* sedang dalam mode *add* atau *edit* dikarenakan jika *user* sedang dalam mode *edit* maka kode yang diinput boleh sama jika kode tersebut memang dari awal punya *header* tersebut. Berikut merupakan *pseudo code* yang digunakan untuk melakukan hal tersebut:

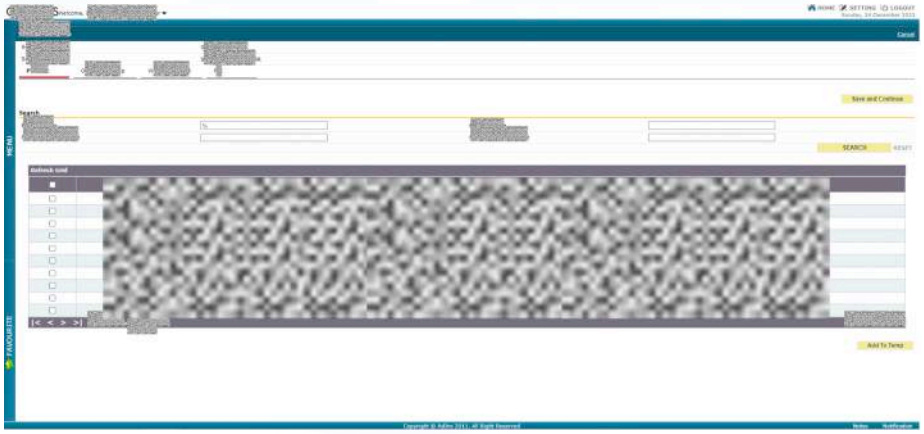
```
DECLARE bool check
DECLARE list existingHeader
DECLARE string inputCode
DECLARE string mode
DECLARE int id

IF mode equal "EDIT"
    FOR EACH Header h IN existingHeader
        IF h.Id != id AND h.Code == inputCode
            SET check = false
        END IF
    END FOR EACH
END IF
ELSE IF mode equal "ADD"
    FOR EACH Header h IN existingHeader
        IF h.Code == inputCode
            SET check = false
        END IF
    END FOR EACH
END ELSE IF
ELSE
    SET check = true
END ELSE
```

Gambar 3.27 *Pseudo code* penjagaan kode

Dapat dilihat dari *pseudo code* di atas, bahwa terdapat dua *mode* yang akan dicek oleh sistem, yaitu *Add* dan *Edit*. Jika *mode* sedang berada dalam *mode Edit* maka sistem akan melihat apakah sistem akan mengambil id *header* yang mau dilakukan *update* dan mengambil input kode yang baru, jika kode sudah terdapat untuk ide yang tidak sama dengan *header* yang sedang di-*edit* maka sistem akan mengembalikan *false*. Disisi lain, jika *mode* sedang berada pada *Add* maka sistem hanya akan mengecek apakah sudah ada kode yang sama pada *database*. Jika iya, maka akan mengembalikan *false* jika tidak maka sistem akan mengembalikan *true*.

Setelah itu, *user* akan dibawa ke halaman selanjutnya, di mana *user* dapat melakukan *searching* atas properti yang ingin dikaitkan

kepada *header* tersebut, di mana pada menu ini terdapat 4 tipe properti yang dapat dipilih, dan setiap properti memiliki *search criteria* nya masing-masing. Keempat properti tersebut memiliki *tab* masing-masing yang dapat diakses pada halaman ini, dan dari setiap *tab* tersebut terdapat tombol *Save and Continue* untuk membantu *user* melakukan *Save, Update,* ataupun *Delete* untuk tipe properti tersebut dan langsung di bawa ke halaman selanjutnya. Berikut merupakan hasil jadi dari halaman *setting* ini:

