RANCANG BANGUN SISTEM PENGELOLAAN PERMINTAAN DARAH DARURAT (DONORA) DENGAN FRAMEWORK LARAVEL DAN TEKNOLOGI REST API

Zabina Anastasya Nurhaliza, Eka Dyar Wahyuni
Sistem Informasi, Universitas Pembangunan Nasional "Veteran" Jawa Timur
Jl. Rungkut Madya No.1, Gn. Anyar, Kec. Gn. Anyar, Surabaya, Jawa Timur
20082010104@student.upnjatim.ac.id

ABSTRAK

Masalah kurangnya pasokan darah di Indonesia mempengaruhi pelayanan kesehatan, terutama dalam penanganan transfusi darah mendesak. Penelitian ini bertujuan untuk mengembangkan sistem pengelolaan kebutuhan darah darurat yang efisien, Donora. Donora dirancang untuk mengatasi masalah ini dengan mengirimkan notifikasi cepat kepada pengguna untuk merespons kebutuhan donor darah mendesak. Metode pengembangan menggunakan kerangka kerja Scrum, yang mencakup tahapan Product Backlog, Sprint Planning, Sprint Execution, Daily Scrum, Sprint Review, dan Sprint Retrospective. Penggunaan teknologi seperti Laravel, Object Relational Mapping, dan Firebase Cloud Messaging memperkuat fungsionalitas dan responsivitas sistem. Selain itu terdapat pengujian perangkat lunak secara black box menggunakan metode manual testing. Hasil akhir dari evaluasi didapatkan bahwa sistem donora telah lolos pada seluruh tahap pengujian yang berarti sistem telah berjalan dengan baik.

Kata kunci : *sistem pengelolaan, darah darurat, notifikasi cepat, UTD PMI, Laravel, Firebase Cloud Messaging*

1. PENDAHULUAN

Berdasarkan informasi dari Ditjen Pelayanan Kesehatan, Kementerian Kesehatan RI, dan standar WHO, setiap negara diharapkan memiliki persediaan darah minimal 2% dari total penduduknya [1]. Namun, Indonesia masih jauh dari target ini, hanya sekitar 77.438 kantong darah tercatat di UTD PMI hingga Juni 2023 [2]. Masalah ini disebabkan oleh beberapa faktor, termasuk kadaluarsa kantong darah, keterbatasan teknologi, dan kurangnya kesadaran masyarakat tentang pentingnya donor darah. Ketersediaan darah yang terbatas mengakibatkan penurunan kualitas layanan kesehatan, terutama dalam penanganan transfusi darah mendesak.

Bank darah rumah sakit juga terpengaruh oleh keterbatasan pasokan dari PMI, mengakibatkan kekosongan stok dan memaksa pasien mencari pendonor secara mandiri. Proses ini seringkali lambat dan tidak efisien, meningkatkan risiko bagi keselamatan pasien, terutama bagi mereka yang membutuhkan darah dengan cepat.

Berdasarkan penelitian terdahulu, penerapan sistem informasi manajemen layanan donor darah dapat memudahkan dan mempercepat masyarakat dalam mendapatkan informasi tentang donor darah yang dibutuhkan karena informasi tersebut dapat diakses kapan saja. Fitur-fitur yang disediakan, seperti menu pemesanan dan stok darah memberikan manfaat untuk meningkatkan proses donor darah dan mengurangi risiko pemborosan darah akibat kadaluarsa [3].

Sebagai inovasi, penelitian ini akan menciptakan Donora, sebuah sistem pengelolaan kebutuhan darah darurat yang mengirimkan notifikasi secara cepat kepada pengguna untuk merespons kebutuhan donor darah mendesak.

Donora akan dikembangkan secara bertahap dengan memanfaatkan API (Application Programming Interface) untuk memudahkan kolaborasi dengan sistem lain dan memastikan interoperabilitas data serta informasi di berbagai platform. Penggunaan API memungkinkan aplikasi untuk terus berkembang seiring waktu dengan fleksibilitas menambahkan atau menghapus layanan sesuai dengan kebutuhan pengguna. Dengan demikian, Donora dapat terus berkembang sesuai dengan permintaan dan tuntutan pengguna [4].

Selama proses pengembangan Donora, pendekatan SCRUM diterapkan sebagai metode perancangan dalam Software Development Life Cycle (SDLC). SCRUM dipilih karena kemampuannya menghasilkan perangkat lunak berkualitas sesuai kebutuhan pengguna, serta kesesuaian penggunaan pada proyek skala besar maupun kecil, dan fleksibilitasnya dalam menghadapi perubahan, termasuk dalam perancangan aplikasi [5]. Dalam lingkungan industri yang dinamis, Metode SCRUM juga dinilai sebagai metode yang lebih modern dalam lingkungan industri yang dinamis, dan mampu memberikan kendali kualitas yang optimal melalui pengujian fungsionalitas hasilnya [6].

2. TINJAUAN PUSTAKA

2.1. Object Relational Mapping

ORM atau Object Relational Mapping adalah teknologi penghubung antara aplikasi dan database yang bekerja dengan mengubah data dalam database menjadi bentuk objek dalam aplikasi. Dengan ORM, pengembangan aplikasi menjadi lebih efisien dan

mudah dilakukan karena menyederhanakan proses pengaksesan dan pengelolaan data dalam database melalui objek-objek yang mewakili tabel-tabel dalam database. Hal ini akan berakibat pada penghematan waktu, peningkatan performa, dan pengurangan tantangan arsitektur yang terkait dengan penggunaan ORM [7].

