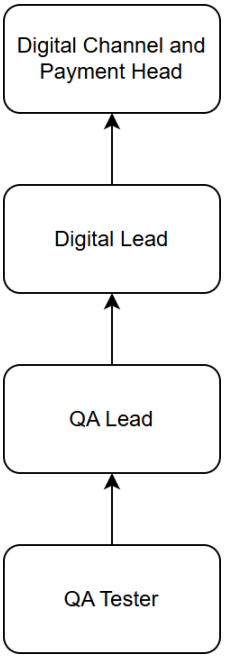
BAB III
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

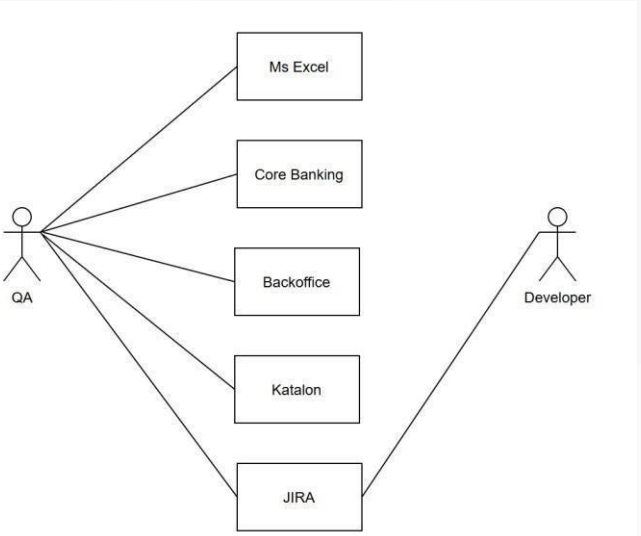
Pada PT. Bank Danamon Indonesia Tbk. terdapat kedudukan dari masing masing jabatan, Untuk posisi mahasiswa pelaksana kerja magang terdapat di bagian terbawah. Dari segi koordinasi, mahsiswa magang sering kali melakukan koordinasi dengan QA Lead yang berperan sekaligus sebagai supervisi di program magang MBKM, koordinasi yang dilakukan pada tahap ini hanyalah pembagian tugas mengenai *test case scenario* yang akan dieksekusi oleh tiap orang, selain itu juga jika ditemukan masalah didalamnya, perlu melapor sebelum membuat tiket terhadap masalah yang ditemukan. Untuk detail bagan organisasi dapat dilihat pada Gambar 3.1.



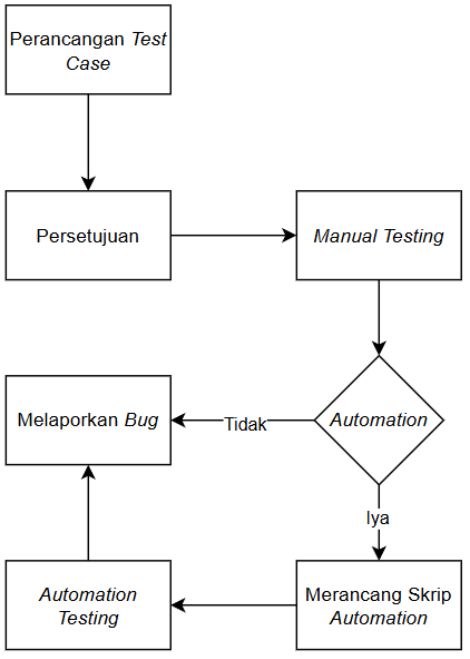
Gambar 3. 1 Kedudukan Organisasi

Selama pelaksanaan kerja magang terdapat beberapa aplikasi yang digunakan dalam melakukan koordinasi antara QA dan *Developer*. Aplikasi yang pertama digunakan oleh tim QA dalam melakukan *testing* adalah *Microsoft Excel* yang berguna sebagai pencatatan *test case* yang termasuk dalam cakupan fitur yang

akan di tes pada *sprint* yang akan berjalan. Selain *Microsoft Excel*, terdapat beberapa *tools* yang dapat dilihat pada Gambar 3.2.



Gambar 3. 2 Alur Koordinasi QA dan Developer

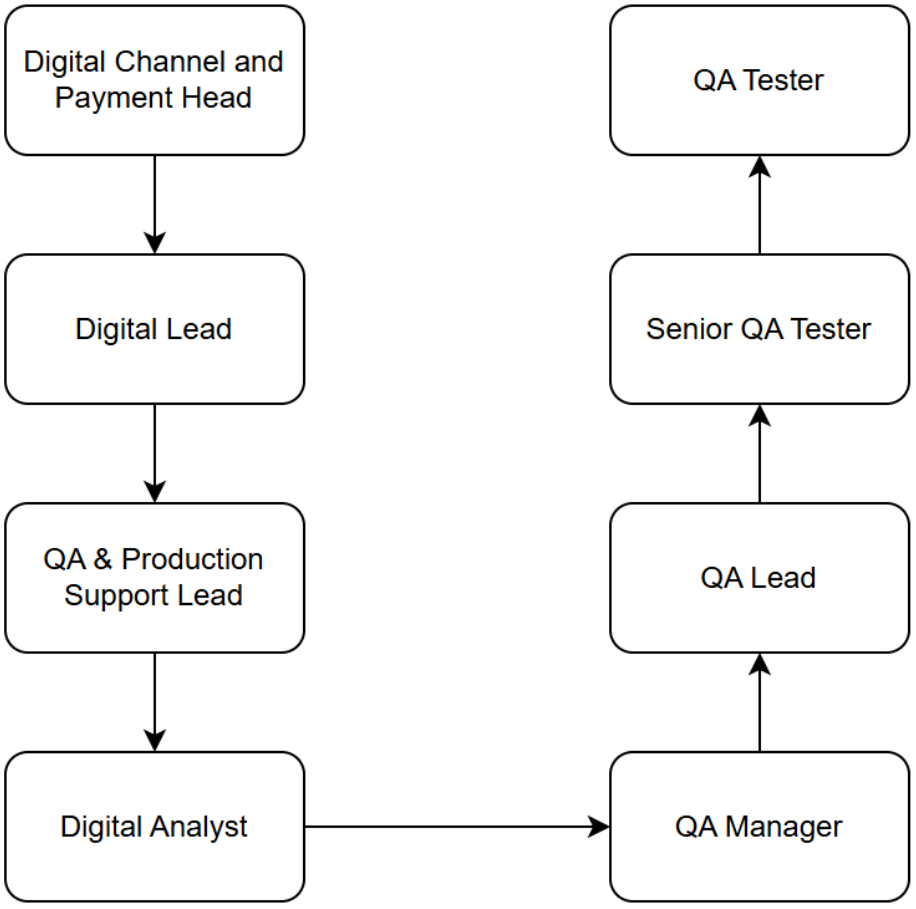


Gambar 3. 3 Siklus Kerja Tim QA

Pada pelaksanaan magang, terdapat sebuah siklus yang menjadi acuan pada proses *testing*. Seperti yang tertera di Gambar 3.3, siklus pengerjaan tim QA selalu

diawali dengan penyusunan *test case* yang didasarkan pada fitur yang sudah selesai di kembangkan. Selain itu, terdapat batasan *testing* yang akan dilakukan untuk memastikan *testing* yang dilakukan tetap berfokus kepada fitur yang dituju.

Tahap lanjutan setelah perancangan *test case* selesai adalah tahap persetujuan . Pada tahap ini, tim QA akan meneruskan berkas berisi *test case* kepada jajaran yang bertugas untuk inspeksi dan memberikan persetujuan. Persetujuan yang dilakukan bertujuan untuk memastikan *test case* yang telah dirancang sudah memenuhi cakupan yang akan dilakukan *testing*, sehingga pengujian yang dilakukan dapat dipertanggungjawabkan. Jabatan yang bertanggung jawab dalam melakukan inspeksi serta menyetujui berkas tersebut dapat dilihat pada Gambar 3.4.