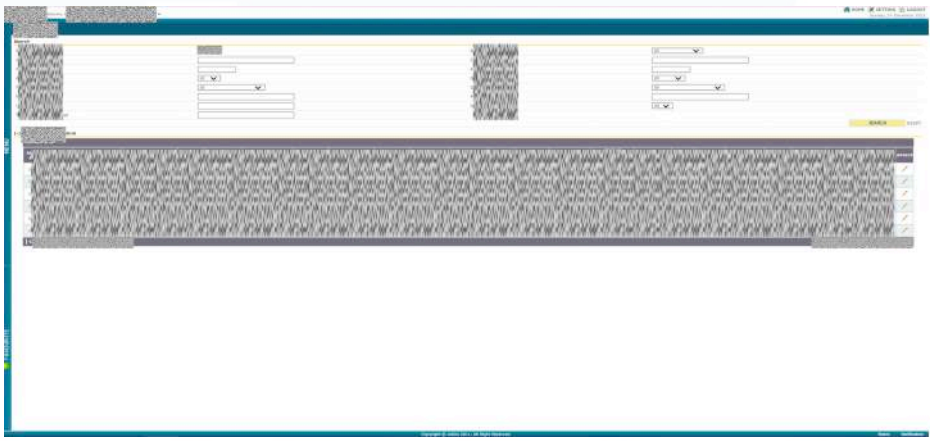


Gambar 3.28 Tampilan menu *setting*

3.2.6 Tiket ABC (*new*): Melakukan pengembangan menu untuk mencari dan meng-*update* data berbentuk jurnal kepada *database*

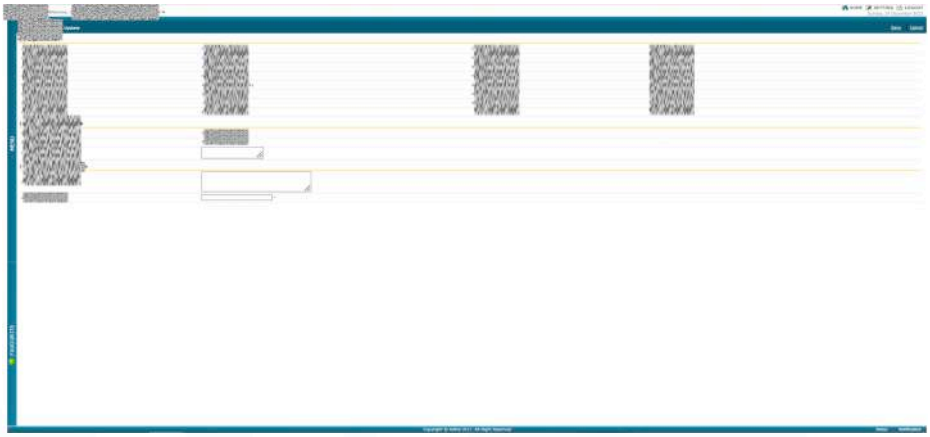
Selain menu *setting* tersebut, peserta magang juga diberi tugas untuk membuat menu yang dapat mencari dan meng-*update* data dalam bentuk jurnal (serupa dengan *Header* dan *Detail*) kepada *database*. Halaman ini akan membantu *user* untuk melakukan pencarian *header* dan *detail* dari jurnal tersebut. Selain itu, *user* dapat menyimpan *history* dari jurnal tersebut yang baru di-*input*. Terdapat 28 *column* yang akan diperlihat pada menu ini pada tampilan pencarian. Semua *column* tersebut akan diambil dari *database* menggunakan *Stored Procedure* atau SP yang akan mengembalikan satu tabel berisi semua data yang diperlukan. Berikut merupakan *outline query* yang digunakan untuk mengambil data tersebut:

Query tersebut selain mengembalikan data dan *column* yang diperlukan, juga mengembalikan total *row* atau jumlah data yang telah diambil dari berjalannya *query* tersebut. Data total *row* ini akan digunakan untuk men-*display* jumlah data yang ada kpada *user* di tampilan nantinya. *Query* tersebut juga telah dilengkapi dengan variabel @whereCond, yang berguna untuk menfilter data sesuai keinginan *user*. Variabel ini akan nantinya diisi dengan *syntax* WHERE .... , di mana kondisi *where* akan diambil dari filter yang di-*input* oleh *user* dari menu tampilan. Berikut merupakan tampilan akhir dari menu yang telah dibuat:



Gambar 3.29 Tampilan menu jurnal

Berikut merupakan tampilan dari menu untuk melakukan *upadate* kepada jurnal tersebut. Terdapat 2 mode pada menu ini, A dan B (*dummy name*), mode ini akan ditentukan dengan *user login* jika *user login* dari satu *privilage* tertentu akan masuk ke mode A dan jika tidak akan masuk ke mode B. Tidak hanya itu, *text* dan *update* yang dilakukan kepada *database* juga berbeda tergantung *mode* yang dilakukan. Berikut merupakan tampilan akhir dari menu untuk melakukan *update*:



Gambar 3.30 Tampilan menu *update* jurnal

3.2.7 Tiket ABC (*new*): Melakukan koreksi dari hasil *feedback* dan *testing internal* yang diberikan oleh *Team Leader* dan *Supervisor*