2.2. RESTful API

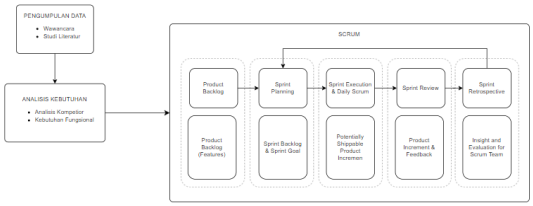
REST (Representational State Transfer) adalah model arsitektur untuk sistem hypermedia terdistribusi yang diperkenalkan oleh Roy Fielding pada tahun 2000 [7]. Sederhananya, REST merupakan salah satu desain arsitektur yang ada dalam API. Semantara RESTful API merupakan merupakan sebuah implementasi dari arsitektur REST yang menyediakan antarmuka pemrograman yang efisien dan seragam bagi aplikasi klien.

2.3. Firebase Cloud Messaging (FCM)

Firebase Cloud Messaging (FCM) adalah layanan yang disediakan oleh Firebase untuk mendukung pengembangan aplikasi mobile dan web. FCM menyediakan dua jenis pesan: notifikasi, yang digunakan untuk menampilkan informasi kepada pengguna melalui bilah notifikasi visual, dan pesan data, yang digunakan untuk mengirim informasi tambahan pada aplikasi yang sedang berjalan, termasuk kemungkinan pengambilan tindakan berdasarkan data yang diterima. Penggunaan kedua layanan ini memungkinkan pengiriman push notification beserta informasi detail secara real-time kepada pengguna, bahkan saat aplikasi tidak dibuka [8].

3. METODE PENELITIAN

Metodologi penelitian merupakan langkah-langkah yang dilakukan penulis untuk mencapai tujuan penelitian ini.



Gambar 1. Metode Penelitian

3.1. Pengumpulan Data

Pengumpulan data dilakukan dengan melakukan studi literatur melalui pencarian referensi dalam bentuk jurnal penelitian dan buku secara online yang terkait dengan topik Rancang Bangun Sistem Pengelolaan Stok Darah atau Sistem Donor Darah, serta Metodologi Pengembangan Perangkat Lunak Scrum. Informasi yang relevan dan diperlukan dalam penelitian ini digunakan untuk menyusun tinjauan pustaka, merancang metodologi penelitian, dan mengembangkan sistem yang diinginkan.

3.2. Analisis Kebutuhan

Analisis kebutuhan dalam penelitian ini melibatkan pengolahan data dari pengumpulan informasi mengenai alur permintaan darah saat ini yang dilakukan secara konvensional dan ditemukan solusinya melalui Donora. Kemudian, dilakukan analisis kompetitor terhadap aplikasi serupa di Google Playstore untuk mendapatkan informasi yang relevan dan mengidentifikasi kelebihan serta kekurangan fitur aplikasi referensi yang ada. Hasil analisis kemudian digunakan untuk menentukan kebutuhan fungsional yang akan diimplementasikan dalam sistem Donora yang sedang dibangun.

3.3. Metode pengembangan aplikasi yang dikembangkan Scrum

Scrum digunakan sebagai kerangka kerja pengembangan sistem dalam penelitian ini karena menawarkan seperangkat nilai, prinsip, dan praktik yang terstruktur. Tahapan-tahapan dalam Scrum mencakup Product Backlog, Sprint Planning, Sprint Execution, Daily Scrum, Sprint Review, dan Sprint Retrospective [9].

4. HASIL DAN PEMBAHASAN

Hasil dari penelitian ini merupakan sistem Donora, yaitu sebuah sistem pengelolaan kebutuhan darah darurat yang mengirimkan notifikasi secara cepat kepada pengguna untuk merespons kebutuhan donor darah mendesak.

4.1. Analisis Kebutuhan

Saat pasien dinyatakan membutuhkan darah, dokter akan mengeluarkan formulir permintaan serta mengambil sampel darah. Sampel dan formulir kemudian dibawa ke Unit Transfusi Darah (UTD) PMI untuk pengujian dan analisis kesesuaian darah. Namun, saat ini, terdapat masalah terkait ketersediaan stok darah yang tidak terupdate secara berkala di beberapa website UTD PMI, menyebabkan pasien harus melakukan konfirmasi tambahan. Jika stok darah tidak tersedia, pasien harus menunggu pendonor atau mencari sendiri, yang dapat memakan waktu berharga dan meningkatkan risiko terhadap kesehatan pasien.

Untuk mengatasi masalah ini, Donora diusulkan sebagai sistem untuk mengelola permintaan darah darurat dengan fitur push-notification dan pelacakan stok darah secara real-time. Ini memungkinkan UTD PMI untuk memperbarui stok darah secara langsung, memungkinkan pengguna untuk melacak ketersediaan darah tanpa harus datang langsung ke lokasi UTD PMI. Ini diharapkan dapat mempercepat proses permintaan darah dan mengurangi risiko serta kerumitan dalam mendapatkan darah yang dibutuhkan pasien.

