

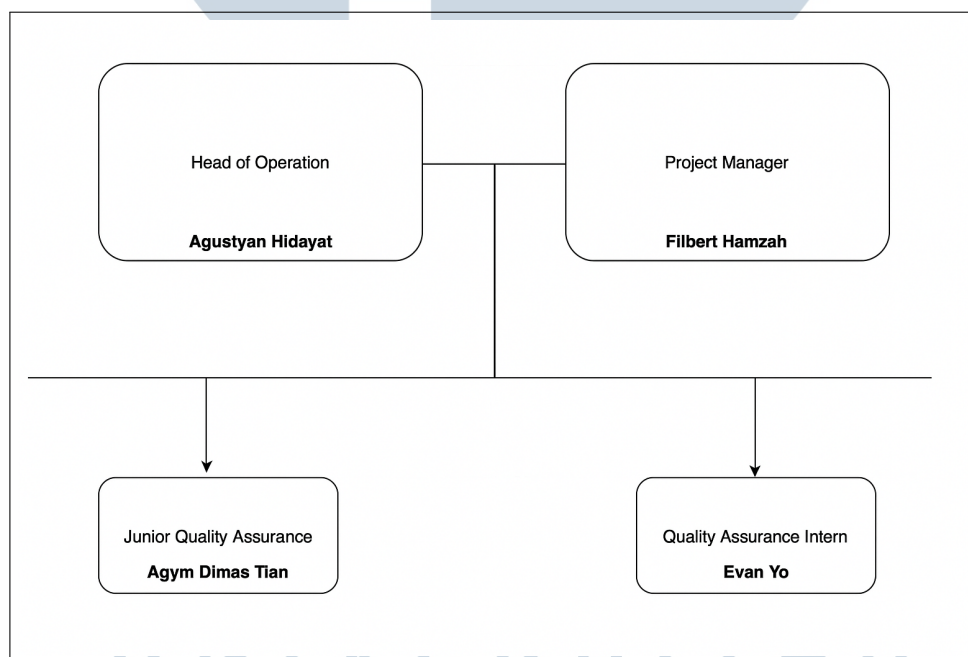
BAB 3

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Selama proses kerja magang di PT. Ganda Visi Jayatama, kedudukan yang ditempati adalah sebagai *Quality Assurance Intern* yang termasuk ke dalam divisi *QA Engineer*, yang berada di bawah *Head of Operations*. Dalam posisi ini terdapat 2 *Junior Quality Assurance*, dan 1 orang yang berperan sebagai *Quality Assurance Engineer Intern* yang dibimbing oleh Bapak Agustyan Hidayat. Dalam proses magang ini, terdapat beberapa tugas yang diberikan, yaitu mengecek seluruh fungsional *website* dan *apps* secara manual maupun otomatis, membuat *report* dalam proses *automation*, serta membuat *test case*.

Struktur organisasi perusahaan Concise bisa dilihat pada 3.1.



Gambar 3.1. Struktur organisasi perusahaan Concise

Dalam proyek ini, terdapat beberapa struktur tim proyek yang dapat dilihat pada Gambar 3.1. Proyek ini berisikan 1 orang sebagai *Project Manager*, yaitu Bapak Filbert Hamzah, 2 orang yang berperan sebagai *Frontend Engineer*, yaitu Bapak Adhitya Bagus Wicaksono dan Bapak Enrico Nathaniel, serta 1 orang *Backend Engineer*, yaitu Bapak Alwin Alamsyah. Selama proses ini, tim sering melakukan

sprint planning. *Sprint planning* sendiri dilakukan setiap 2–3 minggu sekali, yang bertujuan untuk mengetahui sejauh mana progres tugas yang telah diberikan. Setelah *sprint planning*, akan dilanjutkan dengan *sprint closing*, yang bertujuan untuk menutup dan mengevaluasi *sprint* sebelumnya. Selain itu, setiap 2–3 minggu sekali juga diadakan sesi *1-on-1* dengan *Technical Lead* untuk membahas kendala-kendala yang dihadapi selama mengerjakan proyek. Dalam sesi ini, *Technical Lead* akan memberikan *feedback* terkait permasalahan yang dialami, sehingga proses pengembangan dapat berjalan lebih baik ke depannya.

3.2 Tugas yang Dilakukan

Selama pelaksanaan kerja magang, diajari beberapa hal seperti penggunaan *Playwright*, *JavaScript*, *TypeScript*, dan lain-lain. Adapun tugas-tugas yang di kerjakan di antaranya adalah:

- Membuat *automation* untuk melakukan pengecekan pada *website* MRO.
- Melakukan pengecekan fungsional secara manual di aplikasi MRO.
- Membuat *Test Case* versi 1.2.
- Membuat *report* dari hasil *automation website* MRO.
- Membuat *Test Case* versi 1.3.

3.3 Uraian Pelaksanaan Magang

Periode magang dimulai pada tanggal 3 Januari 2025 hingga 3 Juni 2025 di PT. Ganda Visi Jayatama (Concise), dengan mengerjakan beberapa tugas yang bertujuan untuk melakukan pengecekan terhadap *website* maupun aplikasi MRO. Pengecekan ini bertujuan untuk memastikan bahwa seluruh fitur dapat berjalan dengan lancar dan tanpa bug.

Untuk pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

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

Minggu Ke -	Pekerjaan yang dilakukan
1	Masa pengenalan lingkungan kerja, peraturan kerja dan membaca boilerplate playwright

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

Minggu Ke -	Pekerjaan yang dilakukan
2	Membaca dan memahami code yang dibikin oleh <i>Junior Quality Assurance</i> dengan project yang lain
3	Mencoba untuk membuat automation dibagian <i>login</i> di <i>website HRIS</i>
4	Membuat <i>test case</i> untuk <i>website MRO</i>
5	Membuat <i>code automation website MRO</i> di halaman <i>Login page</i>
6	Membuat <i>Code Automation website MRO</i> di halaman <i>User N Role</i>
7	Membuat <i>Code Automation Website MRO</i> di halaman <i>Master Data</i>
8	Membuat <i>Code Automation Website MRO</i> di <i>Vessel Requisition</i>
9	Memperbaiki <i>code Automation</i> sesuai dengan SOP Perusahaan
10	Melakukan <i>Testing</i> dari awal hingga akhir
11	Membuat hasil <i>report</i> dari <i>test automation</i> pada <i>website MRO</i>
12	<i>Deplyoment</i> melalui Github perusahaan
13	Membuat <i>test case</i> versi 1.3 untuk <i>website MRO</i>
14	Melakukan <i>testing manual</i> pada aplikasi MRO
15	Membuat serta memberikan hasil <i>report ke Project Manager</i>
16	Melakukan <i>testing manual</i> versi 1.2 pada aplikasi MRO

Pada saat mengerjakan tugas dan proyek di perusahaan, mahasiswa juga menggunakan beberapa *tools* untuk kebutuhan dalam menjalankan tugas, yaitu:

- Playwright

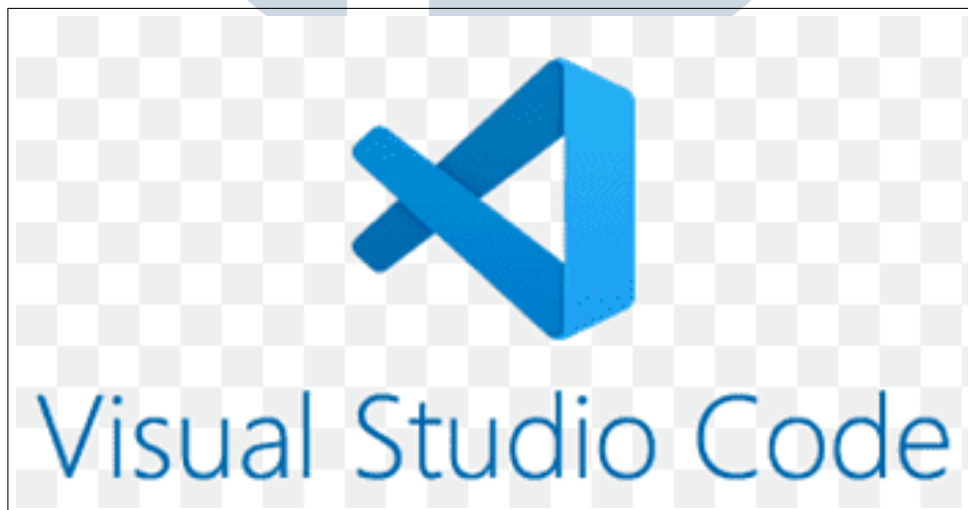


Gambar 3.2. Logo Playwright

Gambar 3.2 Playwright merupakan alat *open-source* untuk otomatisasi *browser* yang dikembangkan oleh Microsoft dan mulai diperkenalkan pada

tahun 2020. Alat ini dirancang untuk melakukan pengujian *end-to-end* pada aplikasi web modern. Keunggulan Playwright terletak pada kemampuannya yang lintas platform, lintas *browser*, dan mendukung berbagai bahasa pemrograman. Salah satu fitur menariknya adalah *auto-waiting*, yang membantu proses pengujian menjadi lebih mudah, bahkan pada proyek yang kompleks. Playwright juga dikenal memiliki performa eksekusi yang cepat, sehingga banyak dipilih untuk kebutuhan *testing* otomatis. Dengan memanfaatkan *library Node.js*, Playwright menyediakan *API* untuk mengendalikan *browser* melalui protokol *DevTools*. Alat ini mendukung pengujian di *browser* populer seperti *Chromium*, *Firefox*, dan *WebKit*, serta dapat digunakan dengan bahasa pemrograman seperti *JavaScript*, *TypeScript*, *Python*, *Java*, dan *.NET*. Salah satu fitur andalannya adalah Playwright *Codegen*, yang bisa merekam interaksi pengguna di situs web dan secara otomatis menghasilkan kode pemrograman dari aktivitas tersebut.. [1]

- Visual Code Studio



Gambar 3.3. Logo Visual Code Studio

Gambar 3.4 Visual Studio Code merupakan editor kode yang dibuat oleh Microsoft, salah satu perusahaan teknologi terkemuka. Editor ini bisa digunakan di berbagai sistem operasi *desktop*, termasuk *macOS*, *Windows*, dan *Linux*. Visual Studio Code terkenal karena fleksibilitasnya yang tinggi serta dukungannya terhadap berbagai bahasa pemrograman, seperti *JavaScript*, *TypeScript*, *Python*, *PHP*, hingga *Java*. [2]

- Jira



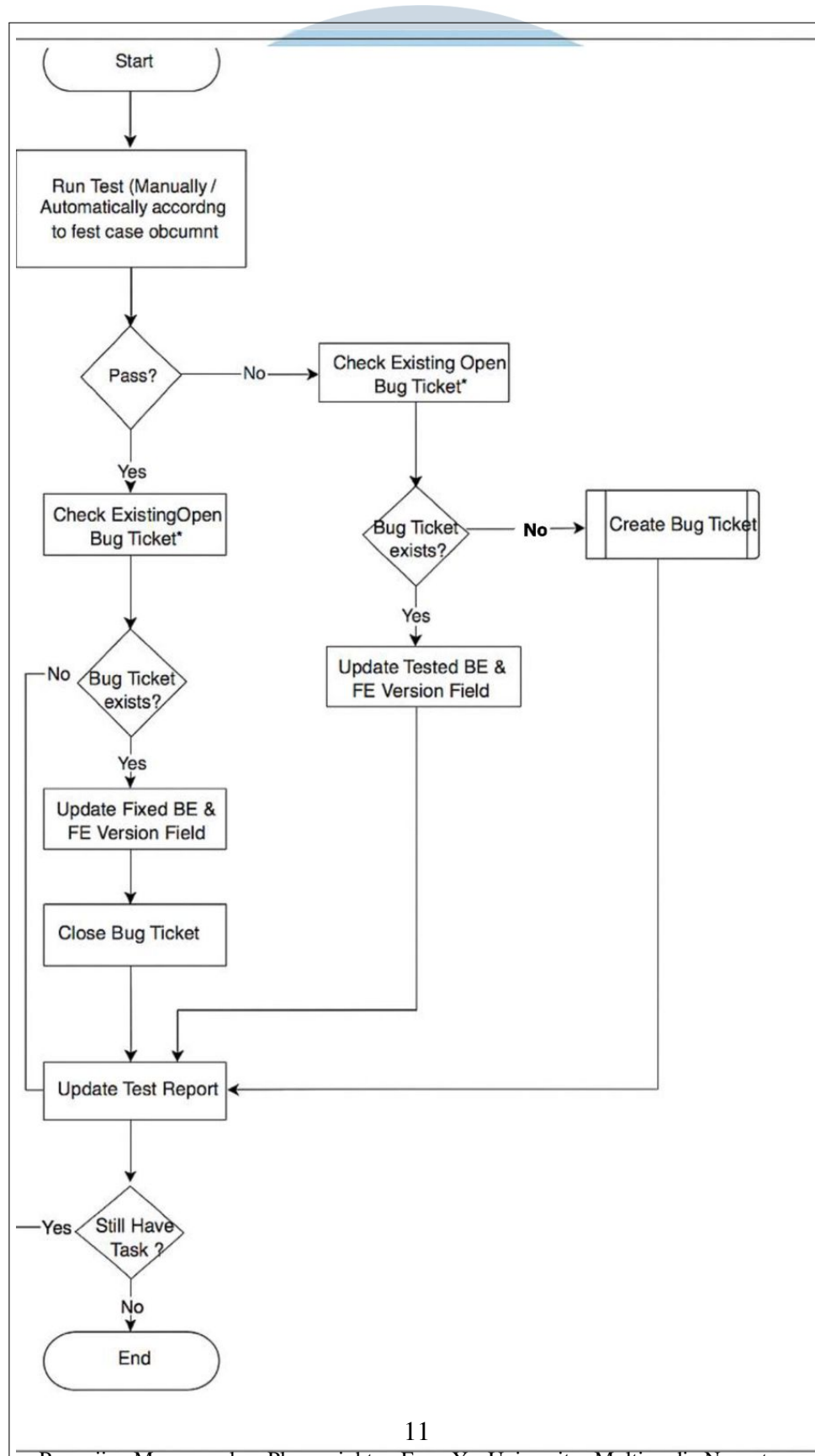
Gambar 3.4. Logo Jira

Gambar 3.5 Jira merupakan sebuah alat bantu dalam pengembangan *perangkat lunak* yang berfungsi untuk mengelola dan memantau *task*, *feature*, serta *bug*. Aplikasi ini dikembangkan oleh Atlassian, sebuah perusahaan teknologi yang berbasis di Australia. Jira menyediakan berbagai macam fitur yang mempermudah proses pengelolaan proyek, terutama dalam konteks pengembangan *perangkat lunak*.