Dari hasil *development* yang telah dilakukan, sesuai dengan proses dan tugas dari setiap personel pada satu tim, *Team Leader* dan *Supervisor* telah melakukan koreksi dan memberikan hasil *feedback* dari semua hasil pengembangan yang telah dilakukan oleh peserta magang secara teknis. Seperti yang sudah dijelaskan sebelumnya, hal ini dilakukan untuk memastikan bahwa praktek teknis yang dilakukan telah memenuhi standar yang telah ditetapkan oleh perusahaan dan juga performa dari *code* yang telah ditulis oleh peserta magang sudang efisien dan efektif performanya. Tidak hanya itu, proses ini juga bertujuan untuk memastikan bahwa seluruh rangkaian pengembangan yang telah dilakukan oleh peserta magang telah sesuai dengan kriteria dan kebutuhan yang telah dituliskan pada dokumen *Functional Specification Document* atau FSD. Terakhir, *Team Leader* dan *Supervisor* juga memastikan bahwa seluruh *query* yang dibuat telah mengambil data dari tabel dan kolom yang benar, dan juga waktu *runtime* yang diperlukan untuk menjalankan *query* tersebut tidak melebihi waktu optimal.

Maka dari itu, beberapa revisi yang diberikan oleh *Team Leader* dan *Supervisor* adalah sebagai berikut:

1. Penambahan kolom sesuai dengan permintaan dari *user* dalam *query* yang sudah dibuat.

2. Koreksi *query* dalam pengambilan data (FROM) agar mengambil data yang benar, baik dari cara pengambilannya dan sumber pengambilannya.

Diluar dari kesalahan peserta magang, terdapat juga revisi yang diberikan oleh *user* yang belum atau lupa diberikan pada saat awal pengerjaan FSD:

1. Penambahan dan perubahan tabel yang dibutuhkan dalam memodifikasi *database* untuk keperluan tiket ini.

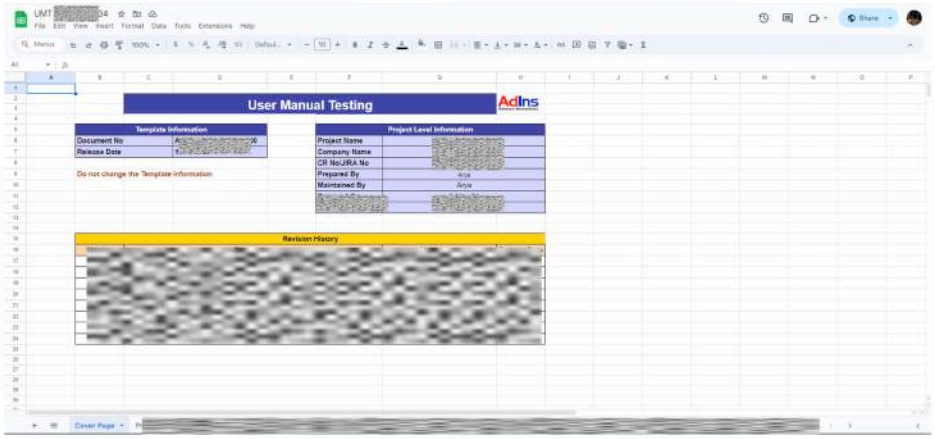
2. Penambahan kolom yang perlu ditampilkan pada menu, yang belum ditetapkan sebelumnya, dari 25 kolom menjadi 28 kolom.

3. Perubahan sumber data, sesuai dengan *request* dari klien yang telah mengklarifikasi ulang permintaan dan kebutuhan, sehingga terdapat modifikasi *query* dalam mengambil data.