Gambar 3. 4 Alur Pengecekan Test Case

Proses pengecekan dilakukan melalui birokrasi yang terstruktur dan rumit guna menjaga keabsahan berkas. Dalam proses pengecekan, para pemangku jabatan dapat memberikan kritik dan saran jika cakupan test case belum memenuhi standar. Mahasiswa magang juga dilibatkan dalam proses ini untuk memastikan tidak ada eksklusivitas dalam alur kerja di Bank Danamon Indonesia. Hal ini memberikan pengalaman menyeluruh bagi mahasiswa yang menjalankan proses kerja magang. Setelah tahap persetujuan selesai, *QA Lead* akan membagi *test case* secara merata. Selanjutnya, akan dilakukan testing serentak pada sistem operasi iOS dan Android. Setelah itu, akan dilakukan pelaporan bug dan penentuan kelayakan *automation testing*. Jika dianggap layak, pekerja magang akan segera merancang skrip *automation*. Untuk mendukung koordinasi antar anggota tim, terdapat beberapa aplikasi yang digunakan sebagai berikut :

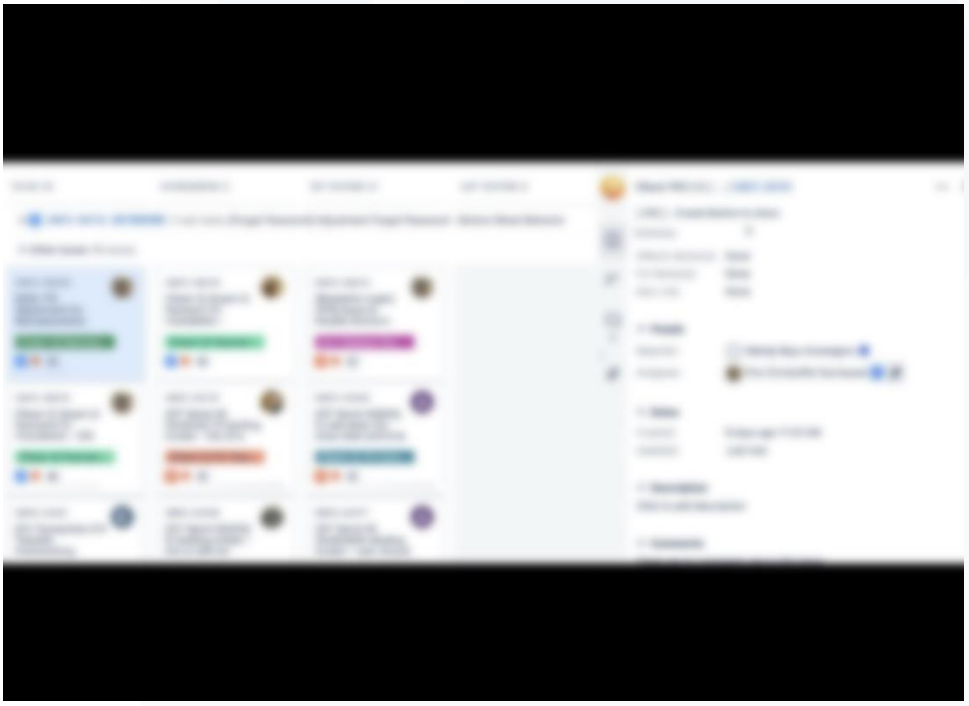
Gambar 3. 5 Tampilan Core Banking

Selain *Microsoft Excel*, terdapat aplikasi yang digunakan dalam menunjang koordinasi antar sesama tim QA dalam melakukan *testing* yang dapat dilihat pada Gambar 3.5 terdapat aplikasi *Core Banking* yang berfungsi sebagai penunjang tim QA dalam mengeksekusi tahapan *testing*, aplikasi *Core Banking* sangat diperlukan karena *testing* yang dilakukan bersinggungan dengan data nasabah dan terdapat beberapa kondisi tertentu dalam *test case* yang telah ditetapkan, sehingga dibutuhkan aplikasi *Core Banking* untuk mendukung skenario *testing*.

Gambar 3. 6 Tampilan Dalam Core Banking

Halaman *core banking* digunakan oleh tim QA agar dapat memanipulasi data nasabah buatan guna memenuhi kondisi yang dibutuhkan dalam melakukan *testing*. Manipulasi yang dilakukan pada halaman *core banking* meliputi pengubahan data nasabah buatan, pengecekan terhadap riwayat transaksi, dan pengaturan harga konversi kurs valuta asing pada sistem perbankan. Proses manipulasi bertujuan untuk memastikan proses *testing* dapat berjalan dengan optimal, detail dari halaman utama *core banking* dapat dilihat pada Gambar 3.6.

Pada sistem koordinasi PT. Bank Danamon Indonesia, terdapat aplikasi yang menjadi titik temu antara tim QA dan tim *developer* yaitu JIRA.Seperti yang tertera pada Gambar 3.7, JIRA merupakan aplikasi yang menawarkan jasa manajemen proyek dan pelacakan *bug,* yang pada Bank Danamon Indonesia digunakan sebagai aplikasi untuk melacak progres dari kinerja seluruh tim IT. Dalam konteks koordinasi, aplikasi ini berfungsi sebagai tempat dimana tim QA melaporkan *bug* yang ditemukan selama proses *testing*. Setelah dilakukannya pelaporan, tiket akan dibuat dan ditugaskan kepada tim *developer* selaku penanggungjawab dari fitur yang telah dikembangkan.



Gambar 3. 7 Tampilan JIRA

3.2 Tugas dan Uraian Kerja Magang

Pada pelaksanaan program kerja magang sebagai *Quality Assurance* di Bank Danamon Indonesia terdapat beberapa tugas yang diberikan di setiap periode waktu tertentu dengan tenggat waktu sekitar 2 minggu untuk setiap tugas yang diberikan, Tabel 3.1 merupakan serangkaian pekerjaan yang diberikan selama proses kerja magang berlangsung.

Tabel 3. 1 Deskripsi Kerja Magang

Job Description	Rincian Pekerjaan	Waktu Mulai	Waktu Selesai
- Berkolaborasi dengan tim pengembang (*developer*), QA, dan pemangku kepentingan bisnis	*Daily Standup Meeting*	Setiap Hari 10.30 WIB	Setiap Hari 11.30 WIB
	Retrospective Meeting	Setiap Bulan 17.00	Setiap Bulan 18.00

Job Description	Rincian Pekerjaan	Waktu Mulai	Waktu Selesai
untuk memahami persyaratan sistem dan memvalidasi hasil pengujian. - Berpartisipasi dalam *meeting* dengan tim untuk memberikan *feedback* dan solusi terhadap masalah yang ditemukan selama UAT.			
- Melakukan pengujian *User Acceptance Testing* (UAT) berdasarkan skenario yang telah ditentukan. - Membantu membuat, memelihara, dan memperbarui *test case* serta skenario pengujian. - Menyusun laporan hasil pengujian secara berkala. - Mengidentifikasi dan mendokumentasikan *bug*, *error*, atau masalah lainnya secara rinci.	*Sprint* 7 : *Testing* Produk A dan B	1 Oktober 2024	14 Oktober 2024
- Melakukan *regression*	Regression	21 Oktober	14 November

Job Description	Rincian Pekerjaan	Waktu Mulai	Waktu Selesai
testing untuk memastikan bahwa masalah yang telah diperbaiki tidak menyebabkan kerusakan lain. - Membantu membuat, memelihara, dan memperbarui *test case* serta skenario pengujian. - Menyusun laporan hasil pengujian secara berkala. - Mengidentifikasi dan mendokumentasikan *bug*, *error*, atau masalah lainnya secara rinci	*Testing* : Produk C	2024	2024
- Melakukan pengujian *User Acceptance Testing* (UAT) berdasarkan skenario yang telah ditentukan. - Membantu membuat, memelihara, dan memperbarui *test case* serta skenario pengujian. - Menyusun laporan hasil pengujian secara berkala.	Sprint 8 : *Testing* Produk D dan E	28 Oktober 2024	7 November 2024
	Testing Mini Release : Produk F	8 November 2024	16 November 2024
	Sprint 9 : *Testing* Produk G	21 November 2024	6 Desember 2024

Job Description	Rincian Pekerjaan	Waktu Mulai	Waktu Selesai
- Mengidentifikasi dan mendokumentasikan *bug*, *error*, atau masalah lainnya secara rinci.	*Sprint* 10 : *Testing* Produk H	12 Desember 2024	31 Desember 2024

Selama proses kerja magang, tentu terdapat 2 pekerjaan yaitu teknis maupun non teknis yang dimana pada konteks non teknis adalah sebuah pekerjaan yang dilakukan diluar dari *testing*. Pada rician Tabel 3.1 terdapat 2 pekerjaan non teknis yaitu *Daily Standup Meeting* dan *Retrospective Meeting*. Kegiatan tersebut merupakan kegiatan evaluasi harian dan bulanan dalam mengerjakan proyek yang berlangsung, untuk kegiatan non teknis dapat dijabarkan sebagai berikut :

3.2.1 *Daily Standup Meeting*

Gambar 3. 8 Daily Standup Meeting

Daily standup meeting merupakan kegiatan harian yang dilakukan setiap jam 10 pagi. Kegiatan yang dilakukan pada program kegiatan ini adalah pelaporan terhadap pekerjaan yang dilakukan kemarin dan yang akan dilakukan pada hari

kegiatan ini berlangsung. Dapat dilihat pada Gambar 3.8 merupakan gambaran dari kegiatan *Daily standup meeting* dan dihadiri oleh seluruh anggota tim. Kegiatan ini dipimpin oleh *Scrum Master* yang secara aktif menanyakan *progress* dari pengembangan fitur pada *sprint* yang sedang berjalan kepada *developer* terhadap *sprint* yang masih berjalan, dan juga menayakan kepada *business analyst* mengenai *progress* dari *requirement* untuk sprint berikutnya, untuk tim *quality assurance* pelaporan *bug* yang menghambat *testing* sehingga akan dimasukan kedalam *list* prioritas untuk segera di perbaiki.

3.2.2 *Retrospective Meeting*

Gambar 3. 9 Retrospective Meeting

Kegiatan *Retrospective Meeting* memiliki esensi yang sama dengan *Daily standup meeting* tetapi memiliki tensi yang lebih tinggi dan dilakukan setiap dua bulan. Seperti yang tertera pada Gambar 3.9, kegiatan ini bertujuan untuk membangun kerja sama antar anggota tim serta menjadi forum yang menampung keluhan tiap anggota tim dalam pelaksanaan proyek. Maka dari itu, kegiatan ini menjadi sarana refleksi agar memperbaiki komunikasi serta meningkatkan efisiensi dalam bekerja.