Selain itu, Jira juga memfasilitasi perusahaan dalam memantau kebutuhan proyek dan perkembangannya secara menyeluruh. Dukungan komunitas dan keberadaan berbagai *forum* menjadikan penggunaannya semakin luas dan mudah dipelajari. Secara umum, Jira banyak digunakan dalam proses *Application Lifecycle Management (ALM)*, manajemen pengujian, dan pengelolaan proyek secara keseluruhan, terutama untuk memantau *bug* serta permasalahan lainnya yang muncul dalam pengembangan aplikasi.[3].

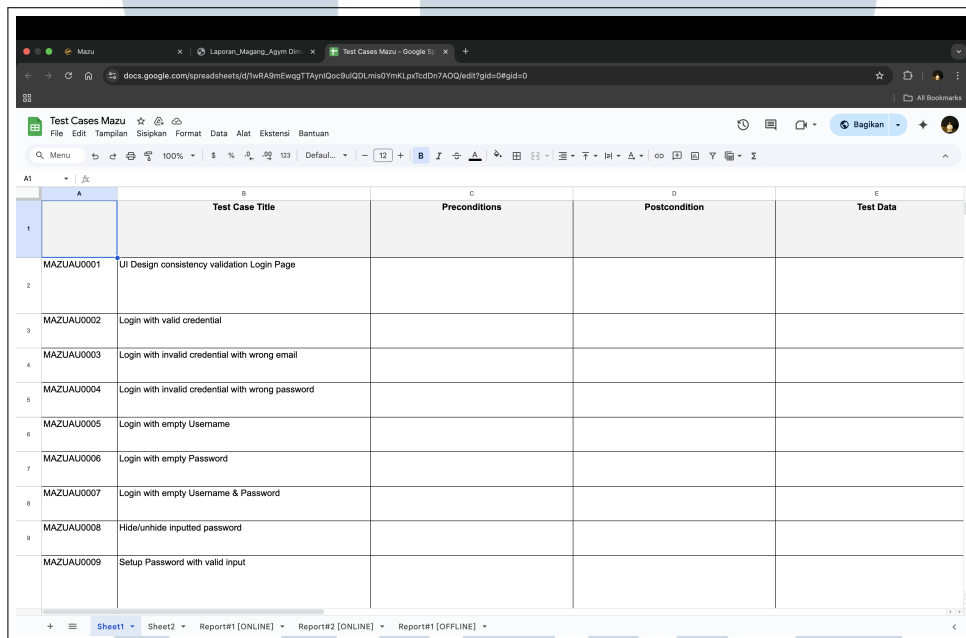
Pada perusahaan ini, Jira digunakan untuk mengatur *planning* dalam pelaksanaan *task* oleh tim. Tim nantinya akan diberikan *sprint planning* selama dua minggu sebagai bagian dari proses pengerjaan tugas dan proyek di perusahaan.

3.3.1 Flowchart



3.3.2 Tugas 1: Membuat Test Case untuk website MRO

Tugas pertama dalam pelaksanaan kerja magang di PT. Ganda Jaya Tama adalah membuat *test case* terlebih dahulu, yang bertujuan untuk memastikan apakah aplikasi atau *website* berjalan sesuai aturan atau tidak. Hal ini dilakukan untuk mengetahui bug yang mungkin muncul, serta memberikan masukan terhadap pengembangan sistem. Proses ini biasanya dilakukan sebelum *deployment*, agar dapat meminimalisir terjadinya *bug* atau hal-hal lain yang tidak sesuai dengan permintaan. Di dalam sebuah *test case* terdapat beberapa elemen penting, yaitu: *test case title*, *precondition*, *postcondition*, *test data*, *steps*, dan *actual result*. Berikut adalah gambaran *test case* yang dibuat pada *website* MRO.



	A	B	C	D	E
		Test Case Title	Preconditions	Postcondition	Test Data
1					
	MAZUAU0001	UI Design consistency validation Login Page			
2					
	MAZUAU0002	Login with valid credential			
3					
	MAZUAU0003	Login with invalid credential with wrong email			
4					
	MAZUAU0004	Login with invalid credential with wrong password			
5					
	MAZUAU0005	Login with empty Username			
6					
	MAZUAU0006	Login with empty Password			
7					
	MAZUAU0007	Login with empty Username & Password			
8					
	MAZUAU0008	Hide/unhide inputted password			
9					
	MAZUAU0009	Setup Password with valid input			

Gambar 3.6. Test Case

Gambar 3.6 Berikut adalah hasil *test case* yang telah dibuat, *test case* tersebut mengecek bagian *login page*

A	B	C	D	E
MAZUAU0010	Setup Password with invalid current password			
MAZUAU0011	Setup Password with invalid new password			
MAZUAU0012	Setup Password with invalid confirm password			

Gambar 3.7. Test case

Gambar 3.7 Berikut adalah hasil *test case* yang telah dibuat, *test case* tersebut mengecek dibagian *login page*

A	B	C	D	E
MAZUUS0006	Cancel Add new user			
MAZUUS0007	Edit user			
MAZUUS0008	Edit status user			
MAZUUS0009	Cancel Edit User			
MAZUUS0010	Delete User			
MAZUUS0011	Search User by username			

Gambar 3.8. Test case

Gambar 3.8 Berikut adalah hasil *test case* yang telah dibuat, *test case* tersebut mengecek dibagian *user page*

A	B	C	D	E
MAZUUS0012	Refactor Link usage on Edit Ikon			
MAZUUS0013	Refactor Link usage on Create New User Button			
MAZURL0001	UI Design consistency validation Role			
MAZURL0002	Add New Role			
MAZURL0003	Search User by Role			
MAZURL0004	Add new Role with not filling required fields			
MAZURL0005	Cancel Add new user			
MAZURL0006	Edit Role			

Gambar 3.9. Test case

Gambar 3.9 Berikut adalah hasil *test case* yang telah dibuat, *test case* tersebut mengecek dibagian *role page*

A	B	C	D	E
MAZUVR0002	Search Requisition by Number			VR/0000367/HANK/DV/2025
MAZUVR0003	View Option Setting			
MAZUVR0004	Filter Checkbox			
MAZUVR0005	View Document Detail			
MAZUVR0006	Print File PDF on the List Vessel Requisition			
MAZUVR0007	View Description on the List Vessel Requisition			
MAZUVR0008	View Remarks on the List Vessel Requisition			

Gambar 3.10. Test case

Gambar 3.10 Berikut adalah hasil *test case* yang telah dibuat, *test case* tersebut mengecek dibagian *vessel requisition page*

	A	B	C	D	E
45	MAZUVR0009	Print File PDF on the Detail Vessel Requisition			
46	MAZUVR0010	View Remarks on the Detail Vessel Requisition			
47	MAZUVR0011	Create Vessel Requisition Document [Master Engine]			
48	MAZUVR0012	Create Vessel Requisition with product remarks [Master Engine]			
49	MAZUVR0013	Edit Quantity on the Vessel Requisition Draft [Master Engine]			

Gambar 3.11. Test case

Gambar 3.11 Berikut adalah hasil *test case* yang telah dibuat, *test case* tersebut mengecek dibagian *vessel requisition page*