3.2.8 Tiket ABC (*new*): Membuat dokumen UMT atas hasil kerja yang telah dilakukan

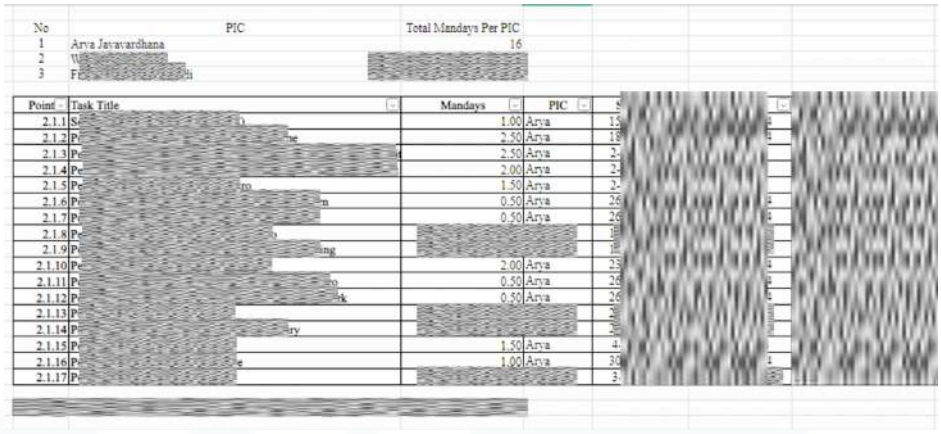
Setelah seluruh proses *development* dan revisi telah dibuat, maka terdapat proses pembuatan dokumen *User Manual* Testing atau UMT untuk membantu dalam 2 hal, yaitu melakukan manual *testing* untuk memastikan bahwa seluruh menu dan fitur yang telah dikembangkan telah berfungsi secara benar dan baik, dan untuk membuat suatu dokumen berbentuk dokumentasi dan sekaligus tutorial untuk klien agar para *user* dapat mengetahui cara untuk menggunakan menu yang telah dibuat.

Dokumen UMT akan dibuat menggunakan Excel, dan setiap menu yang telah dikembangkan oleh peserta magang akan mendapatkan *page*-nya masing-masing. Berikut merupakan contoh dari salah satu tampilan dokumen UMT yang telah dibuat:



Gambar 3.31 Tampilan *Cover Page* UMT

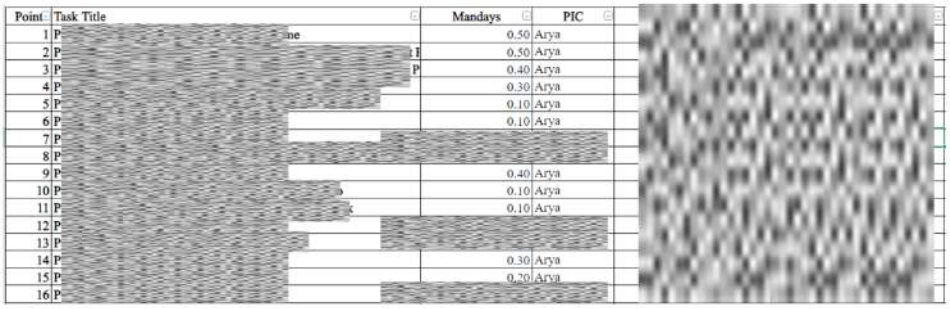
Selain pembuatan UMT, peserta magang juga wajib untuk melakukan *update* kepada *Excel* yang telah dibuat untuk melakukan *tracking* dan pembagian tugas oleh *Team Leader*. Status dari *development* wajib dijadikan *done* dan peserta magang harus menyertakan tanggal mulai dan tanggal selesai dari periode *development*. Selain itu, pada Excel tersebut, tertera juga jumlah *expected mandays* atau *effort* yang dikeluarkan oleh peserta magang selama masa *development*. Dari gambar 3.32 dan 3.33 di bawah, dapat dilihat bahwa peserta magang telah mengeluarkan atau mengerjakan tugas setara dengan 16 *mandays* untuk *development* dan 3 *mandays* untuk *Unit Testing* yang berarti menjadi total 19 *mandays* atau setara dengan 152 jam kerja. Tentunya, kalkulasi ini hanya estimasi, dan pembenaran *error*, *bugs*, dan kendala lainnya tidak termasuk dalam estimasi ini.



No	PIC	Total Mandays Per PIC
1	Arya Jayavardhana	16
2	W[redacted]	[redacted]
3	F[redacted]	

Point	Task Title	Mandays	PIC	5
2.1.1	S[redacted]	1.00	Arya	15
2.1.2	P[redacted]	2.50	Arya	18
2.1.3	P[redacted]	2.50	Arya	2
2.1.4	Pe[redacted]	2.00	Arya	2
2.1.5	P[redacted]	1.50	Arya	2
2.1.6	P[redacted]	0.50	Arya	25
2.1.7	P[redacted]	0.50	Arya	25
2.1.8	P[redacted]	[redacted]		
2.1.9	P[redacted]			
2.1.10	P[redacted]	2.00	Arya	23
2.1.11	P[redacted]	0.50	Arya	25
2.1.12	P[redacted]	0.50	Arya	26
2.1.13	P[redacted]	[redacted]		
2.1.14	P[redacted]	[redacted]		
2.1.15	P[redacted]	1.50	Arya	4
2.1.16	P[redacted]	1.00	Arya	30
2.1.17	P[redacted]	[redacted]		3