Kedua kegiatan yang disebutkan diatas berguna untuk membentuk komunikasi yang lebih efektif antar pekerja. Setelah menjalani kedua proses

tersebut, pekerja magang diberikan tanggung jawab untuk terjun langsung kepada proyek yang sedang berjalan. Rincian tanggung jawab yang diberikan kepada pekerja magang dapat disimak pada Tabel 3.2.

Tabel 3. 2 Timeline Pekerjaan Teknis

Bulan	Minggu	Proyek	Deskripsi Pekerjaan
September	3	-	– Melakukan pengenalan terhadap lingkungan kantor. – Membaca tentang proyek sebelumnya. – Mempelajari cara kerja para karyawan. – Mengurus keperluan penunjang testing.
	4	-	– Mempelajari cangkupan *testing* proyek yang sedang berjalan. – Melakukan *testing* secara mandiri terhadap produk X yang sedang dikembangkan. – Melakukan pengecekan serta penambahan data terhadap akun akun yang akan digunakan untuk *testing*. – Mempelajari alur dari produk yang akan di tes pada proyek berikutnya

Bulan	Minggu	Proyek	Deskripsi Pekerjaan
Oktober	1	*Sprint* 7 : *Testing* Produk A dan B	– Membaca cangkupan testing untuk *Sprint* 7. – Mempelajari alur kerja dari produk A dan B dari Figma yang telah ditetapkan oleh tim *UX Designer*. – Melakukan *testing* pada produk A dan B dengan *target* minggu 1 sebesar 50%. – Melakukan pelaporan berbentuk tiket sebanyak 4 buah tiket.
	2	*Sprint* 7 : *Testing* Produk A dan B	– Melanjutkan *testing* terhadap skenario yang telah dibentuk dengan *target* 100%. – Menemukan total 5 *bug* tambahan pada tes gelombang kedua yang dikonversikan menjadi 3 buah tiket. – Melakukan tes ulang terhadap *bug* yang sebelumnya ditemukan. – Menutup tiket dari *bug* yang sudah diperbaiki dan berjalan dengan semestinya.

Bulan	Minggu	Proyek	Deskripsi Pekerjaan
	3	*Regression Testing :* Produk C	– Melakukan tes ulang terhadap *Sprint* 7 untuk memastikan Produk A dan B 100% QA *Passed*. – Mempelajari cara kerja produk C. – Melakukan *testing* terhadap produk C dengan *target* 40%. – Melakukan pelaporan terhadap 6 *bug* dalam bentuk tiket.
	4	*Regression Testing :* Produk C + *Sprint* 8 : *Testing* Produk D dan E	– Melanjutkan *testing* Produk C dengan *target* 80%. – Melakukan tes ulang terhadap temuan *bug* minggu 3. – Melakukan *testing* secara paralel terhadap Produk D dan E dengan *target* 60%. – Melaporkan *bug* pada produk C, D, dan E dengan jumlah 7 tiket.
November	1	*Regression Testing :* Produk C + *Sprint* 8 : *Testing* Produk D dan E	– Melakukan tes ulang terhadap bug yang ditemukan pada Oktober minggu 4.

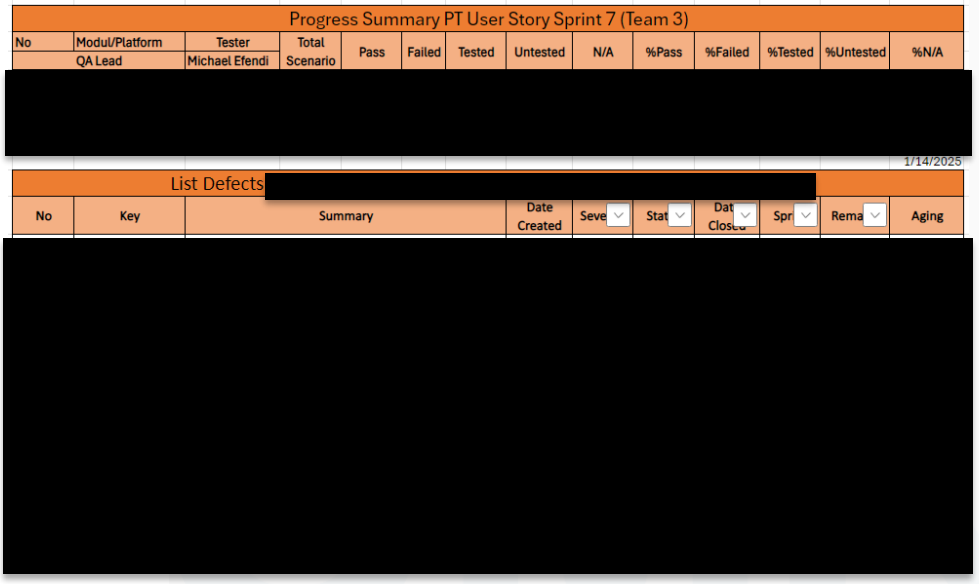
Bulan	Minggu	Proyek	Deskripsi Pekerjaan
			– Melakukan penutupan tiket terhadap *bug* yang telah di perbaiki. – Menyelesaikan *testing* untuk produk D dan E. – Memastikan produk D dan E 100% QA *Passed*.
	2	*Regression Testing :* Produk C + *Testing Mini Release* : Produk F	– Melanjutkan *testing* produk C dengan *target* 100%. – Melakukan pencatatan *bug* dengan total 12 tiket. – Melakukan test ulang terhadap *bug* yang masih ada. – Memastikan produk C 100% QA *Passed*. – Mempelajari cara kerja produk F. – Melakukan *testing* terhadap produk F dengan *target* 60%.
	3	*Testing Mini Release* : Produk F	– Menyelesaikan *testing mini release.* – Melakukan tes ulang terhadap *bug* yang sudah diselesaikan oleh tim pengembang. – Memastikan produk F 100% QA *Passed.*

Bulan	Minggu	Proyek	Deskripsi Pekerjaan
	4	*Sprint* 9 : *Testing* Produk G	– Mempelajari alur kerja produk G melalui Figma. – Melakukan *testing* terhadap produk G dengan *target* 40%. – Membuat tiket pelaporan *bug* sebanyak 20 tiket. – Melakukan tes ulang terhhadap *bug* yang telah di pecahkan oleh tim pengembang.
Desember	1	*Sprint* 9 : *Testing* Produk G	– Menyelesaikan *testing* pada produk G. – Melakukan tes ulang terhadap *bug* yang sudah di perbaiki tim pengembang. – Menutup tiket *bug* dengan kondisi alur kerja sudah berjalan dengan normal.
	2	*Sprint* 10 : *Testing* Produk H	– Mempelajari alur kerja produk H melalui Figma. – Melakukan *testing* terhadap produk H dengan *target* 50%. – Membuat tiket pelaporan *bug* sebanyak 6 tiket. – Melakukan tes ulang terhadap *bug* yang telah di

Bulan	Minggu	Proyek	Deskripsi Pekerjaan
			pecahkan oleh tim pengembang.
	3	*Sprint* 10 : *Testing* Produk H	– Melakukan *testing* terhadap produk H dengan *target* 75%. – Melakukan tes ulang terhadap *bug* yang telah di pecahkan oleh tim pengembang. – Melakukan penutupan terhadap tiket dnegan kondisi *bug* telah di perbaiki.
	4	*Sprint* 10 : *Testing* Produk H	– Menyelesaikan *testing* terhadap produk H. – Melakukan tes ulang terhadap *bug* yang telah di pecahkan oleh tim pengembang. – Melakukan penutupan terhadap tiket dnegan kondisi *bug* telah di perbaiki. – Memastikan produk H 100% QA *Passed.*