Dari masalah dan solusi yang telah dijabarkan, Donora akan dibangun menjadi sebuah sistem dengan

memiliki tiga level user, yaitu : admin, UTD PMI, dan user. Dengan *functional requirement* sebagai berikut :

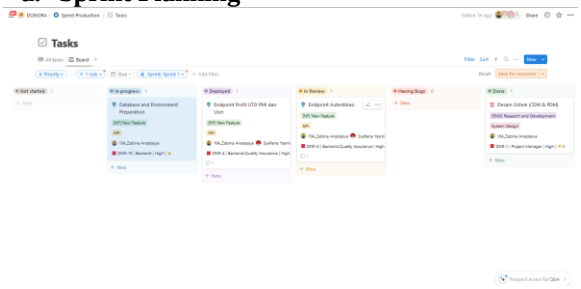
Tabel 1 Kebutuhan fungsional sistem Donora.

KODE	Kebutuhan Fungsional
FR-ADM-7	Admin dapat melihat ajuan darah darurat
FR-ADM-8	Admin dapat menerima ajuan darah darurat
FR-ADM-9	Admin dapat memproses ajuan darah darurat
FR-ADM-10	Admin dapat menyelesaikan ajuan darah darurat
FR-ADM-11	Admin dapat mengirimkan ulang push notification ajuan darah darurat
FR-ADM-12	Admin dapat menolak ajuan darah darurat
FR-ADM-13	Admin dapat menghapus ajuan darah darurat
FR-ADM-14	Admin dapat melihat list pendonor darurat
FR-ADM-15	Admin dapat mengupdate call status pendonor darurat
FR-UTD-8	UTD PMI dapat melihat pendonor darurat
FR-USER-10	User dapat membuat ajuan darah darurat
FR-USER-11	User dapat melihat ajuan darah darurat
FR-USER-12	User dapat menghapus ajuan darah darurat

4.2. Eksekusi Proyek dengan Scrum

4.2.1. Iterasi Sprint ke-1

a. Sprint Planning



Gambar 2. *Sprint board iterasi ke-1*

Dilakukan diskusi untuk menentukan sprint backlog atau kumpulan tugas yang akan dikembangkan pada iterasi ke-1, yaitu :

1. Membuat desain sistem untuk sistem donora
2. Melakukan inisialisasi dan persiapan database.
3. Membuat API endpoint Autentikasi
4. Membuat API profile UTD PMI dan User

b. Sprint Execution & Daily Scrum

Pada tahap ini, sprint backlog dieksekusi oleh tim dan dilakukan daily scrum agar tim memantau satu sama lain perkembangan proyek memonitor perkembangan proyek secara bersama-sama.

c. Sprint Review

Selama satu sprint tim menghasilkan desain sistem yang meliputi :

1. Use Case Diagram

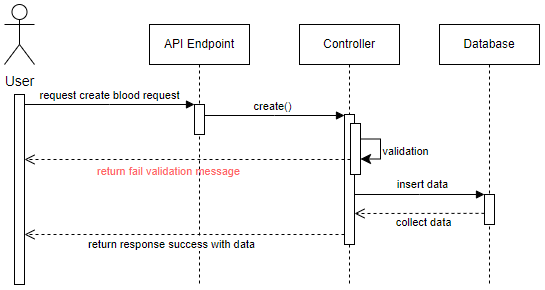
Merupakan pemodal untuk behavior sistem informasi yang akan dibuat agar mengetahui fungsi apa saja yang ada di dalam sistem informasi tersebut dan siapa saja yang berhak menggunakan fungsi tersebut.

Use case sistem Donora memiliki 3 aktor, yaitu user (pengguna), UTD PMI, dan Admin. Dengan case pengelolaan pada ajuan riwayat donor, stok darah, jadwal donor darah, akun dan profil, *push-notifikasi*, dan reward, yang telah disesuaikan sesuai dengan para aktor.

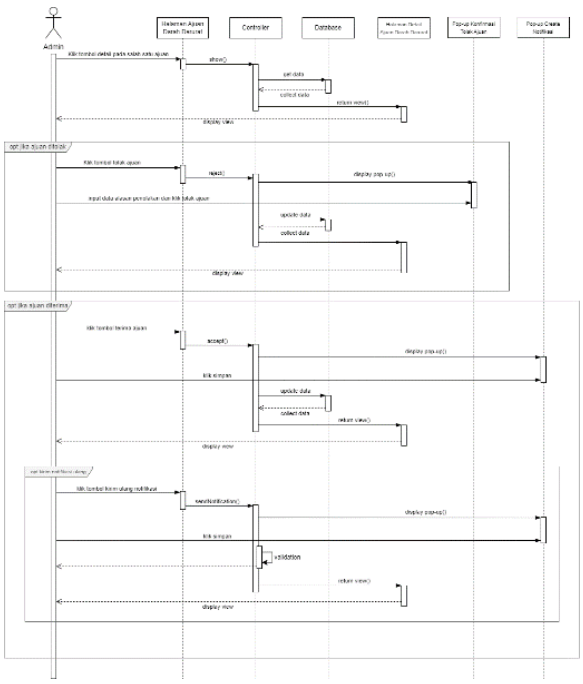
2. Sequence Diagram

Sequence Diagram menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek.

Gambar 3 menampilkan alokasi function yang digunakan untuk menjalankan use case create ajuan darah darurat pada aktor user.