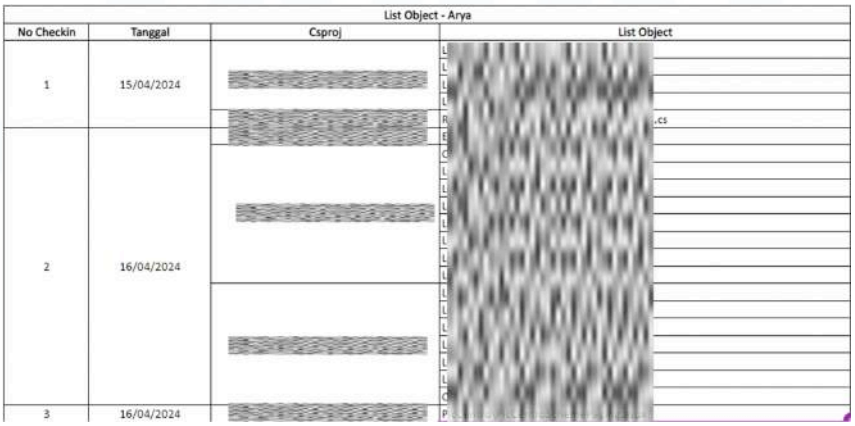
Gambar 3.32 Excel pembagian tugas *Development*



Point	Task Title	Mandays	PIC
1	P[redacted]	0.50	Arya
2	P[redacted]	0.50	Arya
3	P[redacted]	0.40	Arya
4	P[redacted]	0.30	Arya
5	P[redacted]	0.10	Arya
6	P[redacted]	0.10	Arya
7	P[redacted]		
8	P[redacted]		
9	P[redacted]	0.40	Arya
10	P[redacted]	0.10	Arya
11	P[redacted]	0.10	Arya
12	P[redacted]		
13	P[redacted]		
14	P[redacted]	0.30	Arya
15	P[redacted]	0.20	Arya
16	P[redacted]		

Gambar 3.33 Excel pembagian tugas *Unit Testing*

Terakhir, peserta magang juga diwajibkan untuk menuliskan setiap *object* yang telah dibuat dan akan digunakan atau menjadi *dependencies* dari tiket ini. Berikut merupakan contoh dari *list* yang telah dibuat:



List Object - Arya

No Checkin	Tanggal	Csproj	List Object
1	15/04/2024	[redacted]	[redacted] .cs
2	16/04/2024	[redacted]	
3	16/04/2024	[redacted]	

Gambar 3.34 Excel *List Object* dari pengembangan

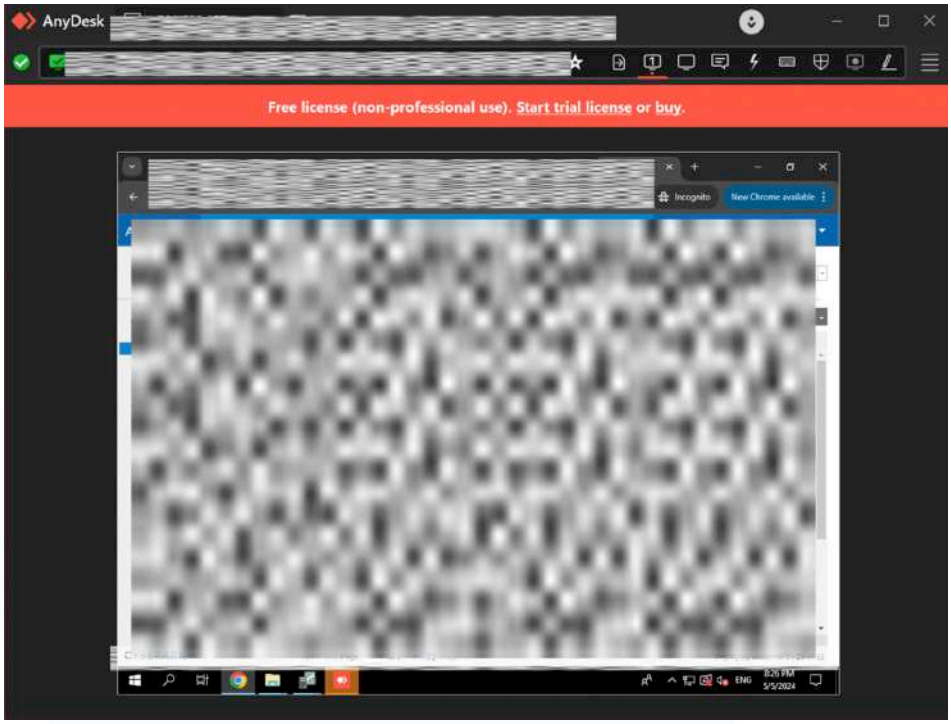
3.2.9 Tiket ABC (*new*): Mempersiapkan semua *file*, dokumen, dan kebutuhan lainnya agar tiket siap untuk masuk ke tahap *production*