3.2.3 Sprint 7 : Produk A dan B

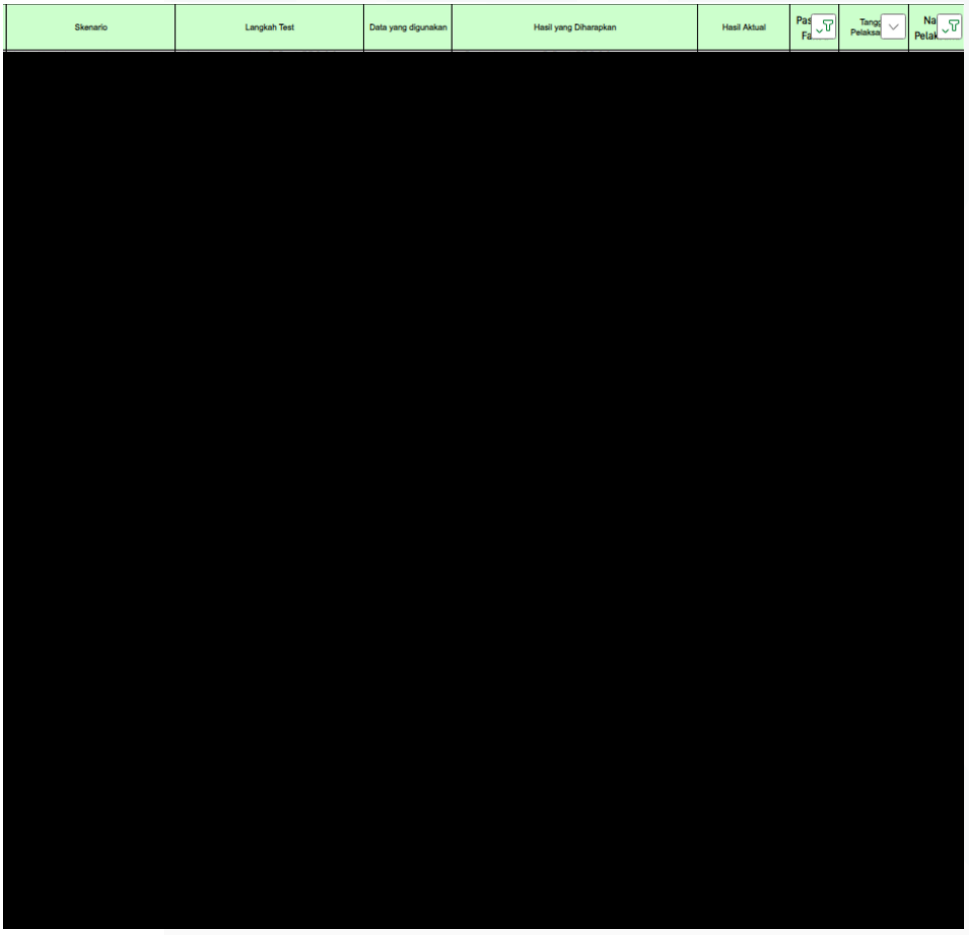
Pada *Sprint* 7, dilakukan pengujian terhadap produk A dan B sebagai bagian dari pengembangan produk layanan yang telah ada pada aplikasi *D-Bank Pro.* Pengujian ini dilakukan untuk mengalihkan layanan yang sudah ada ke dalam aplikasi yang baru. Dalam *Sprint* 7 ini, terdapat skenario tes sebanyak 648 yang dapat dilihat lebih terperinci pada Gambar 3.10.



Gambar 3. 10 Tampilan Utama Tes Skenario Sprint 7

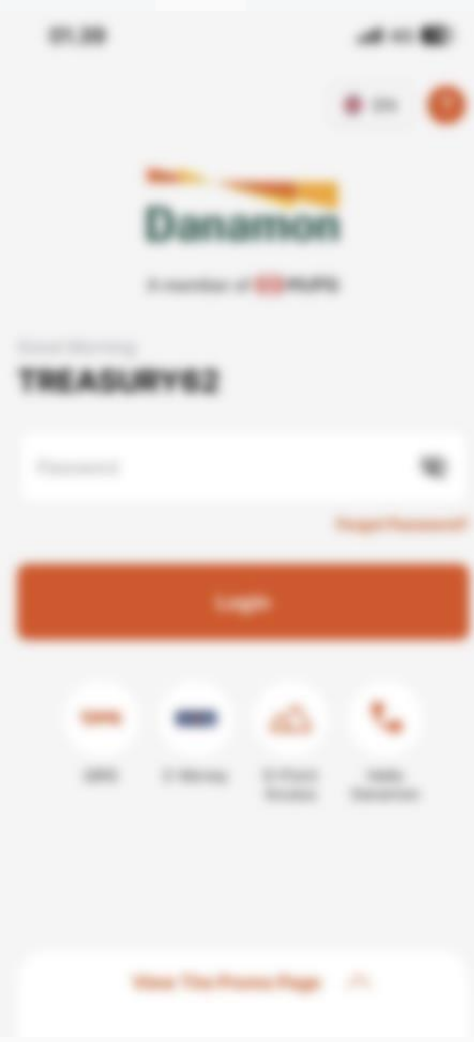
Selain tampilan utama tes skenario, terdapat juga halaman yang memuat *test case* untuk masing masing sistem operasi yaitu, iOS dan Android. Untuk tampilan skenario iOS, terdiri dari 324 *test case* serta 325 *test case* untuk Android dengan meliputi fitur produk A dan B yang akan di uji. Pada alur *testing sprint* 7, terdapat 4 orang yang bertugas dalam melaksanakan *testing* termasuk mahasiswa magang. Mahasiswa magang akan melakukan *testing* secara manual dan memberikan status *passed* dan *failed*. Jika terdapat *failed*, maka tim QA akan melaporkan kepada tim *developer* melalui aplikasi JIRA untuk segera di perbaiki. Setelah itu, akan dilakukan *testing* ulang untuk memastikan skenario tersebut berjalan dengan lancar. Selain itu, *test case* yang sudah mendapatkan status *passed* harus didokumentasikan dan dikemas dalam bentuk berkas *Microsoft Word* dan harus dilampirkan setelah

siklus testing berakhir. Untuk bentuk dari tampilan *test case* dapat dilihat pada Gambar 3.11.



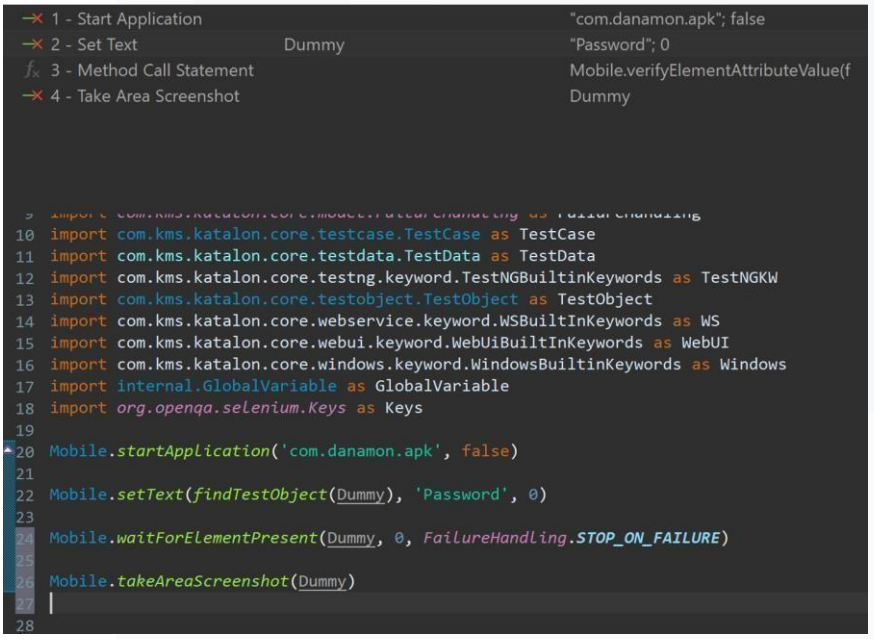
Gambar 3. 11 Tampilan Tes Skenario iOS dan Android Sprint 7

Setelah manual *testing* selesai, akan dilakukan penyusunan skrip *automation* yang dilakukan oleh pekerja magang menggunakan Katalon Studio sebagai alat utama yang digunakan. Sebelum proses penyusunan skrip *automation* dilakukan, biasanya tim QA akan melakukan pemahaman terhadap alur kerja dari produk yang akan di tes terlebih dahulu, yang dimana pada produk A dan B hanya beberapa bagian yang dapat dilakukan *automation* dengan rincian yang dapat dilihat pada Gambar 3.12.



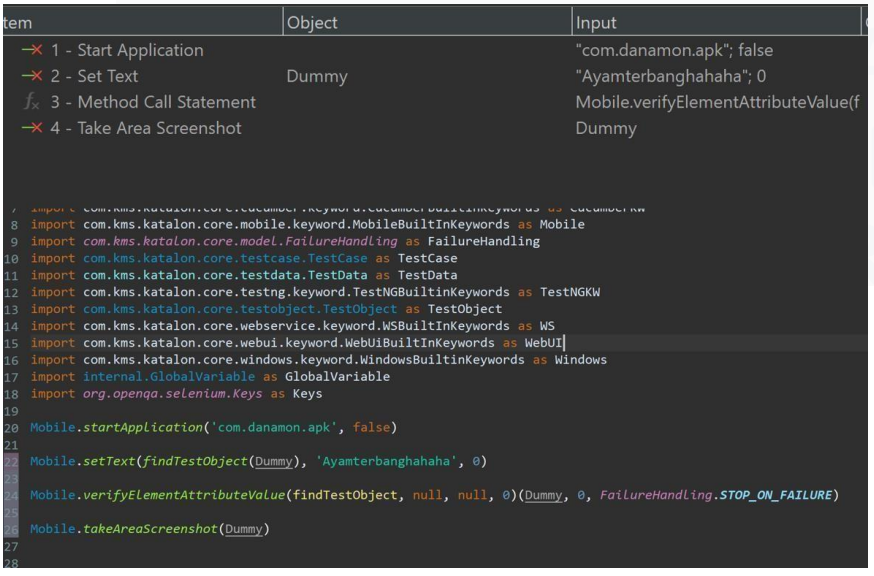
Gambar 3. 12 Tampilan Login

Bagian pertama yang dapat dibentuk *automation* adalah halaman *login.* Proses penyusunan skrip *automation* pada bagian login tergolong cukup mudah dan sederhana, dimana pengguna hanya perlu mengisi kata sandi dan menekan tombol *login*. sehingga skrip *automation* dari halaman *login* ini dapat dilihat pada Gambar 3.13. Pada gambar tersebut tertera dengan sangat jelas bahwa skrip dimulai dengan membuka aplikasi dengan *package* Danamon pada *device* android, kemudian dilakukan pengisian kata sandi untuk objek *Dummy* dengan *string "Password*". Selanjutnya, dilakukan verfikasi *login* dengan melalukan validasi respon yang diberikan oleh aplikasi terhadap tombol *login* yang di *trigger, test case* ditutup dengan pengambilan gambar untuk kepentingan dokumentasi.