	A	B	C	D	E
50	MAZUVR0014	Edit Quantity on the Vessel Requisition List [Technical Manager]			
51	MAZUVR0015	Edit Draft Vessel Requisition Document			
52	MAZUVR0016	On Edit Draft Vessel Requisition do Delete product			
53	MAZUVR0017	On Edit draft Vessel Requisition add product			

Gambar 3.12. Test case

Gambar 3.12 Berikut adalah hasil *test case* yang telah dibuat, *test case* tersebut mengecek dibagian *vessel requisition page*

A	B	C	D	E
MAZUVR0024	Reject on Vessel Requisition Document (Second Level Approval) [Technical Manager]			
MAZUVR0025	Vessel Requisition Approval with Product Addition (First Level Approval) [Technical Super]			
MAZUVR0026	Vessel Requisition Approval with Product Addition (Second Level Approval) [Technical Manager]			
MAZUVR0027	Vessel Requisition Approval with Product Reduction (First Level Approval) [Technical Super]			

Gambar 3.13. Test case

Gambar 3.13 Berikut adalah hasil *test case* yang telah dibuat, *test case* tersebut mengecek dibagian *vessel requisition page*

A	B	C	D	E
MAZUVR0028	Vessel Requisition Approval with Product Reduction (Second Level Approval) [Technical Manager]			
MAZUVR0029	Vessel Requisition Approval with Product Editing (First Level Approval) [Technical Super]			
MAZUVR0030	Vessel Requisition Approval with Product Editing (Second Level Approval) [Technical Manager]			

Gambar 3.14. Test case

Gambar 3.14 Berikut adalah hasil *test case* yang telah dibuat, *test case* tersebut mengecek dibagian *vessel requisition page*

A	B	C	D	E
67	MAZUVR0031	Vessel Requisition Approval with product edit and document date update [Technical Manager]		
68	MAZUVR0032	Approval of Vessel Requisition Documents with no changes to the product (First Level Approval) [Technical Super]		
69	MAZUVR0033	Approval on Vessel Requisition Document that has no product, then added product (First Level Approval) [Technical Super]		

Gambar 3.15. Test case

Gambar 3.15 Berikut adalah hasil *test case* yang telah dibuat, *test case* tersebut mengecek dibagian *vessel requisition page*

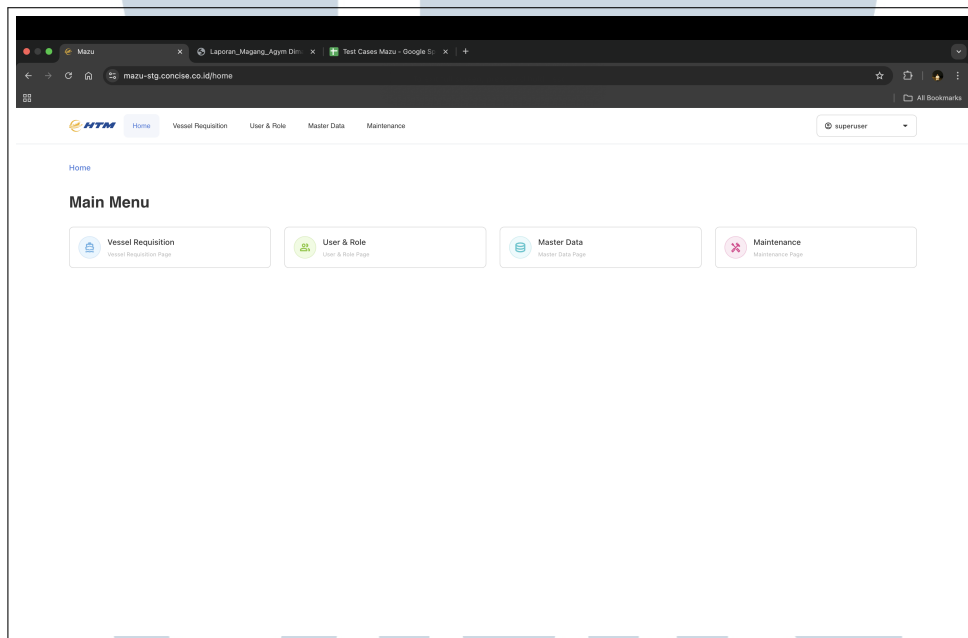
A	B	C	D	E
70	MAZUVR0034	Approval of the Vessel Requisition Document but there are changes to the quantity and remarks (First Level Approval) [Technical Super]		
71	MAZUVR0035	Approval on Vessel Requisition Document product that previously existed is then Deleted (First Level Approval) [Technical Super]		
72	MAZUVR0036	Create Vessel Requisition Document and then added 200 products [Master Engine]		

Gambar 3.16. Test case

Gambar 3.16 Berikut adalah hasil *test case* yang telah dibuat, *test case* tersebut mengecek dibagian *vessel requisition page*

3.3.3 Tugas 2: Membuat *Code* untuk Melakukan *Automation Website MRO*

Setelah selesai membuat *test case*, untuk tugas , pihak kantor memberikan tugas lanjutan, yaitu membuat *automation* pada *website MRO*. Sebelumnya, sudah diajarkan bagaimana cara menggunakan Playwright serta pengenalan bahasa *JavaScript* dan *TypeScript*. dibimbing langsung oleh dua *Junior Quality Assurance*, yaitu Bapak Jovan Haliem dan Bapak Agym Dimas Tian. Untuk penugasan awal yang diberikan, pihak kantor meminta membuat *automation* untuk *login page*, kemudian dilanjutkan dengan *User n Role*, *Master Data*, dan yang terakhir adalah *Vessel Requisition*. Untuk setiap *module*, terdapat banyak *fitur-fitur* yang perlu diuji. Berikut merupakan gambaran *website* serta potongan *code* dari *website MRO*.



Gambar 3.17. Tampilan Home Page pada website MRO

Gambar 3.17 Tampilan awal beranda utama website setelah login.

No	Vessel	Requisition No.	Requisition Date	Account Code	Requisition Description	Type	Status	Approval	Remarks	Action
1792	HANK - MV. HABCO ANKAA	VR0000430HANK/DV1/2025	02/06/2025, 14:49	45189 - General Stores - Catalogues (SN: HANK)	View Description	Store Order	In Review			View
1791	HANK - MV. HABCO ANKAA	VR0000429HANK/DV1/2025	02/06/2025, 14:49	45189 - General Stores - Catalogues (SN: HANK)	View Description	Store Order	In Review			View
1790	HANK - MV. HABCO ANKAA	VR0000428HANK/DV1/2025	01/01/2025, 07:00	45189 - General Stores - Catalogues (SN: HANK)	View Description	Store Order	Quotation			View
1789	HANK - MV. HABCO ANKAA	VR0000427HANK/DV1/2025	02/06/2025, 13:14	45189 - General Stores - Catalogues (SN: HANK)	View Description	Store Order	In Review			View
1788	HANK - MV. HABCO ANKAA	VR0000426HANK/DV1/2025	02/06/2025, 13:14	45189 - General Stores - Catalogues (SN: HANK)	View Description	Store Order	In Review			View
1787	HANK - MV. HABCO ANKAA	VR0000425HANK/DV1/2025	01/01/2025, 07:00	45189 - General Stores - Catalogues (SN: HANK)	View Description	Store Order	Quotation			View
	HANK - MV. HABCO ANKAA	VR0000424HANK/DV1/2025	01/01/2025,	45189 - General Stores - Catalogues	View	Store Order	Quotation			View