Setelah seluruh dokumen, *file,* revisi, dan *testing* telah dilakukan, maka tiket siap untuk naik ke tahap *production*, di mana pada tahap ini akan dilakukan proses *System Integration Testing* atau SIT dan *User Acceptance Testing* atau UAT. Namun sebelum itu, peserta magang wajib merapihkan, menyiapkan, dan memastikan bahwa setiap *script*, *file* .aspx, dan dokumen lainnya sudah rapih dalam satu folder dan siap untuk dikirimkan kepada klien. Berikut merupakan contoh *file-file* yang telah disiapkan oleh peserta magang:

Gambar 3.35 Contoh *folder* yang akan dikirim ke klien

Selanjutnya, setelah semua *file* telah dikumpulkan dan dilengkapi, peserta magang akan melakukan pendaftaran VPN dan VM (*Virutal Machine*) untuk dapat melakukan koneksi kepada *server production*. Berikut merupakan Excel VM yang tersedia dan list VPN yang tersedia, dan juga tampilan dari VM:

Gambar 3.36 List VPN yang tersedia



Gambar 3.37 Tampilan VM *server production*

3.3 Kendala yang Ditemukan

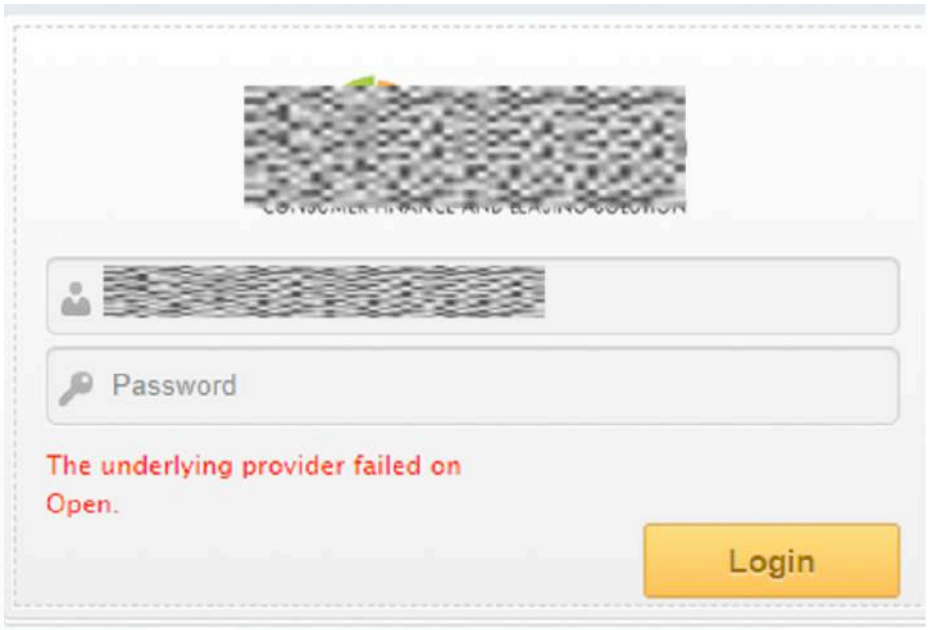
Berdasarkan pekerjaan dan tugas yang telah dilakukan dan dijalankan oleh peserta magang, berikut merupakan beberapa kendala yang telah ditemukan selama masa pengerjaan, dari periode *training* dan saat periode magang:

1. Tidak dapat melakukan koneksi kepada *server* walaupun sudah terkoneksi dengan VPN

Pengkuncian akses terhadap server penting dalam konteks keamanan dan pengelolaan infrastruktur IT perusahaan. Pertama-tama, dengan membatasi akses hanya pada jaringan internal perusahaan, keamanan dapat lebih terjaga dengan hanya memberikan akses pada pengguna yang berada dalam jaringan yang aman dan diizinkan untuk mengakses server klien. Dengan demikian, risiko serangan dari pihak luar dapat dihindari. Kedua, pengkuncian akses juga memungkinkan

adanya kontrol yang lebih baik terhadap pengelolaan infrastruktur IT perusahaan. Tim IT juga dapat lebih mudah memantau dan mengelola siapa yang memiliki akses pada sistem. Hal ini dapat membantu mendeteksi anomali atau aktivitas yang mencurigakan. Dengan kata lain, penguncian akses memungkinkan perusahaan untuk menjaga integritas sistem dengan memastikan bahwa hanya pengguna yang sah dan terotorisasi yang dapat mengakses sumber daya penting.