Gambar 3. Sequence diagram create ajuan permintaan darah darurat oleh user



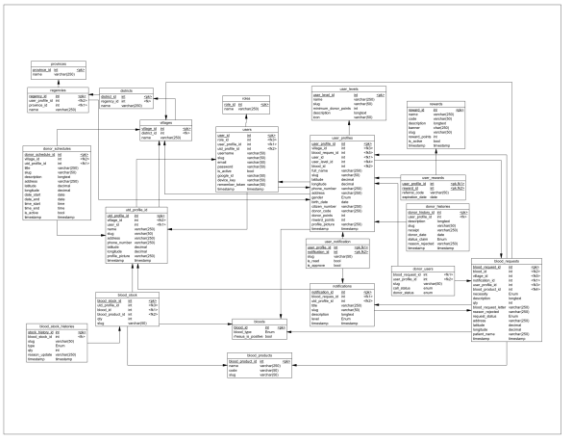
Gambar 4. Sequence diagram kelola ajuan darah darurat user oleh admin

Gambar 4 menampilkan alokasi function yang digunakan untuk menjalankan use case lihat ajuan darah darurat oleh aktor admin. Adapun *option* yang dapat dilakukan oleh admin, yaitu

melok ajuan atau menerima ajuan. Pada terima ajuan, admin memiliki alternatif untuk melakukan *push-notifikasi* ulang.

3. Physical Data Model (PDM)

Menggambarkan relasi antar entitas dan tabel dengan foreign key dalam sistem yang akan dibuat.



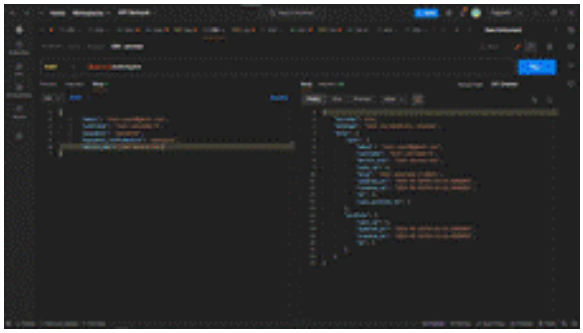
Gambar 5. Physical data model sistem Donora

Di samping itu, dalam iterasi pertama juga dilakukan persiapan database dengan tujuan menginisialisasi struktur database sesuai dengan desain sistem menggunakan migrasi yang telah tersedia di Laravel. Dibawah ini, merupakan contoh dari *function* migrasi untuk tabel blood_request yang akan menampung data ajuan permintaan darah yang dibuat oleh user.

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateBloodRequestsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('blood_requests', function (Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('user_profile_id');
            $table->unsignedBigInteger('blood_id');
            $table->unsignedBigInteger('blood_product_id');
            $table->unsignedBigInteger('village_id');
            $table->string('patient_name')->nullable();
            $table->string('slug')->nullable();
            $table->enum('necessity', ['accident', 'illness', 'surgery', 'chronic condition', 'other'])->default('illness');
            $table->longText('description')->nullable();
            $table->integer('qty')->nullable();
            $table->string('blood_request_letter')->nullable();
            $table->string('reason_rejected')->nullable();
            $table->enum('request_status', ['pending', 'accepted', 'in process', 'rejected', 'finish'])->default('pending');
            $table->longText('address');
            $table->decimal('latitude', 10, 8)->nullable();
            $table->decimal('longitude', 11, 8)->nullable();
            $table->timestamps();