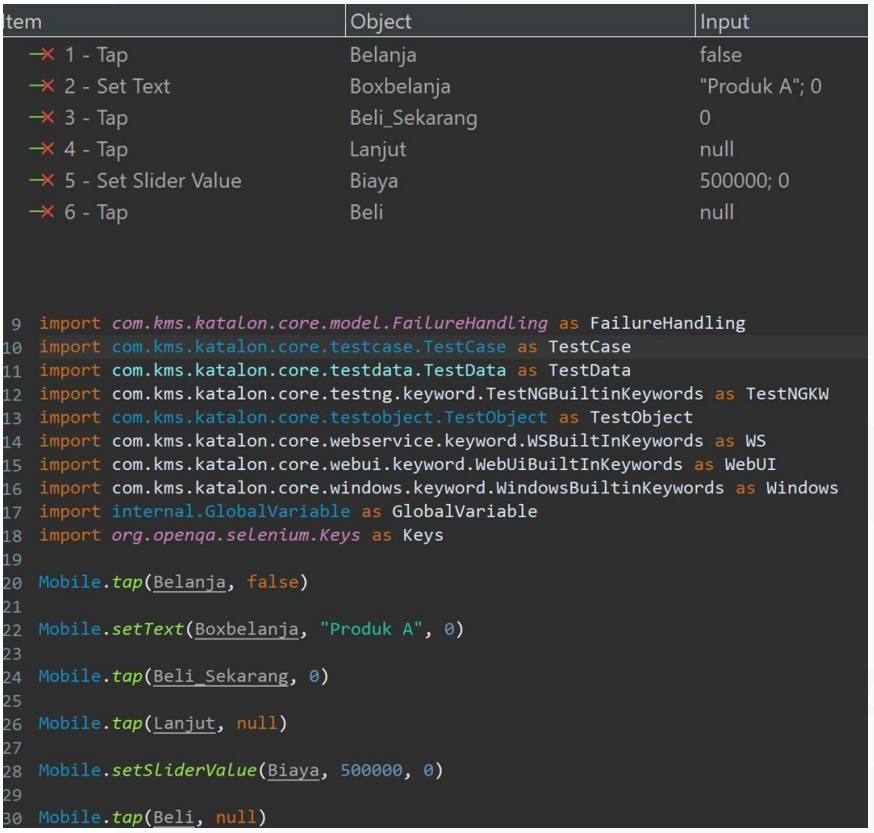
Gambar 3. 13 Automation Skrip Login

Pengerjaan yang dilakukan selanjutnya adalah pembetukan skrip yang berikutnya yaitu skrip *login* gagal. Skenario ini memiliki isi yang sama dengan login berhasil, namun memiliki perbedaan ketika memasukan kata sandi yang salah yaitu "Ayamterbanghahaha" dan menghasilkan *inline error* yang diverifikasi. Untuk rincian skrip *login* gagal dapat dilihat pada Gambar 3.14.



Gambar 3. 14 *Automation* Skrip Login Gagal

Skrip kedua yang dibentuk adalah pembelanjaan produk A. Skrip *automation* untuk pembelanjaan produk cukup sulit karena melibatkan banyak objek dan alur kerja yang cukup panjang. Pada Gambar 3.15, terdapat 6 tahap yang dirancang untuk melakukan skenario belanja. Tahapan dimulai dari halaman utama dengan menekan objek belanja hingga menekan tombol beli yang terdapat pada bagian terakhir. Hasil akhirnya akan muncul struk pembelanjaan yang akan terkirim ke *email* nasabah.



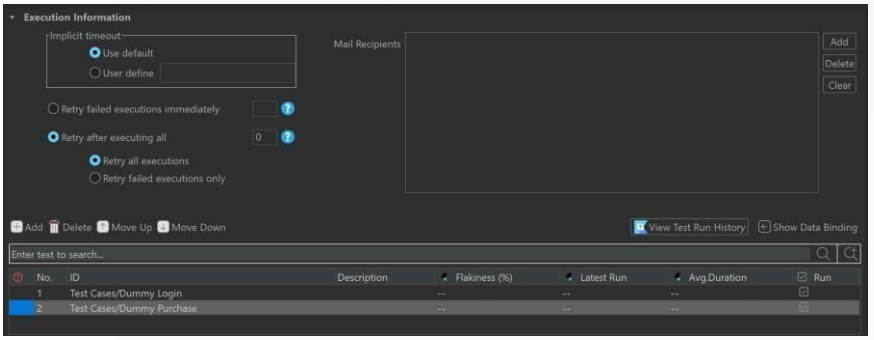
tem	Object	Input
1 - Tap	Belanja	false
2 - Set Text	Boxbelanja	"Produk A"; 0
3 - Tap	Beli_Sekarang	0
4 - Tap	Lanjut	null
5 - Set Slider Value	Biaya	500000; 0
6 - Tap	Beli	null

```
import com.kms.katalon.core.model.FailureHandling as FailureHandling
import com.kms.katalon.core.testcase.TestCase as TestCase
import com.kms.katalon.core.testdata.TestData as TestData
import com.kms.katalon.core.testng.keyword.TestNGBuiltinKeywords as TestNGKW
import com.kms.katalon.core.testobject.TestObject as TestObject
import com.kms.katalon.core.webservice.keyword.WSBuiltInKeywords as WS
import com.kms.katalon.core.webui.keyword.WebUiBuiltInKeywords as WebUI
import com.kms.katalon.core.windows.keyword.WindowsBuiltinKeywords as Windows
import internal.GlobalVariable as GlobalVariable
import org.openqa.selenium.Keys as Keys

Mobile.tap(Belanja, false)

Mobile.setText(Boxbelanja, "Produk A", 0)

Mobile.tap(Beli_Sekarang, 0)

Mobile.tap(Lanjut, null)

Mobile.setSliderValue(Biaya, 500000, 0)

Mobile.tap(Beli, null)
```

Gambar 3. 15 Tampilan Skrip Pemesanan Produk A

Setelah *test case* berhasil dirancang, sekumpulan *test case* tersebut akan digabung menjadi sebuah *test suite. Test Suite* memiliki fitur yang sangat mendukung *testing* karena dapat menentukan *test case* mana saja yang ingin diuji serta menentukan jumlah pengujian *test case* dalam satu siklus. Pada Gambar 3.16, tampilan dari *test suite* ini lebih mudah dimengerti oleh orang yang tidak terbiasa dengan skrip *automation*. Sebagai penjelasan, skrip *test suite* merupakan

pembentukan skrip dengan memanggil skrip yang telah dibentuk sebelumnya, sehingga dapat ditentukan *test case* apa saja yang akan dipanggil seperti pada Gambar 3.16, pada *test suite* terpanggil dua buah skrip yang akan di eksekusi secara berurutan yaitu *Login dan Puchase*, hal ini dilakukan untuk memantau hasil dari pengujian kedua *test case* secara berurutan.



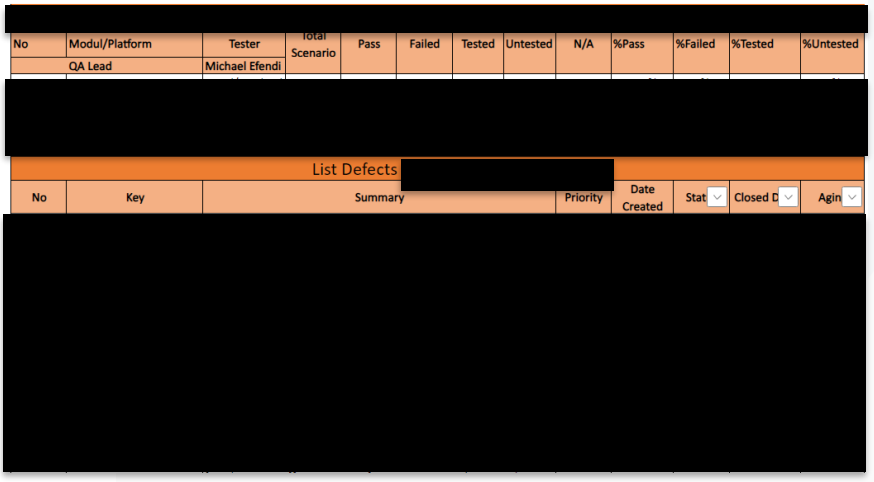
Gambar 3. 16 Tampilan Skrip Test Suite

3.2.4 *Regression Testing* : Produk C

Pada proyek *Regression Testing* terhadap produk C, sebagian besar *testing* dilakukan secara manual mengingat segala tindakan yang dilakukan pada produk C terkesan acak dan hanya beberapa memiliki pola yang tetap, sehingga hanya sebagian yang dapat dilakukan *automation*. Untuk proyek *Regression Testing* dilakukan pengujian pada tiga platform yaitu Android, iOS, dan *Web Browser*. Mayoritas pengujian pada *regression testing* dilakukan secara manual karena memiliki fitur yang sangat kompleks serta terdapat variabel yang berubah ubah, yang tidak memungkinkan *automation testing* sehingga dilakukan *testing* manual. Maka dari itu, dilakukaan perancangan *test case* dengan lebih terperinci untuk menguji hingga aktivitas terkecil yang berpotensi menimbulkan *bug* ketika diakses oleh nasabah.