Meskipun demikian, hal ini menjadi hambatan serius bagi produktivitas peserta magang saat perlu melakukan pengembangan atau pengujian dari luar kantor atau dari rumah. Keterbatasan akses ini dapat menghambat peserta magang dalam mengakses sumber daya penting dan menjalankan tugas yang diberikan secara efisien. Kendala ini dapat mengganggu alur kerja peserta magang, terutama saat masalah atau pekerjaan yang harus diselesaikan bersifat *urgent*. Dapat dilihat dari gambar 3.40, terdapat *error* "*The underlying provider failed on Open*" yang berarti bahwa koneksi yang telah dicoba untuk sistem agar dapat terhubung dan menarik data dari *database* telah gagal dan ditolak. Hal ini berarti peserta magang tidak dapat melakukan koneksi ke *database* saat melakukan *login*, alhasil sistem tidak dapat menentukan apakah *username* dan *password* yang telah diinput oleh user merupakan kredensial yang benar atau salah.



Gambar 3.38 Error *underlying*

2. Kurangnya pemahaman akan metode yang digunakan saat masuk periode magang dikarenakan tidak diajarkan pada *training*

Kurangnya pemahaman atau pengetahuan tentang beberapa metode atau teknik pengembangan yang diperlukan untuk menyelesaikan tugas disebabkan oleh materi yang belum diajarkan selama pelatihan atau training. Hal ini dapat menghambat peserta magang dalam menjalankan tugas dengan efektif dan efisien, serta mengurangi kualitas hasil kerja. Kesenjangan pengetahuan ini dapat memperlambat kemajuan proyek secara keseluruhan.

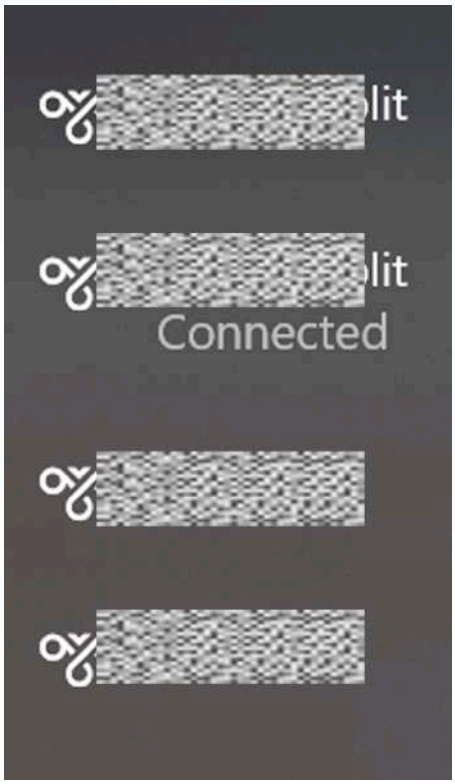
3.4 Solusi atas Kendala yang Ditemukan

Berdasarkan kendala yang sudah dijelaskan di atas, berikut merupakan solusi yang telah dilakukan untuk dapat membantu menyelesaikan masalah di atas:

1. Mencoba dan menggunakan beberapa koneksi VPN yang berbeda

Penggunaan koneksi VPN yang berbeda-beda menjadi solusi yang tepat dalam hal ini, dimana peserta magang dapat memilih salah

satu dari empat opsi yang tersedia dan menggantinya dengan yang lain disaat salah satu sedang tidak berfungsi. Dengan menggunakan koneksi VPN yang sesuai, peserta magang dapat memastikan akses yang stabil dan aman menuju server dari luar kantor.