Gambar 3.18. Tampilan Vessel Requisition pada website MRO

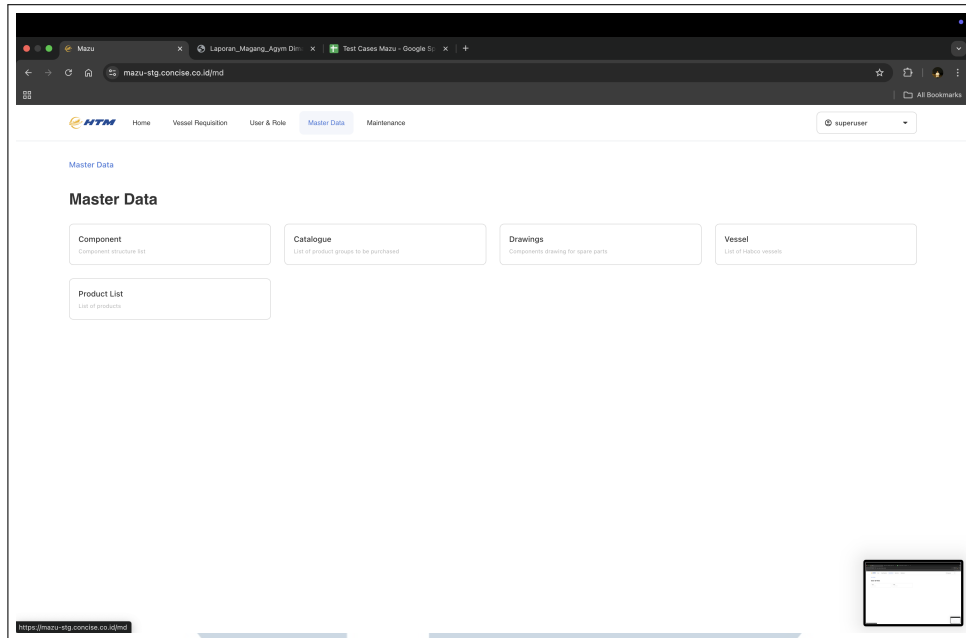
Gambar 3.18 Merupakan tampilan Vessel Requisition, yang berisikan pemesanan perlengkapan yang dibutuhkan kapal

User	Role
List of all users	List of all roles

Gambar 3.19. Tampilan User n Role pada website MRO

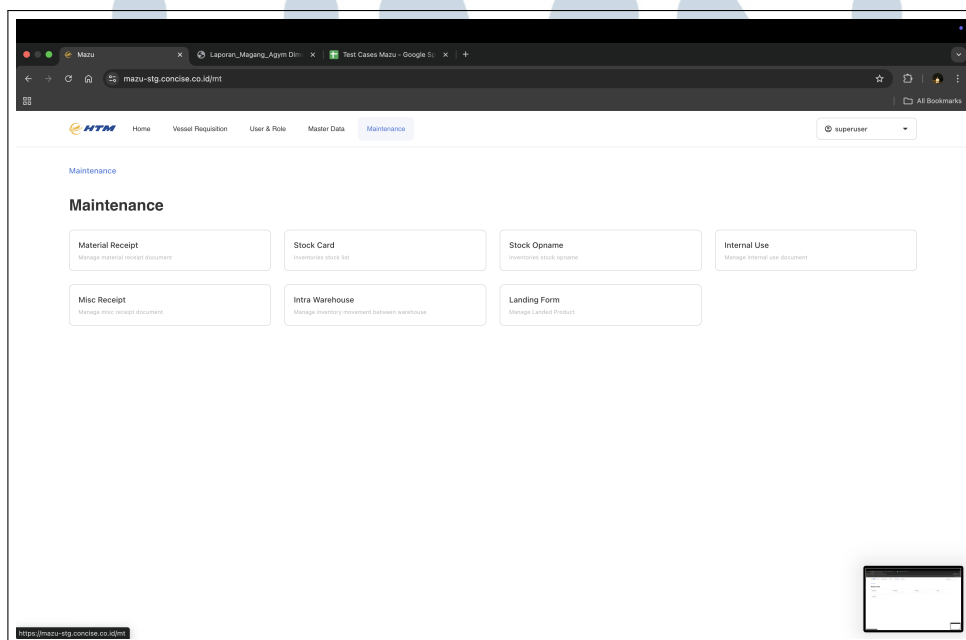
Gambar 3.19 Merupakan tampilan User n Role, yang memiliki 2 tujuan dari setiap fitur. User bertujuan untuk membuat user baru saja, jika role bertujuan untuk

memberikan role ke setiap user.



Gambar 3.20. Tampilan Master Data pada website MRO

Gambar 3.20 Merupakan tampilan Master Data, yang berisikan beberapa fitur, yaitu component, catalogue, drawings, vessel, dan product list. Di setiap fitur memiliki fungsinya masing-masing.



Gambar 3.21. Tampilan Maintenance pada website MRO

Gambar 3.21 merupakan tampilan Maintenance, yang berisikan beberapa fitur yaitu, material receipt, stock card, stock opname, internal use, misc receipt, intra warehouse, dan landing form. masing-masing dari fitur memiliki fungsinya sendiri, salah satunya ada untuk mengecek ketersediaan barang kapan, serta dapat juga memesan barang yang diperlukan.

```

1 import { Page, Locator } from '@playwright/test'
2 import dotenv from 'dotenv'
3 import { BasePage } from '../basePage'
4
5 // Load environment variables
6 dotenv.config()
7
8 export class LoginPage extends BasePage {
9   private readonly usernameField: Locator
10  private readonly passwordField: Locator
11  private readonly loginButton: Locator
12  // private readonly messageLocator: Locator
13  private readonly currentPasswordField: Locator
14  private readonly newPasswordField: Locator
15  private readonly confirmPasswordField: Locator
16
17  constructor(public readonly page: Page) {
18    super(page)
19    this.usernameField = page.getByPlaceholder('Your username')
20    this.passwordField = page.getByPlaceholder('Your Password')
21    this.loginButton = page.locator("[data-testid='login_btn']")
22    // this.messageLocator = page.locator('.toastMessages')
23    this.currentPasswordField = page.locator('#currpassword')
24    this.newPasswordField = page.locator('#newpassword')
25    this.confirmPasswordField = page.locator('#confirmpassword')
26  }
27 }

```

Kode 3.1: Kode TypeScript untuk Halaman Login Playwright

```

1 import type { Locator, Page, expect } from '@playwright/test'
2 import { randomNumber, randomString } from '../utils/
  randomGenerator'
3 import { BasePage } from '../basePage'
4
5 export class VesselPage extends BasePage {
6   locator(arg0: string) {
7     throw new Error('Method not implemented.')