            $table->foreign('user_profile_id')->references('id')->on('user_profiles')->onDelete('cascade');
            $table->foreign('blood_id')->references('id')->on('bloods')->onDelete('cascade');
            $table->foreign('blood_product_id')->references('id')->on('blood_products')->onDelete('cascade');
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('blood_requests');
    }
}
```

Gambar 6. Potongan kode migrasi tabel blood_request

Setelah database dan *environment* lain telah siap, dibuat juga API Endpoint untuk autentikasi, profile UTD PMI dan User yang harus ada sebelum fitur utama, ajuan permintaan darah darurat dapat dibuat.



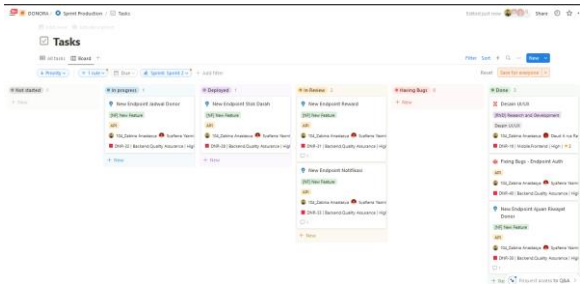
Gambar 7. Endpoint dan response API ajuan permintaan darah darurat

d. Sprint Retrospective

Dilakukan diskusi *sprint retrospective* pada akhir dari iterasi sprit ke-1. Dengan hasil yaitu, performa tim dalam mengerjakan task dinilai baik karena telah mencapai target dengan menyelesaikan backlog tepat waktu

4.2.2. Iterasi Sprint ke-2

a. Sprint Planning



Gambar 8. Sprint board iterasi ke-2

Dilakukan diskusi untuk menentukan sprint backlog atau kumpulan tugas yang akan dikembangkan pada iterasi ke-2, yaitu

1. Desain antarmuka (UI/UX) Dashboard Admin dan Dashboard UTD PMI
2. Membuat API endpoint untuk *push-notifikasi*
3. Membuat API endpoint untuk Ajuan donor darah darurat

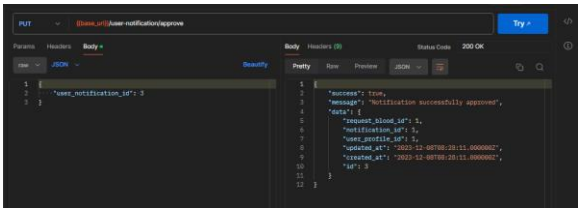
b. Sprint Execution & Daily Scrum

Pada tahap ini, sprint backlog dieksekusi dengan mendahulukan pengerjaan fitur utama yaitu *push-notifikasi* dan ajuan riwayat darah darurat, selain itu dikerjakan pula desain antarmuka untuk dashboard dengan user role admin dan UTD PMI.

c. Sprint Review

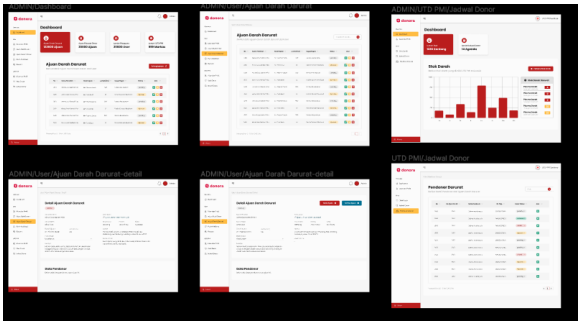
API endpoint notifikasi dibuat agar user yang telah menerima notifikasi darurat dapat melakukan *approve* yang menandakan bahwa

user bersedia menjadi pendonor darurat sesuai ajuan yang tercantum.



Gambar 9. Endpoint dan response API approve ajuan permintaan darah darurat

Selain itu, antarmuka secara keseluruhan telah selesai dibuat pada iterasi ke-2 sprint. Desain meliputi halaman home dashboard, serta menu-menu yang disesuaikan dengan user role, yaitu admin dan UTD PMI.



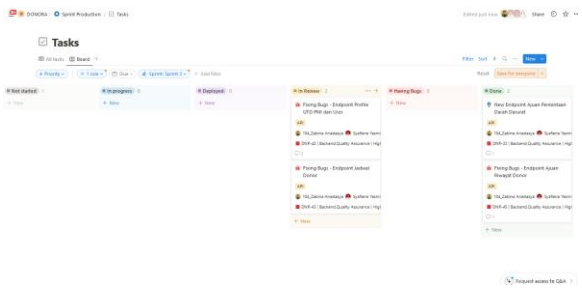
Gambar 10. Desain antarmuka dashboard Donora

d. Sprint Retrospective

Dilakukan diskusi *sprint retrospective* pada akhir dari iterasi sprint ke-2 yang membahas kinerja dari iterasi ini dengan performa yang kurang baik. Fitur *push-notifikasi* memakan waktu pengerjaan lebih lama dari yang diharapkan dan mengalami beberapa bug di akhir sprint. Hal ini menyebabkan pekerjaan pada backlog untuk fitur ajuan darah darurat tidak dapat diselesaikan dalam sprint ini dan akan dikerjakan pada iterasi sprint selanjutnya yaitu iterasi sprint ke-3.

4.2.3. Iterasi Sprint ke-3

a. Sprint Planning



Gambar 11. Sprint board iterasi ke-3

Dilakukan diskusi untuk menentukan sprint backlog atau kumpulan tugas yang akan dikembangkan pada iterasi ke-2, yaitu :

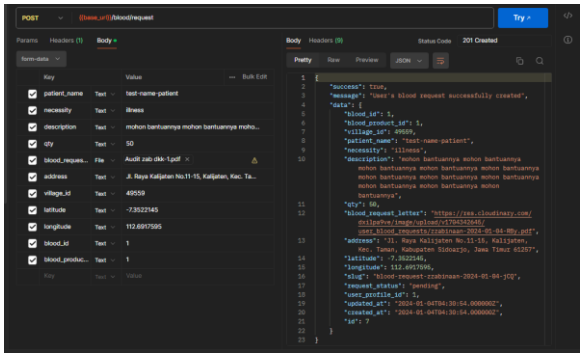
1. Fixing bug pada API UTD PMI dan Profile User
2. Membuat API endpoint Ajuan Permintaan Darah Darurat
3. Persiapan *environment* untuk dashboard Admin dan UTD PMI

b. Sprint Execution & Daily Scrum

Pada tahap ini sprint backlog dieksekusi, dilakukan penyelsaian fixing bug untuk task pada iterasi sebelumnya kemudian melanjutkan pengerjaan fitur utama yaitu API Ajuan permintaan darah darurat.

c. Sprint Review

Fitur ajuan permintaan darah darurat telah dibuat, user dapat mengajukan permintaan darah darurat dan akan diverifikasi dan dikelola oleh admin kedepannya.