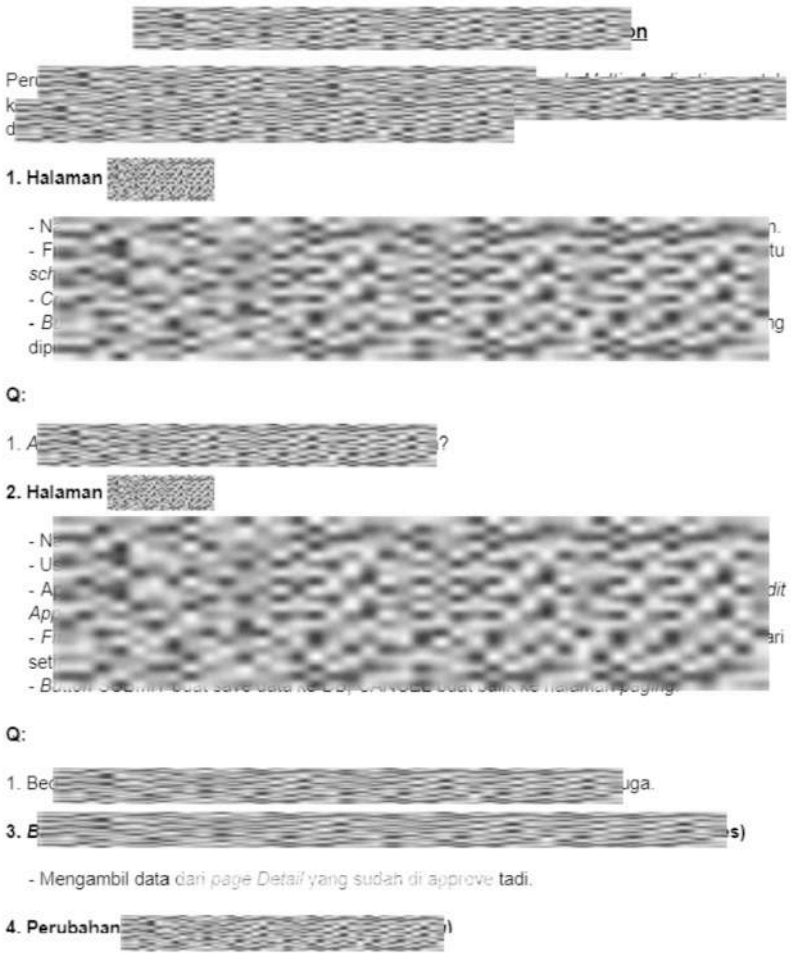


Gambar 3.39 List koneksi VPN AdIns

Dapat dilihat dari gambar 3.41, terdapat 4 koneksi VPN berbeda yang disediakan oleh AdIns. Setiap koneksi VPN tersebut memiliki fungsionalitas dan kapabilitas yang tentunya berbeda-beda. Solusi ini mampu memberikan fleksibilitas kepada peserta magang untuk tetap dapat bekerja secara efisien tanpa batasan lokasi. Selain itu, dengan memiliki beberapa opsi koneksi VPN, tim dapat menyesuaikan pengaturan sesuai dengan kebutuhan masing-masing, meningkatkan kemungkinan mendapatkan koneksi yang optimal. Dengan demikian, implementasi solusi ini diharapkan dapat meningkatkan produktivitas dan kualitas kerja tim dalam menghadapi kendala akses server dari jarak jauh.

2. Membaca referensi kode *existing* yang telah menggunakan metode yang tidak dimengerti, dan coba untuk diikuti

Membaca referensi kode milik orang lain dari framework yang relevan adalah solusi yang diupayakan. Dengan begitu, peserta magang dapat memperoleh pemahaman yang lebih mendalam tentang metode atau teknik baru. Dokumentasi dari framework tersebut dapat menjadi sumber informasi yang berharga dan memberikan panduan langkah demi langkah tentang bagaimana menggunakan metode atau teknik tertentu dengan benar. Hal ini membantu peserta magang dalam menghasilkan solusi yang relevan dan sesuai dalam proyek yang dikerjakan.



Gambar 3.40 Hasil rangkuman membaca referensi *code existing*

Tidak hanya itu, seperti yang dapat dilihat dari gambar 3.42 membaca referensi *code* dari menu-menu *existing*, peserta magang juga dapat membuat sebuah rangkuman singkat mengenai *flow* dari *code* tersebut dan fungsionalitas *code* tersebut. Dengan ini, peserta magang juga dapat memiliki sebuah catatan kecil yang dapat membantu peserta magang jika menemukan masalah yang serupa di pengembangan tiket selanjutnya. Peserta magang juga telah diberi kesempatan oleh *supervisor* untuk melakukan eksplorasi dari berbagai menu *existing* dengan tujuan dapat memperluas ilmu dari beberapa fungsionalitas menu *existing* AdIns dan metode yang digunakan.