Bentuk dari halaman utama tes skenario proyek *Regression Testing* dapat dilihat pada Gambar 3.17. Pada gambar tersebut, terdapat informasi berupa orang yang berpartisipasi dalam *testing*, yang berjumlah enam orang dan dibagi berdasarkan platform dengan pembagian dua orang per platform. Selain itu, terdapat persentase jumlah *tes case* yang sudah mendapatkan status *passed* dan

failed. Juga terdapat tabel berisikan *bug* yang telah dibuatkan dalam bentuk tiket JIRA dan sudah dilaporkan kepada *developer*.



Gambar 3. 17 Tampilan Utama Tes Skenario Regression Testing

Seperti pada sebelumnya, berikut merupakan bentuk dari tes skenario pada iOS untuk proyek *Regression Testing*. Pada Gambar 3.18 terdapat total 70 *test case* yang diuji di platform iOS. Selain platform iOS, terdapat juga Android dan *Web Browser* dengan total *test case* masing masing berjumlah 66 dan 65.

Gambar 3. 18 Tampilan Tes Skenario iOS Regression Testing

Selama pelaksanaan proyek *Regression Testing*, tes skenario berfokus kepada pengembangan sistem transaksi valuta asing. *Testing* yang dilakukan untuk fitur ini dilakukan secara manual karena terdapat penjagaan yang harus dilakukan secara manual. Untuk rincian dari pelaksanaan transaksi tersebut dapat disimak pada Gambar 3.19 yang melakukan *end to end testing* dengan melakukan *testing* mulai dari mengakses halaman utama, memilih opsi transaksi, memasukan pin, hingga menerima pemberitahuan bahwa transaksi telah berhasil.

Gambar 3. 19 Manual Testing Regression Testing 1

Testing yang dilakukan berikutnya adalah pengecekan nilai kurs konversi valuta asing. Proses *testing* ini dilakukan dengan menggunakana bantuan *core banking* yang berfungsi untuk memastikan nilai konversi yang diberikan pada aplikasi akurat. Setelah melakukan validasi konversi, dilakukan pengecekan notifikasi pada bagian *device* iOS untuk memastikan bahwa tampilan sudah sesuai dengan format awal. Tahap berikutnya yang dilakukan adalah memvalidasi detail pembelian pada *email* agar semua pemberitahuan menggunakan satu sumber yang

sama yaitu berdasarkan lemparan dari *backend*. Untuk rincian dari alur *testing* ini dapat disimak pada Gambar 3.20.

Gambar 3. 20 Manual Testing Regression Testing 2

Gambar 3. 21 Manual Testing Regression Testing 3

Testing berikutnya adalah pengecekan *inbox* aplikasi. Ketika transaksi berhasil, nasabah akan diberikan pemberitahuan melalui 3 jalur yaitu, *email* nasabah, *inbox* aplikasi, dan notifikasi *device*, yang bertujuan memfasilitasi informasi dari berbagai arah untuk mempermudah nasabah dalam mengakses riwayat transaksi. Pengecekan kali ini tertunjuk pada Gambar 3.21 yang menerangkan tentang tahapan tes yang dimulai dari tahapan pengecekan *inbox* aplikasi hinga memvalidasi riwayat transaksi nasabah yang berisikan rincian transaksi seperti tanggal transaksi, nominal transaksi, hingga biaya transaksi.

Gambar 3. 22 Manual Testing Regression Testing 3

Tahapan *testing* dilanjutkan dengan melakukan validasi terhadap *transaction success log*. Seperti yang tertera pada Gambar 3.22, *log* ini mencakup banyak informasi, seperti data nasabah pengirim seperti nomor rekening, tanggal pengajuan hingga nominal transfer. Selain itu, terdapat juga data rekening tertuju seperti rekening tujuan, asal bank penerima, hingga kode bank. Pengecekan pada bagian ini dilakukan untuk memastikan transaksi yang dilakukan melibatkan akun

yang valid serta jumlah transaksi yang dilakukan tidak mengalami kesalahan didalamnya.

Gambar 3. 23 Manual Testing Regression Testing 4

Tahapan berikutnya yang akan dilakukan pengecekan adalah validasi sistem *core banking*. Tahapan ini memastikan bahwa *log transaction* yang ada pada Gambar 3.22 tercatat pada sistem *core banking* yang tertera pada Gambar 3.23. Pada tahapan ini penguji akan menggunakan sistem *core banking* dengan kode khusus yang mengizinkan penguji untuk mengakses data transaksi nasabah. Pengecekan dilakukan dari dua sisi yaitu pengirim dan penerima, guna memastikan transaksi yang dilakukan tercatat dengan nominal yang sama. Proses validasi pada tahap ini mencakup nomor rekening nasabah, tanggal transaksi, kurs konversi, dan nominal transaksi. Tahapan pengecekan ini dilakukan untuk memastikan seluruh bukti transaksi yang didapatkan oleh pihak bank selaras.

Gambar 3. 24 Manual Testing Regression Testing 5

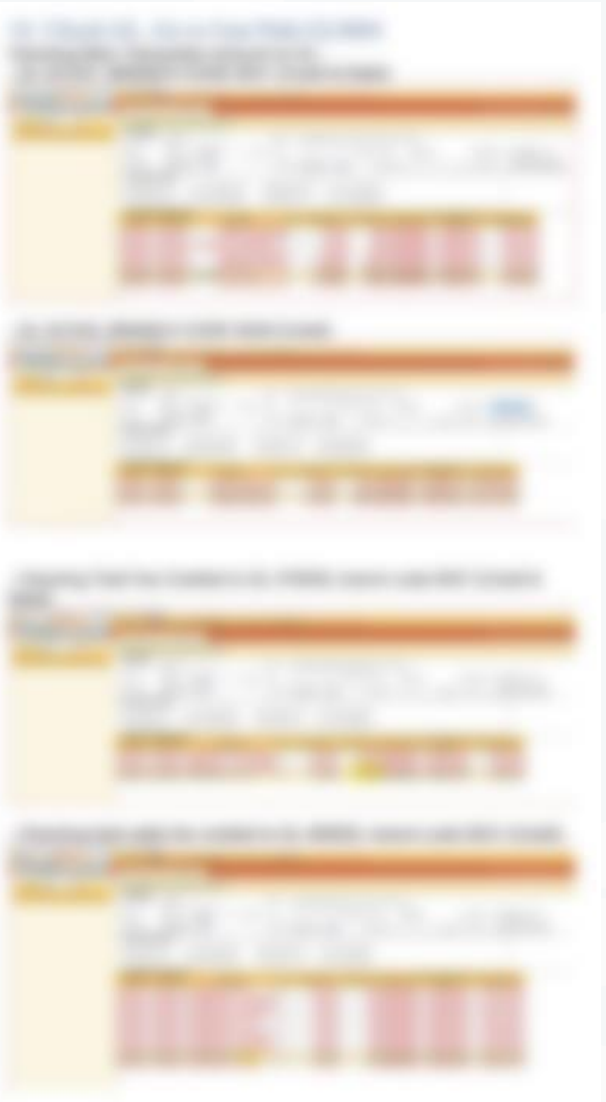
Proses pengecekan berikutnya ada pada pengecekan kurs konversi. Pengecekan ini dilakukan untuk memastikan bahwa kurs konversi yang ditampilan pada aplikasi sudah sesuai dengan kurs konversi yang ditetapkan pada sistem *core banking*. Seperti yang tertera pada Gambar 3.24, penguji memasukan kode khusus untuk mengakses daftar kurs konversi berdasarkan pasangan mata uang yang dipilih oleh nasabah, yang pada nantinya akan dilakukan pencocokan hasil konversi dari

aplikasi dengan kurs konversi yang tersedia pada sistem *core banking*. Pengecekan pada tahap ini dilakukan untuk memastikan bahwa nasabah mendapatkan nilai tukar yang adil dan tidak dirugikan.