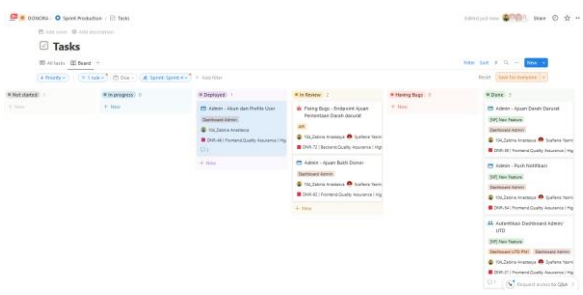
Gambar 12. Endpoint dan response API create ajuan permintaan darah darurat

d. Sprint Retrospective

Dilakukan diskusi *sprint retrospective* pada akhir dari iterasi sprit ke-2. Dengan hasil yaitu, performa tim dalam mengerjakan task dinilai baik karena telah mencapai target dengan menyelesaikan backlog tepat waktu. Namun beberapa bug ditemukan di akhir sprint sehingga belum cukup waktu untuk dapat menyelesaikannya dan akan dikerjakan pada iterasi sprint selanjutnya.

4.2.4. Iterasi Sprint ke-4

a. Sprint Planning



Gambar 13. Sprint board iterasi ke-4

Dilakukan diskusi untuk menentukan sprint backlog atau kumpulan tugas yang akan dikembangkan pada iterasi ke-2, yaitu :

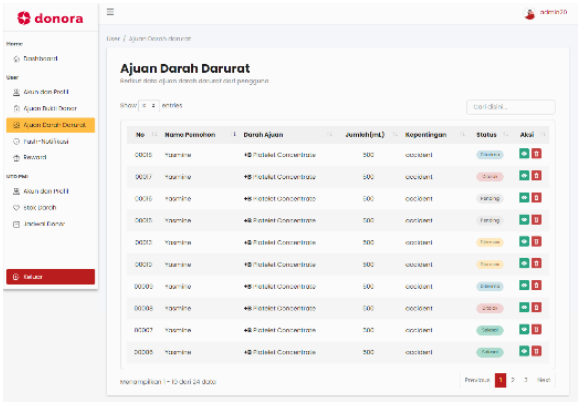
1. Fixing bug API Endpoint Ajuan Permintaan Darah Darurat
2. Membuat halaman ajuan darah darurat pada dashboard admin

b. Sprint Execution & Daily Scrum

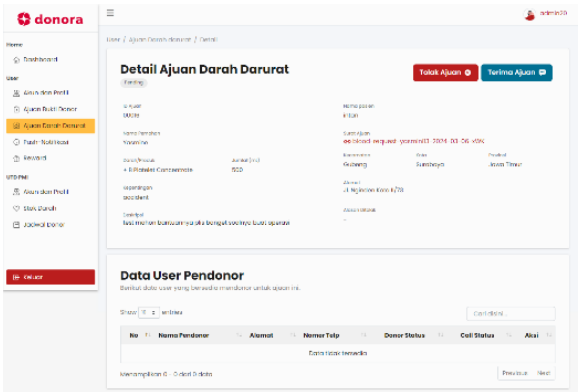
Pada tahap ini sprint backlog dieksekusi, dilakukan penyelsaian fixing bug untuk task pada iterasi sebelumnya kemudian melanjutkan pengerjaan ajuan darah darurat pada halaman dashboard admin dan UTD PMI

c. Sprint Review

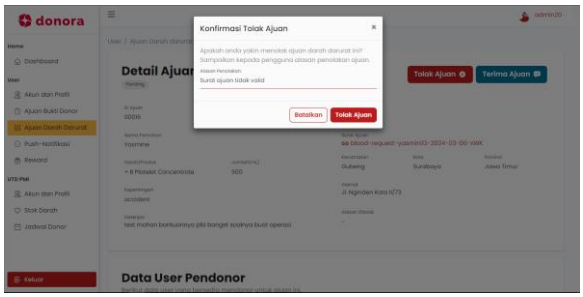
Halaman admin untuk permintaan darah darurat dibuat untuk memberikan kemampuan kepada admin dalam mengelola permintaan darah darurat dari pengguna dengan menerima atau menolak permintaan tersebut. Jika permintaan diterima, admin akan mengirimkan pemberitahuan kepada semua pengguna.



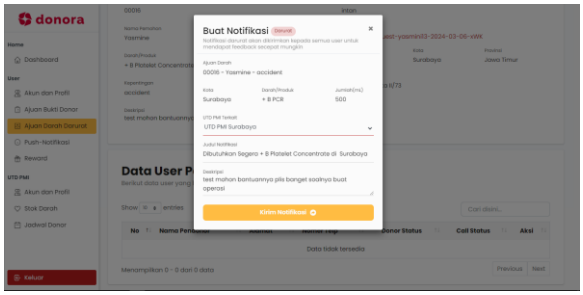
Gambar 14. Dashboard admin halaman ajuan darah darurat



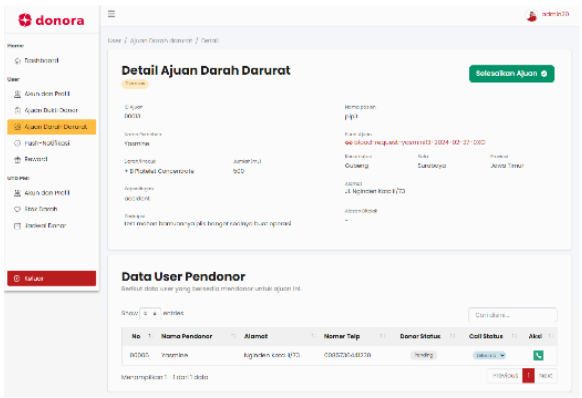
Gambar 15. Dashboard admin halaman detail ajuan darah darurat



Gambar 16. Pop-up alasan penolakan ajuan darah darurat



Gambar 17. Pop-up buat notifikasi untuk ajuan darah darurat



Gambar 18. Dashboard admin halaman detail ajuan darah darurat saat telah memiliki pendonor

d. Sprint Retrospective

Dilakukan diskusi *sprint retrospective* pada akhir dari iterasi sprit ke-4. Dengan hasil yaitu, performa tim dalam mengerjakan task dinilai baik karena telah mencapai target dengan menyelesaikan backlog tepat waktu.

4.3. Testing

Pengujian black box sistem donora dilakukan dengan metode manual testing, yaitu proses pengujian secara manual oleh penguji dengan *test case* yang telah disiapkan sebelumnya. Hasil dari pengujian sistem donora adalah sebagai berikut :

a. Akun dan profil user

Tabel 2. Hasil testing modul akun dan profile user

Fitur	Test case description	Status
View All Pengguna	Admin dapat melihat seluruh list pengguna yang ada	Passed
	Admin dapat melihat list pengguna berdasarkan pagination (klik previous/next)	Passed
	Admin dapat melihat list pengguna dengan memilih banyaknya data yang akan ditampilkan (show entries)	Passed
	Admin dapat melihat list pengguna dan disortir menggunakan ascending dan descending di seluruh kolom	Passed
	Admin dapat mengubah status pengguna menjadi aktif/ disable dengan klik dropdown pada kolom status di pengguna yang dipilih	Passed
Search Pengguna	Admin dapat mencari data pengguna berdasarkan data yang tersedia dan menampilkan hasilnya	Passed
	Admin dapat mencari data pengguna berdasarkan data yang tidak tersedia	Passed
	Admin tidak dapat mencari data pengguna jika search field diisi dengan full spasi	Passed
	Admin dapat melihat detail pengguna yang dipilih	Passed

b. Push-Notifikasi

Tabel 3. Hasil testing modul push-notifikasi

Fitur	Test case description	Status
Create Notifikasi	Admin membuat notifikasi dengan mengisi form buat notifikasi secara lengkap dan sesuai dengan form 7. Tambah Push Notifikasi pada sheet ketentuan input data, tampil di list notifikasi, dan terkirim ke user (mobile dan web)	Passed