```

```

8   }
9   private tempDescription: string
10
11  constructor(page: Page) {
12      super(page)
13  }
14
15  async navigateMenu() {
16      await this.page.getByRole('link', { name: 'Vessel Requisition'
17      }).first().click()
18  }
19
20  async waitLoaderAndHeader() {
21      await this.waitForLoaderAndHeader(/Requisition Summary/)
22  }
23
24  async addNewRequisitionBtn(desc: string) {
25      await this.page.waitForTimeout(3000)
26      await this.page.getByRole('button', { name: 'Create New
27      Requisition' }).click()
28
29      // Requisition Type Field
30      await this.page.locator('#Input_RequisitionType').click()
31      await this.page.getByTestId('Input_RequisitionType_Option_1').
32      click()
33
34      // Requisition Account Code
35      await this.page.locator('#Input_AccountCode').click()
36      await this.page.getByTestId('Input_AccountCode_Option_2').
37      click();
38
39      // Requisition Category
40      await this.page.locator('#Input_Category').click()
41      await this.page.getByTestId('Input_Category_Option_2').click()
42
43      // Description
44      await this.page.getByTestId('Input_RequisitionDescription').
45      fill(desc);
46
47      await this.page.getByTestId('PartsItems_Button_AddPartsItems')
48      .click();
49
50      // First Locator

```

```

45     await this.page.locator('#ModalPartsItems_Input_FirstLocator')
      .click()
46     await this.page.getByTestId('
ModalPartsItems_Input_FirstLocator_Option_1').click()
47
48     // Second Locator
49     await this.page.locator('#ModalPartsItems_Input_SecondLocator
').click()
50     await this.page.getByTestId('
ModalPartsItems_Input_SecondLocator_Option_1').click()
51
52     // Catalogue
53     await this.page.locator('#ModalPartsItems_Input_Catalogue').
      click()
54     await this.page.getByTestId('
ModalPartsItems_Input_Catalogue_Option_1').click()
55   }
56 }

```

Kode 3.2: Kode TypeScript untuk Halaman Vessel Requisition

```

1 import type { Locator, Page } from '@playwright/test'
2 import { randomNumber, randomString } from '../utils/
  randomGenerator'
3 import { BasePage } from './basePage'
4
5 export class UserPage extends BasePage {
6   private tempUserName: string
7
8   constructor(page: Page) {
9     super(page)
10  }
11
12 }

```

Kode 3.3: Kode TypeScript untuk Halaman UserPage

```

1 import type { Locator, Page } from '@playwright/test'
2 import { randomString } from '../utils/randomGenerator'
3 import { BasePage } from './basePage'
4
5 export class RolePage extends BasePage {
6   private tempRoleName: string
7
8   constructor(page: Page) {

```

```

9      super(page)
10   }
11
12   async navigateMenu() {
13       await this.page.getByRole('link', { name: 'Role' }).first().
14       click()
15   }
16
17   async waitLoaderAndHeader() {
18       await this.waitForLoaderAndHeader(/Role/)
19   }
20
21   async addCreateNewRole(name: string) {
22       this.addNewRoleBtn(name)
23       await this.page.getByRole('button', { name: 'Save Role' }).
24       click()
25
26       await this.page.waitForLoadState('networkidle')
27       await this.page.getByTestId('primary_btn').waitFor()
28       await this.page.getByTestId('primary_btn').click()
29   }
30
31   async generateTempRoleName(length: number = 3) {
32       this.tempRoleName = await randomString(length)
33       return this.tempRoleName
34   }
35
36   async getTempRoleName() {
37       return this.tempRoleName
38   }
39
40   async addNewRoleBtn(name: string) {
41       await this.page.getByRole('button', { name: 'Add New Role' }).
42       click()
43       await this.page.getByPlaceholder('Input Role Name').fill(name)
44   }
45
46   async removeRoleBtn(filter: Locator) {
47       const rowLocator = this.page.locator('.roles')
48       .getByRole('listitem')
49       .filter({ has: filter })
50       .locator('.trash_btn')

```

```

49     await rowLocator.waitFor()
50     await rowLocator.click()
51 }
52 }

```

Kode 3.4: Kode TypeScript untuk Halaman RolePage

```

1 import type { Locator, Page, expect } from '@playwright/test'
2 import { randomNumber, randomString } from '../utils/
  randomGenerator'
3 import { BasePage } from './basePage'
4
5 export class CataloguePage extends BasePage {
6   private tempCatalogueName: string
7   tempDescription: string
8
9   constructor(page: Page) {
10     super(page)
11   }
12
13   async navigateMenu() {
14     await this.page.getByRole('link', { name: 'Master Data' }).
      first().click()
15     await this.page.locator('.mainMenu').getByText('Catalogue').
      first().click()
16   }
17
18   async addCatalogueBtn(name: string) {
19     await this.page.getByTestId(name).click();
20   }
21
22   async waitLoaderAndHeader() {
23     await this.waitForLoaderAndHeader(/Catalogue/)
24   }
25
26   async createNewCatalogue() {
27     await this.page.getByRole('button', { name: 'Create New
      Catalogue' }).click();
28   }
29
30   async fillCatalogueName(name: string) {
31     await this.page.getByText('New Catalogue').waitFor()
32     await this.page.getByTestId('Input_CatalogueName').waitFor()
33     const inputField = this.page.getByTestId('Input_CatalogueName

```

```

    ');
34     await inputField.fill(name);
35     await inputField.click();
36     await inputField.click();
37 }
38
39 async addProductToCatalogue() {
40     await this.page.getByRole('button', { name: 'Add Product to
    Catalogue' }).click();
41
42     await this.page.getByTestId('ModalProduct_CheckBox_0').click()
    ;
43     await this.page.getByTestId('addProd_btn').click();
44 }
45
46 async editCatalogueBtn(name: string) {
47     await this.page.getByTestId('Button_Edit_0').click();
48     await this.page.getByTestId('Input_CatalogueName').fill(name);
49     await this.page.getByTestId('Button_Save').click();
50     await this.page.getByTestId('ModalConfirmation_Button_Primary
    ').click();
51 }
52 }

```

Kode 3.5: Kode TypeScript untuk Halaman CataloguePage

```

1 import type { Locator, Page, expect } from '@playwright/test'
2 import { randomNumber, randomString } from '../utils/
    randomGenerator'
3 import { BasePage } from './basePage'
4
5 export class DrawingsPage extends BasePage {
6     private tempDrawingsName: string
7
8     constructor(page: Page) {
9         super(page)
10    }
11 }

```

Kode 3.6: Kode TypeScript untuk Halaman DrawingsPage

```

1 import type { Locator, Page, expect } from '@playwright/test'
2 import { randomNumber, randomString } from '../utils/
    randomGenerator'
3 import { BasePage } from './basePage'

```

```

4
5 export class VesselMDPage extends BasePage {
6   private tempVesselMDName: string
7
8   constructor(page: Page) {
9     super(page)
10  }
11 }