Gambar 3. 25 Manual Testing Regression Testing 6

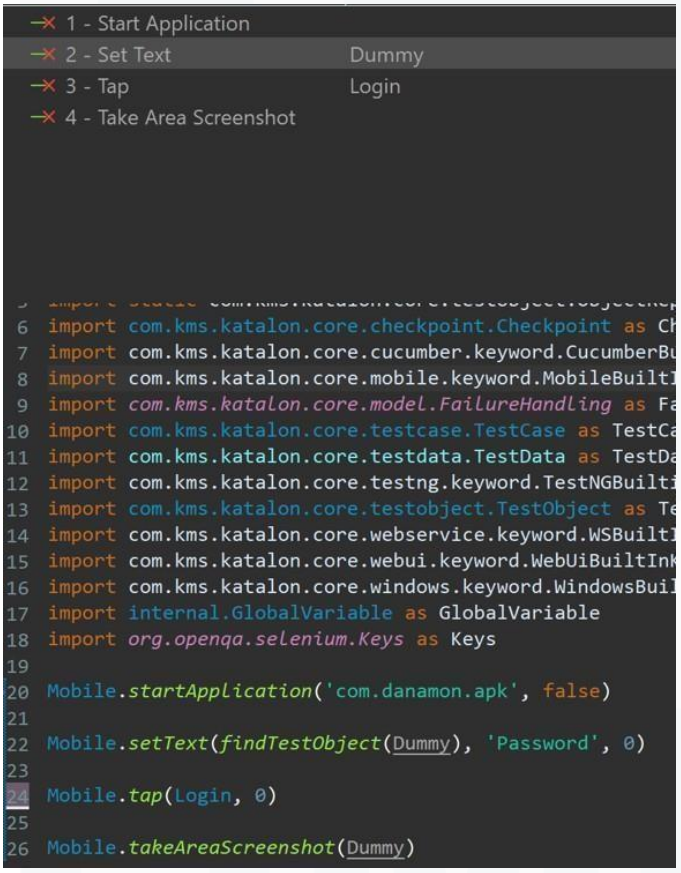
Bagian yang dilakukan pengecekan berikutnya adalah kantor cabang. Seperti yang tertera pada Gambar 3.25, tahap pengecekan ini dilakukan untuk mengetahui kantor cabang mana yang menangani transaksi tersebut. Selain itu, pengecekan kantor cabang dilakukan guna memastikan kantor cabang tersebut

memiliki *limit* dan kas mata uang yang cukup untuk melakukan transaksi. Hal ini sangat penting dilakukan agar tidak ada kesalahan pencatatan yang menyebabkan transaksi harus ditanggung oleh pihak bank.



Gambar 3. 26 Manual Testing Regression Testing 7

Pengecekan yang dilakukan terakhir adalah validasi biaya layanan. Dalam transaksi valuta asing, terdapat tingkatan yang didasari oleh nominal transaksi yang dilakukan oleh nasabah. Pengecekan yang dapat dilihat pada Gambar 3.26 dilakukan untuk memastikan aplikasi mengeluarkan nilai yang sama dengan sistem *core banking* berdasarkan tingkatan yang telah ditentukan dalam transaksi.

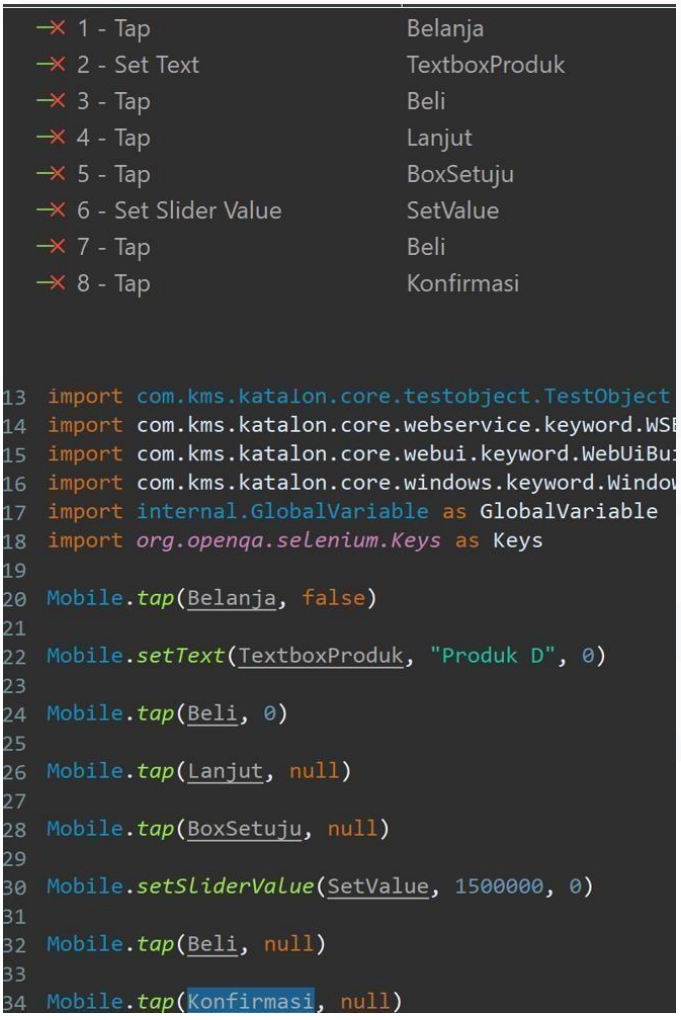


Gambar 3. 27 Automation Skrip Regression Testing 1

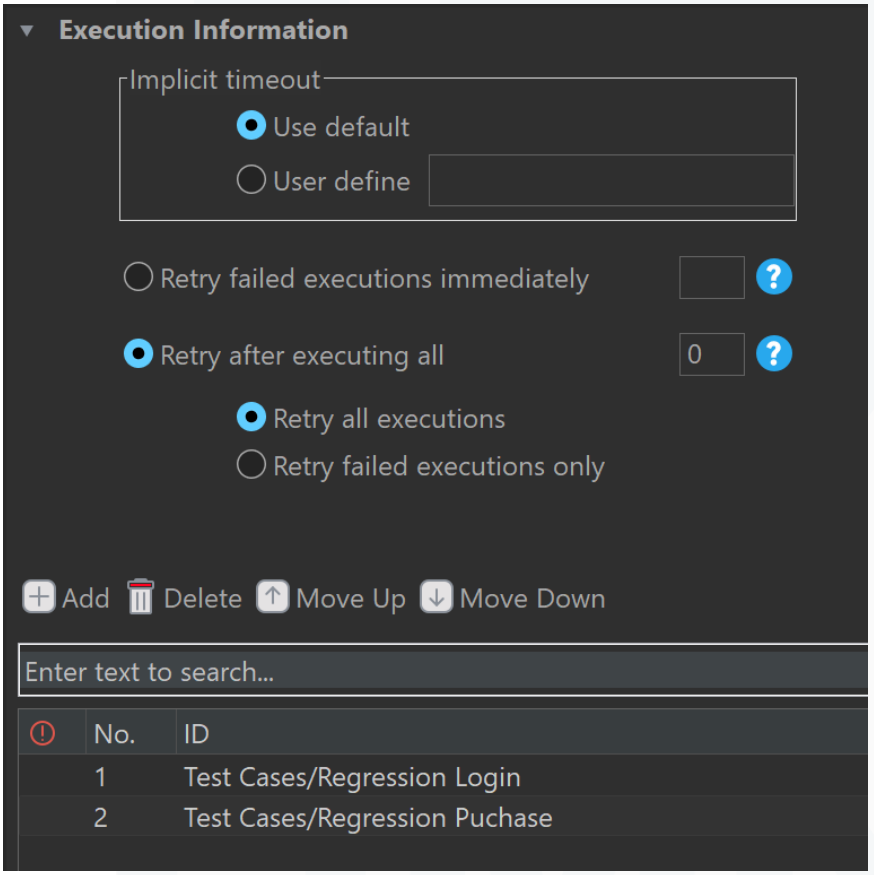
Skrip pertama yang dirancang pada proyek *Regression Testing* adalah *login*. Seperti yang dapat dilihat pada Gambar 3.27,terdapat langkah langkah untuk halaman *login* yang cukup sederhana yaitu memberikan perintah untuk memasukan kata sandi dan menekan tombol *login*. Setelah tahapan tersebut berhasil dilaksanakan, akan dilakukan pengambilan gambar melalui perintah *Mobile.takeAreaScreenshot* yang secara langsung mengambil gambar dari hasil login dan disimpan kedalam *folder local* yang telah ditentukan pada pengaturan aplikasi.

Setelah masuk halaman utama, pengguna akan diberikan perintah untuk memasuki halaman utama. Pada halaman utama, terdapat berbagai fitur produk yang telah dikembangkan sebelumnya. Untuk *testing* kali ini akan dilakukan pembelanjaan Produk D. Seperti yang tertera pada Gambar 3.28, pengguna akan berpindah ke halaman belanja dengan perintah *Mobile.tap* pada objek Belanja.

Setelah itu, akan dilakukan pencarian menggunakan fitur *search* dengan kata kunci "Produk D". Setelah ditemukan Produk D, pengguna akan diarahkan kepada halaman beli ketika menjalankan skrip *Mobile.tap* Beli. Setelah menekan beli, pengguna akan diarahkan kedalam halaman syarat dan ketentuan pembelian produk dengan *checkbox* di akhir yang menandakan bahwa pengguna mengetahui tentang syarat dan ketentuan tersebut. Pada akhirnya pengguna akan menekan tombol beli dan diarahkan kepada halaman detail pembelian yang dikirim melalui *email*, *inbox*, dan notifikasi *device* pengguna. Penggunaan *automation testing* pada skrip ini dilakukan karena memiliki alur yang cukup sederhana dan tidak ada variabel yang berubah ubah didalamnya.