	Admin membuat notifikasi dengan mengosongkan seluruh field pada form buat notifikasi	Passed
	Admin membuat notifikasi dengan mengosongkan/ tidak memilih opsi pada field level	Passed
	Admin membuat notifikasi dengan mengisi field judul notifikasi dengan > 250 karakter	Passed
	Admin membuat notifikasi dengan mengisi field judul notifikasi dengan full spasi	Passed
	Admin membuat notifikasi dengan mengisi field deskripsi dengan < 10 karakter	Passed
	Admin membuat notifikasi dengan mengisi field deskripsi dengan full spasi	Passed
	Admin membuat notifikasi darurat dengan mengosongkan/ tidak memilih pada field ajuan darah	Passed

c. Ajuan Permintaan Darah Darurat

Tabel 4. Hasil testing modul ajuan darah darurat

Fitur	Test case description	Status
View All Ajuan Darah Darurat	Admin dapat melihat list ajuan darah darurat berdasarkan pagination (klik previous/next)	Passed
	Admin dapat melihat list ajuan darah darurat berdasarkan pagination (klik previous/next)	Passed
	Admin dapat melihat list ajuan darah darurat dengan memilih banyaknya data yang akan ditampilkan (show entries)	Passed
	Admin dapat melihat list ajuan darah darurat dan disortir menggunakan ascending dan descending di seluruh kolom	Passed
Search Ajuan Darah Darurat	Admin dapat mencari data ajuan darah darurat berdasarkan data yang tersedia dan menampilkan hasilnya	Passed
	Admin dapat mencari data ajuan darah darurat berdasarkan data yang tidak tersedia	Passed
	Admin tidak dapat mencari data ajuan darah darurat jika search field diisi dengan full spasi	Passed
Reject Ajuan Darah Darurat	Admin dapat menolak ajuan darah darah darurat dengan mengisi field alasan ditolak dan status ajuan berubah menjadi "ditolak"	Passed
	Admin dapat menolak ajuan darah darah darurat dengan mengisi field alasan ditolak menggunakan full spasi	Passed
	Admin dapat menolak ajuan darah darah darurat dengan mengisi field alasan ditolak > 250 karakter	Passed
	Admin dapat menolak ajuan darah darah darurat dengan mengisi field alasan ditolak, kemudian membatalkannya	Passed
Accept Ajuan Darah Darurat	Admin dapat menerima ajuan darah darurat dengan mengklik tombol terima ajuan dan mengisi seluruh field secara lengkap dan benar pada form buat notifikasi sesuai pada Form 7. Tambah Push Notifikasi sheet ketentuan input data	Passed
	Admin dapat menerima ajuan darah darurat dengan mengklik tombol terima ajuan dan mengosongkan seluruh field yang dapat diisi	Passed
	Admin membuat notifikasi dengan mengisi field judul notifikasi/ header dengan > 250 karakter	Passed
	Admin membuat notifikasi dengan mengisi field judul notifikasi/ header dengan full spasi	Passed

	Admin membuat notifikasi dengan mengisi field deskripsi dengan < 10 karakter	Passed
	Admin membuat notifikasi dengan mengisi field deskripsi dengan full spasi	Passed
	Admin membuat notifikasi darurat dengan mengosongkan/ tidak memilih pada field UTD PMI Terkait	Passed
Process Ajuan Darah Darurat	Admin dapat melihat data user pendonor pada halaman detail ajuan darah darurat menggunakan pagination (next and previous)	Passed
	Admin dapat menghubungi pendonor darurat dengan klik **icon phone**, kemudian mengubah status call pada user pendonor ketika bersedia donor menjadi "diterima"	Passed
	Admin dapat menghubungi pendonor darurat dengan klik **icon phone**, kemudian mengubah status call pada user pendonor ketika tidak bersedia donor menjadi "ditolak"	Passed
Process Ajuan Darah Darurat	Admin dapat mengubah status ajuan darah menjadi "diproses" ketika status call : diterima, donor status : pending, dan status ajuan : diterima dengan klik **Proses Ajuan**	Passed