```

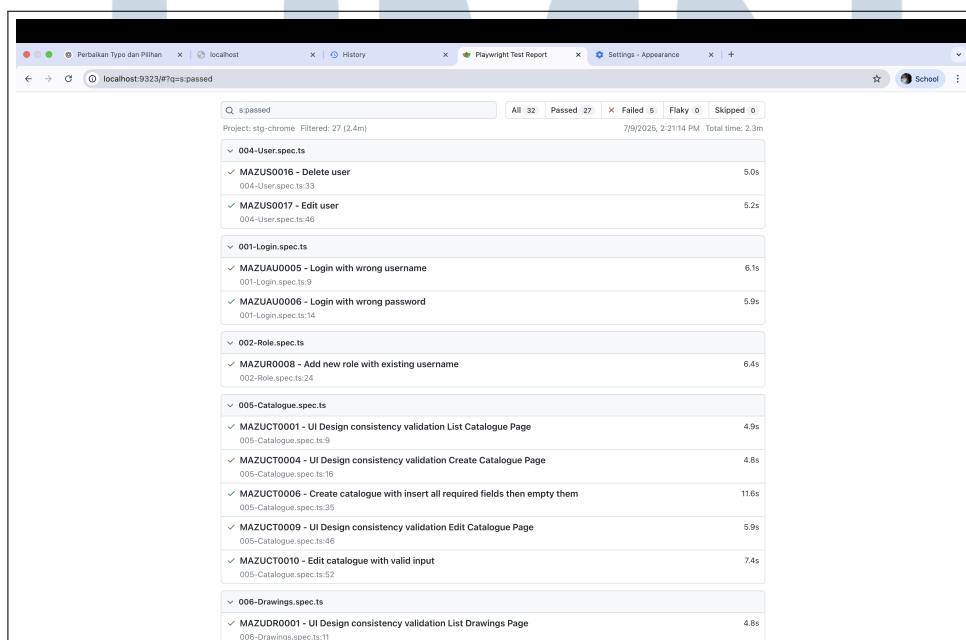
Kode 3.7: Kode TypeScript untuk Halaman VesselMDPage

```

1 import type { Locator, Page, expect } from '@playwright/test'
2 import { randomNumber, randomString } from '../utils/
   randomGenerator'
3 import { BasePage } from '../basePage'
4
5 export class ProductListPage extends BasePage {
6   private tempProductListName: string
7
8   constructor(page: Page) {
9     super(page)
10  }
11 }

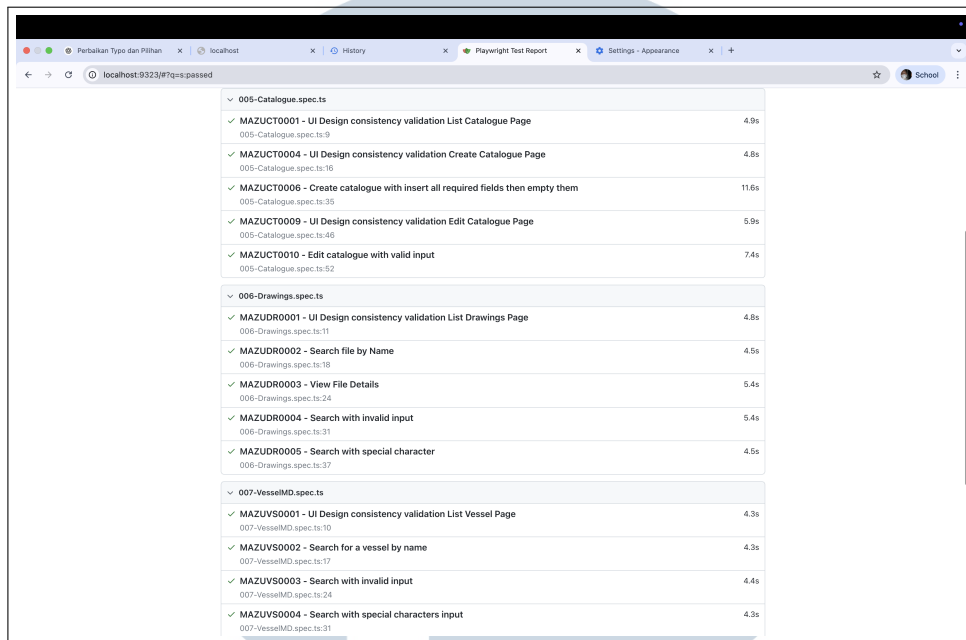
```

Kode 3.8: Kode TypeScript untuk Halaman ProductListPage



Gambar 3.22. Hasil Report setelah melakukan testing secara automation

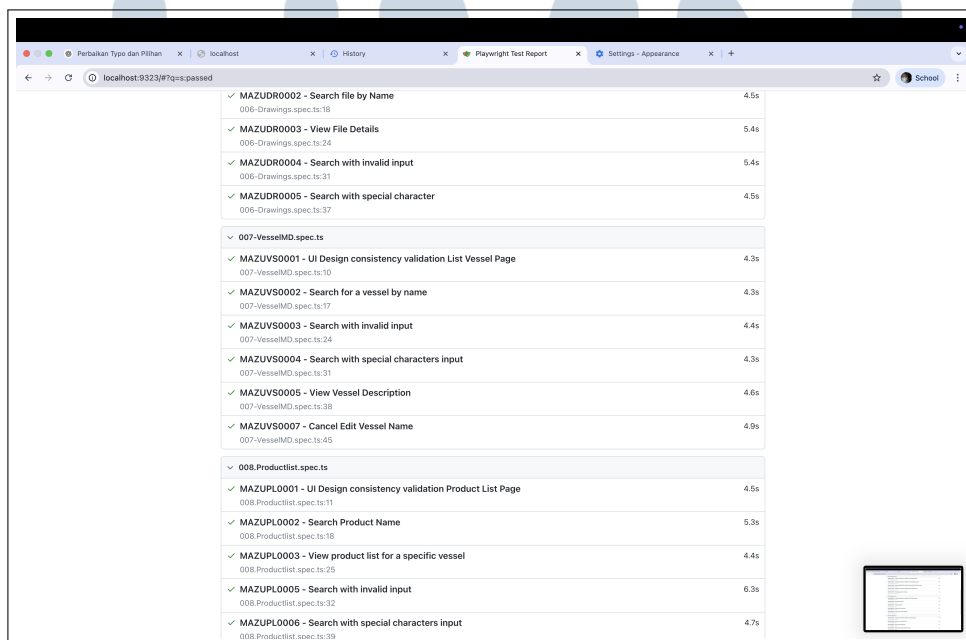
Gambar 3.22 Menunjukkan bahwa *testing* yang dilakukan dijalankan lancar, dan tidak memiliki *failed*



Test Name	Duration
005-Catalogue.spec.ts	
✓ MAZUCT0001 - UI Design consistency validation List Catalogue Page	4.9s
005-Catalogue.spec.ts:9	
✓ MAZUCT0004 - UI Design consistency validation Create Catalogue Page	4.8s
005-Catalogue.spec.ts:16	
✓ MAZUCT0006 - Create catalogue with insert all required fields then empty them	11.6s
005-Catalogue.spec.ts:35	
✓ MAZUCT0009 - UI Design consistency validation Edit Catalogue Page	5.9s
005-Catalogue.spec.ts:46	
✓ MAZUCT0010 - Edit catalogue with valid input	7.4s
005-Catalogue.spec.ts:52	
006-Drawings.spec.ts	
✓ MAZUDR0001 - UI Design consistency validation List Drawings Page	4.8s
006-Drawings.spec.ts:11	
✓ MAZUDR0002 - Search file by Name	4.5s
006-Drawings.spec.ts:18	
✓ MAZUDR0003 - View File Details	5.4s
006-Drawings.spec.ts:24	
✓ MAZUDR0004 - Search with invalid input	5.4s
006-Drawings.spec.ts:31	
✓ MAZUDR0005 - Search with special character	4.5s
006-Drawings.spec.ts:37	
007-VesselMD.spec.ts	
✓ MAZUVS0001 - UI Design consistency validation List Vessel Page	4.3s
007-VesselMD.spec.ts:10	
✓ MAZUVS0002 - Search for a vessel by name	4.3s
007-VesselMD.spec.ts:17	
✓ MAZUVS0003 - Search with invalid input	4.4s
007-VesselMD.spec.ts:24	
✓ MAZUVS0004 - Search with special characters input	4.3s
007-VesselMD.spec.ts:31	