Gambar 3. 28 Automation Skrip Regression Testing 2



Gambar 3. 29 Automation Skrip Regression Testing 3

Persis seperti pengerjaan *automation* pada proyek *Sprint* 7, ketika *test case* sudah berhasil dirancang maka akan dibuat *test suite*. *Test suite* yang dirancang pada proyek *Regression Tesing* memiliki beberapa fitur diantaranya adalah *login* dan *purchase* yang dapat dilihat pada Gambar 3.29. Setelah *Test suite* berhasil dibentuk, akan dilakukan penjalanan *test suite* selama waktu yang ditentukan dan tidak lupa menambahkan fitur *screenshot* agar dapat melihat *bug* yang dibetukan pada tahapan *testing*.

Pengerjaan *automation* pada PT. Bank Danamon Indonesia yang dilaksanakan menggunakan *Katalon Studio* pada proyek *Regression Testing* dan *Sprint 7 Testing*, menghasilkan temuan berupa 32 unit *bug*. *Bug* yang ditemukan mayoritas berada pada proyek *Sprint 7,* yang dimana *bug* tersebut bersifat *intermittent* yang berarti muncul secara acak dan hanya dapat ditemukan jika

dilakukan percobaan berkali kali. Rincian dari *bug* yang ditemukan dapat dilihat pada Gambar 3.30

	7-Oct-24	Critical	Closed	10-Oct-24	Sprint 7
	8-Oct-24	Medium	Closed	10-Oct-24	Sprint 7
	8-Oct-24	Low	Closed	9-Oct-24	Sprint 7
	8-Oct-24	Critical	Closed	11-Oct-24	Sprint 7
	8-Oct-24	Low	Closed	9-Oct-24	Sprint 7
	8-Oct-24	Medium	Closed	10-Oct-24	Sprint 7
	9-Oct-24	Low	Closed	10-Oct-24	Sprint 7
	9-Oct-24	Critical	Closed	11-Oct-24	Sprint 7
	9-Oct-24	Medium	Closed	10-Oct-24	Sprint 7
	9-Oct-24	Medium	Closed	11-Oct-24	Sprint 7
	9-Oct-24	Low	Closed	10-Oct-24	Sprint 7
	9-Oct-24	Medium	Closed	9-Oct-24	Sprint 7
	9-Oct-24	Medium	Closed	10-Oct-24	Sprint 7
	10-Oct-24	Low	Closed	11-Oct-24	Sprint 7
	10-Oct-24	Medium	Closed	16-Oct-24	Sprint 7
	10-Oct-24	Critical	Closed	21-Oct-24	Sprint 7
	10-Oct-24	Medium	Closed	14-Oct-24	Sprint 7
	11-Oct-24	Low	Closed	15-Oct-24	Sprint 7
	11-Oct-24	Medium	Closed	16-Oct-24	Sprint 7
	11-Oct-24	Critical	Closed	14-Oct-24	Sprint 7
	11-Oct-24	Critical	Closed	15-Oct-24	Sprint 7
	11-Oct-24	Low	Closed	14-Oct-24	Sprint 7
	11-Oct-24	Low	Closed	16-Oct-24	Sprint 7
	11-Oct-24	Medium	Closed	16-Oct-24	Sprint 7
	14-Oct-24	Medium	Closed	14-Oct-24	Sprint 7
	14-Oct-24	Critical	Close	22-Oct-24	Sprint 7
	15-Oct-24	Medium	Closed	15-Oct-24	Sprint 7
	15-Oct-24	High	Closed	15-Oct-24	Sprint 7
	15-Oct-24	Medium	Close	28-Oct-24	Sprint 7
	15-Oct-24	Low	Closed	16-Oct-24	Sprint 7
	16-Oct-24	Medium	Closed	16-Oct-24	Sprint 7
	18-Oct-24	Low	Close	4-Nov-24	Sprint 7

Gambar 3. 30 Rincian Bug Hasil Automation

Pada proses magang terjadi *testing* sebanyak 483 dengan rincian 416 *Passed* dan 67 *Failed*. Seperti yang tertera pada Gambar 3.31, proses ini melibatkan 74 *test case* dengan 46 *Passed* dan 28 *Failed*, mayoritas dari *test case* yang berstatus *failed* terbentuk karena kesalahan pada saat menyusun skrip sehingga perlu dilakukan revisi. Setelah dilakukan revisi, *test case* akan kembali dijalankan untuk menguji kelayakan *test case* tersebut. Dari temuan ini, proses pengembangan *automation* dengan Katalon Studio dinyatakan berhasil karena ditemukan *bug* yang berada

diluar jangkauan *test case* yang telah dirancang oleh tim *Developer*, selain itu *bug* yang ditemukan bersifat tidak menentu sehingga ketika dijalankan 300 percobaan, *bug* tersebut hanya muncul dibeberapa fase pengujian saja.

Gambar 3. 31 Execution Result

Sumber : Katalon Studio

3.3 Kendala yang Ditemukan

Pada proses *testing* yang dilaksanakan selama 4 bulan tentu terdapat kendala selama proses pengerjaanya. Kendala yang dialami selama proses kerja magang bekutat pada ketidaktersediaan pegawai yang memiliki kememapuan dalam

menjalankan *automation testing*, jarak tempuh yang jauh, hingga jadwal kerja yang cukup padat. Kendala yang ditemukan selama proses kerja magang telah dirangkum dalam bentuk tabel dan tertera pada Tabel 3.3.

Tabel 3. 3 Kendala yang Ditemukan

No	Kendala	Skala Prioritas	Deskripsi Kendala
1	Kurangnya mentor untuk automation testing	Tinggi	Mentor dalam magang tentu sangat penting dalam proses pembelajaran mahasiswa selama menjalani proses magang, sayangnya selama proses magang ini. kurang banyak mentor yang dapat mengajarkan *automation testing.*
2	Jarak antara tempat tinggal yang jauh	Tinggi	Karena jarak tempat tinggal dan kantor sekitar 40km maka memerlukan waktu tempuh sekitar 2 jam, sehingga berdampak pada jam tidur dan kelelahan yang cukup besar ketika bekerja
3	Kurang inisiatif	Tinggi	Hal ini terjadi dipengaruhi oleh poin sebelumnya yaitu jarak tempuh yang jauh mengakibatkan kurangnya tidur sehingga menyebabkan kelelahan yang luar biasa

No	Kendala	Skala Prioritas	Deskripsi Kendala
			sehingga dirasa tidak ada tenaga untuk memulai tindakan
4	Jadwal kerja yang cukup padat	Sedang	Jadwal kerja yang cukup padat membuat menjadi kesusahan untuk membagi waktu dalam mengerjakan laporan, hal ini juga didukung oleh waktu tempuh antara tempat tinggal dan kantor yang memakan waktu 2 jam sehingga tidak memungkinkan untuk mengerjakan laporan setelah mencapai rumah dan juga selalu diadakan *Standup Meeting* pada jam 10 pagi yang bertabrakan dengan jadwal bimbingan magang.

3.4 Solusi atas Kendala yang Ditemukan

Selain menemukan masalah, tentu seiring berjalannya waktu magang akan ditemukan solusi diantara masalah yang ada. Selama proses magang, mahasiswa sudah melakukan berbagai cara untuk beradaptasi dengan lingkungan. Setelah melakukan kerja magang selama 4 bulan, terdapat solusi yang ditemukan oleh mahasiswa yang dirangkum dalam bentuk Tabel 3.4.

Tabel 3. 4 Solusi Atas Kendala yang Ditemukan

No	Deskripsi Kendala	Prioritas	Solusi
1	Mentor dalam magang tentu sangat penting dalam proses pembelajaran mahasiswa selama menjalani proses magang, sayangnya selama proses magang ini. kurang banyak mentor yang dapat mengajarkan automaation testing	Tinggi	Mencoba mempelajari dari media media seperi Youtube atau komunitas yang menggemari *automation*.
2	Karena jarak tempat tinggal dan kantor sekitar 40km maka memerlukan waktu tempuh sekitar 2 jam. Selain proses tempuh yang mengharuskan mahasiswa untuk melakukan perjalan dengan 3 transportasi yaitu Motor-Kereta-Bis, sehingga berdampak pada jam tidur dan kelelahan yang cukup besar ketika bekerja	Tinggi	Mengatur kualitas tidur

No	Deskripsi Kendala	Prioritas	Solusi
3	Hal ini terjadi dipengaruhi oleh poin sebelumnya yaitu jarak tempuh yang jauh mengakibatkan kurangnya tidur sehingga menyebabkan kelelahan yang luar biasa sehingga dirasa tidak ada tenaga untuk memulai tindakan	Tinggi	Mengatur kualitas tidur dan mencoba untuk menjadi lebih insiatif
4	Jadwal kerja yang cukup padat membuat menjadi kesusahan untuk membagi waktu dalam mengerjakan laporan, hal ini juga didukung oleh waktu tempuh antara tempat tinggal dan kantor yang memakan waktu 2 jam sehingga tidak memungkinkan untuk mengerjakan laporan setelah mencapai rumah dan juga selalu diadakan *Standup Meeting* pada jam 10	Sedang	Memanfaatkan hari sabtu dan minggu untuk mengerjakan laporan magang sebaik mungkin

No	Deskripsi Kendala	Prioritas	Solusi
	pagi yang bertabrakan dengan jadwal bimbingan magang.		