	Admin mengubah status ajuan darah menjadi "diproses" ketika tidak ada status call yg "diterima", dengan klik **Proses Ajuan**	Passed
Finish Ajuan Darah Darurat	Admin dapat mengubah status ajuan darah menjadi "Selesai" ketika terdapat status donor : distribusi, status call : diterima, dan status ajuan : diproses, dengan klik **Selesaikan Ajuan**	Passed
	Admin mengubah status ajuan darah menjadi "Selesai" ketika donor status : "diproses" dengan klik **Proses Ajuan**	Passed
	Admin mengubah status ajuan darah menjadi "Selesai" ketika donor status : "ditolak/ pending" dengan klik **Proses Ajuan**	Passed

5. KESIMPULAN DAN SARAN

Melalui hasil dan pembahasan yang telah dilakukan, Donora telah dibuat menggunakan metode SCRUM yang membutuhkan sebanyak 4 iterasi dengan tahapan Product Backlog, Sprint Planning, Sprint Execution, Daily Scrum, Sprint Review, dan Sprint Retrospective di dalamnya.

Donora terbukti berhasil berfungsi dengan baik, yang terlihat dari hasil pengujian menggunakan metode black box dengan test case, yang semuanya menunjukkan hasil pengujian yang lolos.

Namun, disarankan untuk melakukan pengujian lebih lanjut guna memperdalam pemahaman terhadap kinerja Donora, termasuk evaluasi terhadap penerimaan pengguna terhadap sistem ini. Selain itu, pengembangan aplikasi ini dapat terus dilakukan seiring berjalannya waktu untuk mengatasi permasalahan stok darah yang kosong di Indonesia serta meningkatkan kesadaran masyarakat akan pentingnya donor darah

DAFTAR PUSTAKA

[1] Kementerian Kesehatan Republik Indonesia, "Peringatan Hari Donor Darah Sedunia Tahun 2023," Jun. 16, 2023. Available: https://yankes.kemkes.go.id/read/1188/peringatan-hari-donor-darah-sedunia-tahun-2023

[2] S. Widi, "Stok Darah di Indonesia Sebanyak 77.438 Kantong per 14 Juni 2023," *DataIndonesia.id*, Jun. 14, 2023. Available: https://dataindonesia.id/varia/detail/stok-darah-di-indonesia-sebanyak-77438-kantong-per-14-juni-2023

[3] S. Suartini and A. Ikhwan, "Management Information System Web-Base on Blood Donation Service," *Sink. J. dan Penelit. Tek. Inform.*, vol. 7, no. 1, pp. 222–230, 2023, doi: 10.33395/sinkron.v8i1.11920

[4] Hasanuddin, H. Asgar, and B. Hartono, "Rancang Bangun REST API Aplikasi Weshare sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan," *JINTEKS (Jurnal Inform. Teknol. dan Sains)*, vol. 4, no. 1, pp. 8–14, 2022, doi: 10.51401/jinteks.v4i1.1474

[5] H. R. Suharno, N. Gunantara, and M. Sudarma, "Analisis Penerapan Metode Scrum pada Sistem Informasi Manajemen Proyek dalam Industri & Organisasi Digital," *Maj. Ilm. Teknol. Elektro*, vol. 19, no. 2, pp. 203–210, 2020, doi: 10.24843/MITE.2020.v19i02.P12

[6] Assyifannisa and A. B. L. Mailangkay, "Rancang Bangun Sistem Informasi Rekam Medis Klinik Dokter Gigi Alfa Dental Care dengan Metode RAD (Rapid Application Development) Berbasis Hybridapp," *Semin. Nas. Perbanas Inst.*, vol. 1, no. 1, pp. 158–165, 2021.

[7] P. I. Dharma and Sumarno, "Website-Based Sales Reporting Information System with the Laravel Framework (Case Study of Pramana Agency)," *Procedia Eng. Life Sci.*, vol. 2, no. 2, 2022, doi: 10.21070/pels.v2i2.1278

[8] R. Somya and M. Aprillia, "Perancangan Aplikasi Push Notification Center dengan Teknologi Firebase Cloud Messaging di PT. Sumber Trijaya Lestari," *Simetris J. Tek. Mesin, Elektro dan Ilmu Komput.*, vol. 10, no. 1, pp. 211–222, 2019, doi: 10.24176/simet.v10i1.2935

[9] K. S. Rubin, *Essential Scrum: A Practical Guide to the Most Popular Agile Process*, 1st ed. Addison-Wesley, 2012.

[10] R. M. Wijaya and A. B. Cahyono, "Pengembangan Aplikasi Sajiloka Menggunakan Metode Scrum," *AUTOMATA*, vol. 3, no. 2, 2022.