Gambar 3.23. Hasil Report setelah melakukan testing secara automation

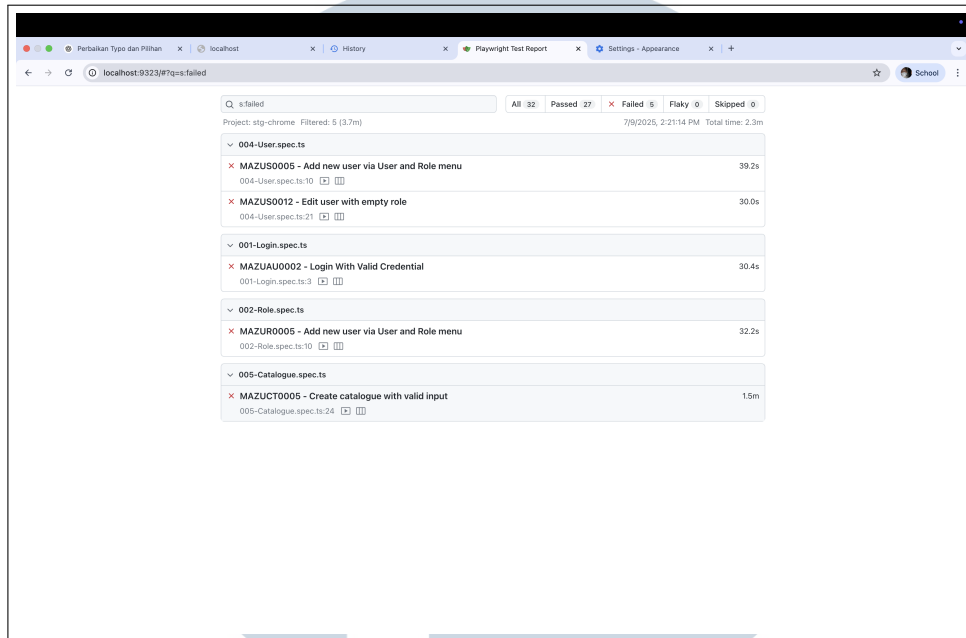
Gambar 3.23 Menunjukkan bahwa *testing* yang dilakukan dijalankan lancar, dan tidak memiliki *failed*



Test Name	Duration
✓ MAZUDR0002 - Search file by Name	4.5s
006-Drawings.spec.ts:18	
✓ MAZUDR0003 - View File Details	5.4s
006-Drawings.spec.ts:24	
✓ MAZUDR0004 - Search with invalid input	5.4s
006-Drawings.spec.ts:31	
✓ MAZUDR0005 - Search with special character	4.5s
006-Drawings.spec.ts:37	
007-VesselMD.spec.ts	
✓ MAZUVS0001 - UI Design consistency validation List Vessel Page	4.3s
007-VesselMD.spec.ts:10	
✓ MAZUVS0002 - Search for a vessel by name	4.3s
007-VesselMD.spec.ts:17	
✓ MAZUVS0003 - Search with invalid input	4.4s
007-VesselMD.spec.ts:24	
✓ MAZUVS0004 - Search with special characters input	4.3s
007-VesselMD.spec.ts:31	
✓ MAZUVS0005 - View Vessel Description	4.6s
007-VesselMD.spec.ts:38	
✓ MAZUVS0007 - Cancel Edit Vessel Name	4.9s
007-VesselMD.spec.ts:45	
008-Productlist.spec.ts	
✓ MAZUPL0001 - UI Design consistency validation Product List Page	4.5s
008-Productlist.spec.ts:11	
✓ MAZUPL0002 - Search Product Name	5.3s
008-Productlist.spec.ts:18	
✓ MAZUPL0003 - View product list for a specific vessel	4.4s
008-Productlist.spec.ts:25	
✓ MAZUPL0005 - Search with invalid input	6.3s
008-Productlist.spec.ts:32	
✓ MAZUPL0006 - Search with special characters input	4.7s
008-Productlist.spec.ts:39	

Gambar 3.24. Hasil Report setelah melakukan testing secara automation

Gambar 3.24 Menunjukkan bahwa *testing* yang dilakukan dijalankan lancar, dan tidak memiliki *failed*



Gambar 3.25. Hasil Report setelah melakukan testing secara automation

Gambar 3.25 Menunjukkan bahwa *testing* yang dilakukan dijalankan tidak lancar, dan memiliki *failed*

3.4 Kendala dan Solusi yang Ditemukan

3.4.1 Kendala

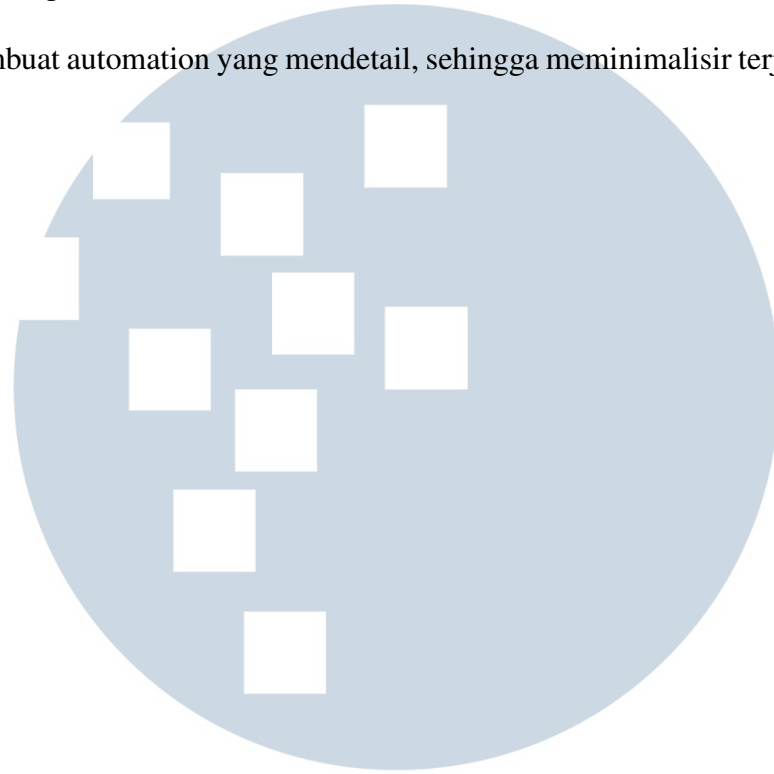
Selama proses pengerjaan website dan aplikasi di PT. Ganda Jaya Tama, terdapat beberapa kendala yang dihadapi, yaitu

- Dalam code automation, masih menggunakan bahasa pemrograman Javascript, sedangkan bahasa yang sekarang digunakan TypeScript.
- Masih terdapat hasil test yang kurang mendetail berdasarkan dari code Javascript sebelumnya

3.4.2 Solusi

Beberapa solusi yang diterapkan untuk mengatasi kendala-kendala tersebut adalah:

- Mengintegrasikan website tersebut menggunakan bahasa pemrograman Typescript
- Membuat automation yang mendetail, sehingga meminimalisir terjadinya bug



UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA