BAB 3
METODOLOGI PENELITIAN

3.1 Metodologi Penelitian

Metode penelitian yang digunakan dalam rancang bangun aplikasi pembelajaran bahasa Korea mnggunakan metode gamifikasi Octalysis berbasis Website adalah sebagai berikut.

1. Studi Literatur
   Pada tahap ini, melakukan pencarian dan mempelajari hal-hal terkait dalam penelitian. Proses studi literatur dilakukan dengan mempelajari artikel-artikel maupun jurnal dari *internet* yang berhubungan dengan sumber-sumber yang dipakai dalam penelitian, mempelajari bahasa Korea lebih dalam lagi untuk mendapatkan pemahaman yang lebih baik dalam perancangan aplikasi. Meneliti prinsip gamifikasi Octalysis untuk diimplementasikan dalam pembuatan aplikasi.

2. Perancangan Aplikasi
   Perancangan aplikasi dilakukan dengan merancang fitur apa saja yang ada dengan metode gamifikasi Octalysis, perancangan model aplikasi, struktur *database* dan UML Diagram. Selanjutnya dilanjutkan dengan perancangan *user interface* dan pemilihan aset.

3. Pembuatan Aplikasi
   Dalam pembuatan aplikasi, frontend akan dikembangkan menggunakan Tailwind dan React.js sedangkan untuk backend akan dikembangkan menggunakan Spring Boot dan PostgreSql sebagai *database*.

4. Pengujian Aplikasi
   Pengujian pada aplikasi yang dibuat akan dilakukan untuk memastikan bahwa aplikasi bekerja dan berfungsi dengan baik sesuai dengan model dan rancangan yang telah dibuat. Pengujian ini akan dilakukan memakai *browser* Google Chrome.

5. Deployment
   Aplikasi akan dijalankan dan dapat di-akses secara publik. Frontend akan

berjalan di atas server Vercel, dan Backend akan berjalan di atas server Google Cloud.

6. Evaluasi
   Aplikasi yang telah dibuat akan disebarkan kepada sejumlah responden yang memiliki minat untuk belajar bahasa Korea. Para pengguna yang telah mencoba aplikasi tersebut akan menjawab beberapa pertanyaan berdasarkan model HMSAM dan menggunakan skala Likert melalui kuesioner yang telah disebarkan secara *online*. Kuesioner yang telah diisi oleh responden tersebut akan digunakan untuk mengukur tingkat *behavioral intention to use* dan *immersion* dari aplikasi yang telah dibuat. Evaluasi dilakukan dengan mengolah data yang diperoleh dari jawaban kuesioner yang diisi oleh para responden setelah mencoba aplikasi tersebut. Kuesioner ini menggunakan model Hedonic-Motivation System Adoption Model (HMSAM) dan skala Likert.

7. Penulisan Laporan
   Mendokumentasikan seluruh proses perancangan dan pembangunan aplikasi dari awal sampai akhir kedalam laporan agar dapat memberikan informasi bagi penelitian serupa. Setelah itu, melakukan konsultasi dengan dosen pembimbing untuk memastikan bahwa penelitian telah dilakukan dengan tepat dan mendapatkan hasil yang memuaskan.

3.2 Perancangan Aplikasi

Perancangan aplikasi terdiri dari rancangan fitur yang ada dengan gamifikasi Octalysis, rancangan model aplikasi, struktur *database*, UML Diagram, rancangan desain antarmuka dan pemilihan aset.

3.2.1 Rancangan Fitur dengan Metode Gamifikasi Octalysis

1. Epic Meaning & Calling.
   Pada *core drive* ini diterapkan *Free Lunch* dimana pengguna akan mendapatkan *item* gratis setelah mendaftar menjadi *user* diaplikasi.

2. Development & Accomplishment.
   Pada *core drive* ini diterapkan *Badges* dan *Leaderboard*. Pengguna akan mendapatkan *badges* jika menyelesaikan kuis dengan nilai sempurna yang

akan ditampilkan pada halaman pengguna. *Leaderboard* diterapkan dengan menampilkan peringkat dari 50 pengguna teratas yang diurutkan sesuai dengan pengguna yang memiliki *score* terbanyak.

3. Empowerment of Creativity & Feedback.
   Pada *core drive* ini diterapkan *Milestone Unlock* yang diterapkan pada kuis dimana kuis hanya dapat diakses juga pengguna telah menyelesaikan pelajaran yang berkaitan dengan kuis tersebut.

4. Ownership & Possession
   Pada *core drive* ini diterapkan *Exchangeable Points, Virtual Goods* dan *Avatar*. Pengguna dapat menukarkan *point* yang didapat menjadi *virtual good* berupa efek *border* di foto profil dan *avatar* yang berupa foto profil pengguna pada halaman toko.

5. Social Influence & Relatedness
   Pada *core drive* ini diterapkan *Touting Flag* dimana pengguna dapat memamerkan *achievement* yang didapatnya pada halaman peringkat jika peringkat pengguna antara peringkat satu sampai tiga.

6. Scarcity & Impatience
   Pada *core drive* ini diterapkan *Price Pacing* dimana pengguna akan mendapatkan *points* saat menyelesaikan kuis atau pelajaran untuk pertama kalinya yang dapat digunakan untuk membeli barang pada halaman toko.

7. Unpredictability & Curiosity
   Pada *core drive* ini diterapkan *Easter Egg* dimana pengguna akan mendapatkan hadiah tersembunyi jika menyelesaikan hal yang berhubungan dengan *easter egg* tersebut.

8. Loss & Avoidance
   Pada *core drive* ini diterapkan *Progress Loss* dimana pengguna dapat hilang dari *leaderboard* karena terdapat pengguna baru yang memiliki *score* lebih tinggi.

3.2.2 Model Aplikasi

Aplikasi pembelajaran yang dibuat untuk belajar bahasa Korea bernama RumahHangeul. Aplikasi ini dirancang dengan tujuan untuk membantu masyarakat

di Indonesia yang berminat untuk belajar bahasa Korea tetapi tidak bisa bahasa Inggris. Aplikasi ini juga dirancang dengan menggunakan metode gamifikasi Octalysis dengan tujuan untuk meningkatkan motiviasi belajar para pengguna. Untuk menggunakan semua fitur aplikasi Rumah Hangeul, pengguna harus masuk terlebih dahulu jika tidak pengguna hanya dapat menggunakan fitur lihat peringkat saja.

Setelah masuk, terdapat dua cara untuk mendapatkan skor dan point. Cara yang pertama adalah menyelesaikan pembelajaran yang ada untuk pertama kalinya. Cara yang kedua adalah menyelesaikan kuis dimana untuk mengerjakan kuis pengguna harus menyelesaikan pembelajaran yang berkaitan terlebih dahulu. Skor yang didiapatkan oleh pengguna dapat membantu pengguna untuk menampilkan namanya di halaman peringkat, semakin tinggi skor pengguna akan semakin tinggi juga peringkatnya.

Poin yang didapatkan oleh pengguna dapat digunakan untuk membeli barang seperti foto profil ataupun efek *border* di foto profil pengguna dengan membuka halaman toko. Pengguna juga dapat mengubah nama depan dan nama belakangnya dengan membuka halaman profil pengguna dan menekan tombol ubah profil. Di halaman ubah profil sendiri terdapat tombol ubah foto yang dapat digunakan pengguna untuk mengubah foto profil yang dipakai ataupun efek *border*.

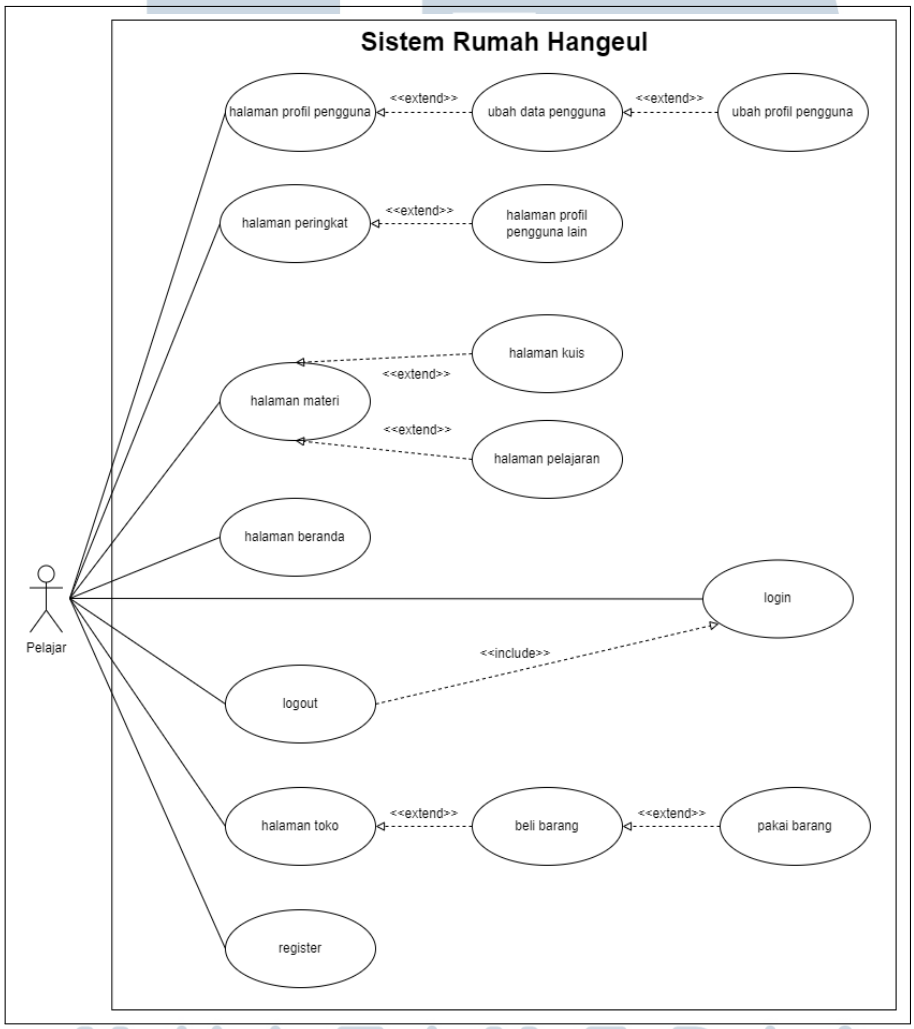
Pada halaman materi, pengguna dapat melihat pembelajaran dan kuis yang telah diselesaikan oleh pengguna dan yang belum diselesaikan serta melihat informasi mengenai cara pembelajaran dan mengerjakan kuis. Pada halaman toko, pengguna dapat membeli barang seperti foto profil dan efek *border* yang disukai, barang yang ditampilkan akan di *filter* terlebih dahulu dengan barang yang sudah dimiliki oleh pengguna dimana jika barang tersebut sudah dimiliki pengguna maka barang tersebut tidak akan ditampilkan pada halaman toko.

Pada halaman peringkat, pengguna dapat melihat profil tiga peringkat paling atas dan juga profil sendiri. Pada halaman lihat profil pengguna juga terdapat pelajaran dan kuis yang telah diselesaikan oleh pengguna, jumlah skor dan sisa poin pengguna serta *achievement* yang didapatkan oleh pengguna dengan menyelesaikan hal-hal tertentu sedangkan jika pengguna melihat profil pengguna lain, sisa poin tidak ditampilkan.

3.2.3 UML Diagram

A Use Case

Use Case Diagram adalah sebuah diagram yang dibuat dengan tujuan untuk menunjukkan bagaimana interaksi suatu sistem dengan entitas di luar sistem terjadi [17]. Interaksi antara entitas pelajar dengan sistem Rumah Hangeul pada aplikasi Rumah Hangeul ditunjukkan pada Use Case Diagram Gambar 3.1.



Gambar 3.1. Use Case Diagram Rumah Hangeul

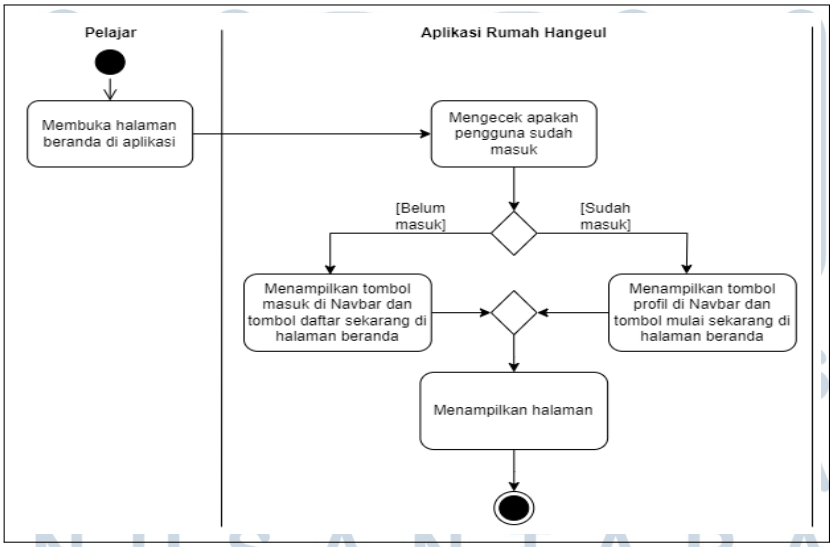
Pada skenario diatas, pelajar memiliki akses langsung untuk membuka halaman beranda, halaman materi, halaman peringkat, halaman toko dan halaman profil pengguna. Pelajar dapat membuka halaman kuis dan halaman pelajaran setelah pelajar membuka halaman materi. Untuk membuka halaman profil

pengguna lain, pelajar harus terlebih dahulu membuka halaman peringkat. Semua halaman tersebut akan memiliki tampilan yang berbeda saat pelajar sudah melakukan *login* dengan belum melakukan . Pelajar juga dapat langsung menggunakan fitur *login* dan *register*, sedangkan untuk fitur *logout* pelajar harus melakukan *login* terlebih dahulu baru dapat mengaksesnya. Pada halaman profil pengguna, pelajar dapat melakukan fitur ubah data pengguna serta ubah profil pengguna saat mengakses fitur ubah data pengguna. Pada halaman toko, pelajar dapat melakukan fitur beli barang dan setelah itu fitur pakai barang.

B Activity Diagram

Activity diagram merupakan diagram yang dibuat untuk menggambarkan urutan kegiatan dan tindakan yang terjadi pada suatu sistem[15].

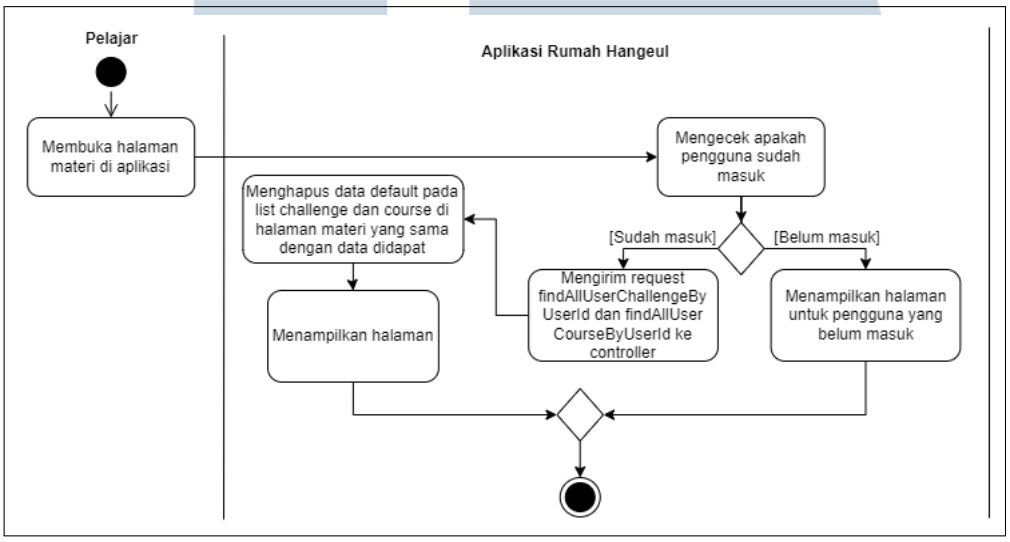
Pada *activity diagram* halaman beranda. Sistem akan mengecek apakah pelajar sudah masuk atau belum jika belum maka akan ditampilkan tombol masuk di Navbar dan tombol daftar sekarang di halaman beranda. Jika sudah maka sistem akan menampilkan tombol profil di Navbar menggantikan tombol masuk dan tombol mulai sekarang menggantikan tombol daftar sekarang. Lalu sistem akan menampilkan halaman beranda. *Activity diagram* halaman beranda ditunjukkan pada Gambar 3.2.



Gambar 3.2. Activity Diagram Halaman Beranda

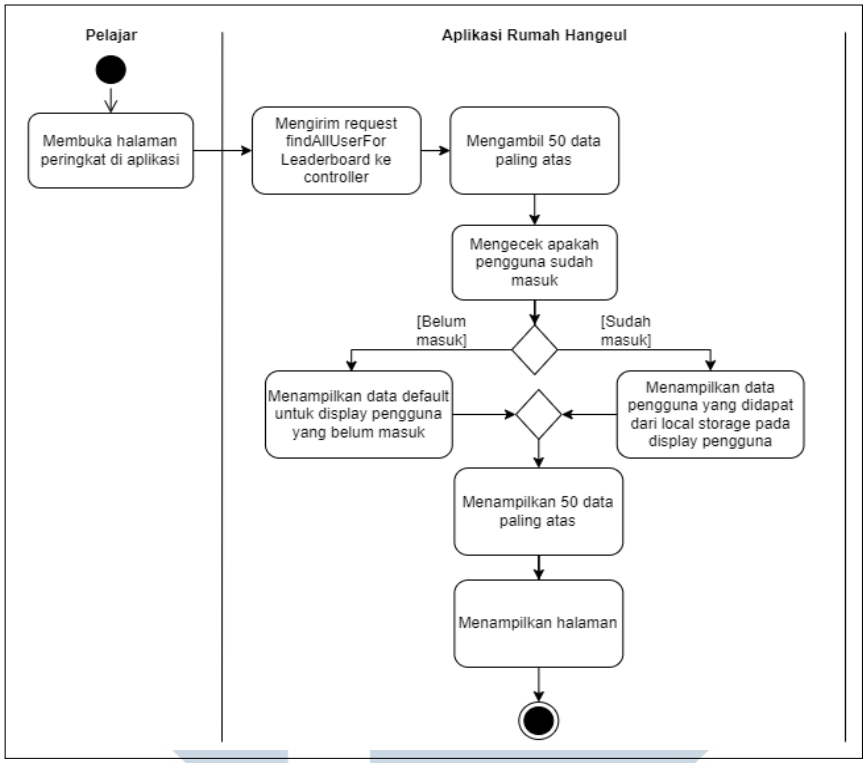
Pada *activity diagram* halaman materi. Sistem akan mengecek apakah pelajar sudah masuk atau belum jika belum maka sistem akan

menampilkan halaman dengan gambar dan tulisan ”Masuk dulu ya!”. Jika sudah maka sistem akan mengirim *request* findAllUserChallengeByUserId dan findAllUserCourseByUserId ke *controller*. Setelah itu sistem akan menghapus data *default* pada *list challenge* dan *course* jika terdapat data yang sama dengan data yang dikembalikan saat mengirimkan *request*. Lalu sistem akan menampilkan halaman dengan data yang sudah diolah tersebut pada halaman materi. *Activity diagram* halaman materi ditunjukkan pada Gambar 3.3.



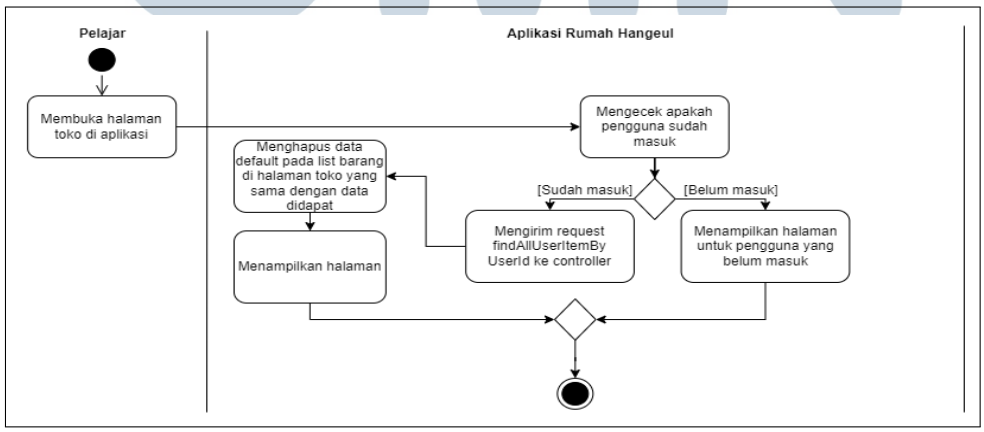
Gambar 3.3. Activity Diagram Halaman Materi

Pada *activity diagram* halaman peringkat. Sistem akan mengirim *request* findAllUserForLeaderboard ke *controller*. Lalu sistem akan mengambil 50 data paling atas dari data yang dikembalikan saat mengirimkan *request*. Lalu sistem akan mengecek apakah pelajar sudah masuk atau belum jika belum maka akan sistem akan menampilkan data *default* berupa ”Anda belum Masuk” sebagai nama depan dan nilai 0 untuk skor pada *display* pengguna. Jika sudah maka sistem akan menampilkan nama depan dan skor sesuai dengan data yang didapatkan dari *local storage* pada *display* pengguna. *Activity diagram* halaman peringkat ditunjukkan pada Gambar 3.4.



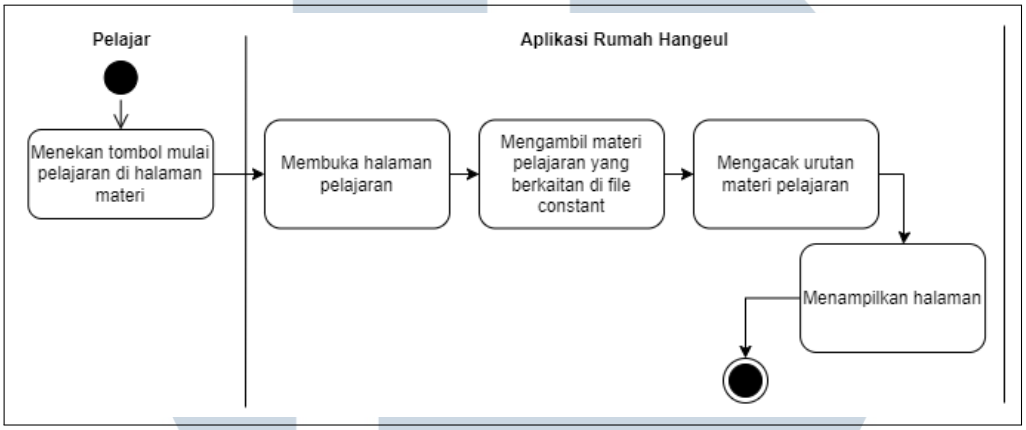
Gambar 3.4. Activity Diagram Halaman Peringkat

Pada *activity diagram* halaman toko. Sistem akan mengecek apakah pelajar sudah masuk atau belum jika belum maka sistem akan menampilkan halaman dengan gambar dan tulisan ”Masuk dulu ya!”. Jika sudah maka sistem akan mengirim *request* findAllUserItemByUserId ke *controller*. Setelah itu sistem akan menghapus data *default* pada *list* barang yang dijual pada halaman toko jika terdapat data yang sama dengan data yang dikembalikan saat mengirimkan *request*. *Activity diagram* halaman toko ditunjukkan pada Gambar 3.5.



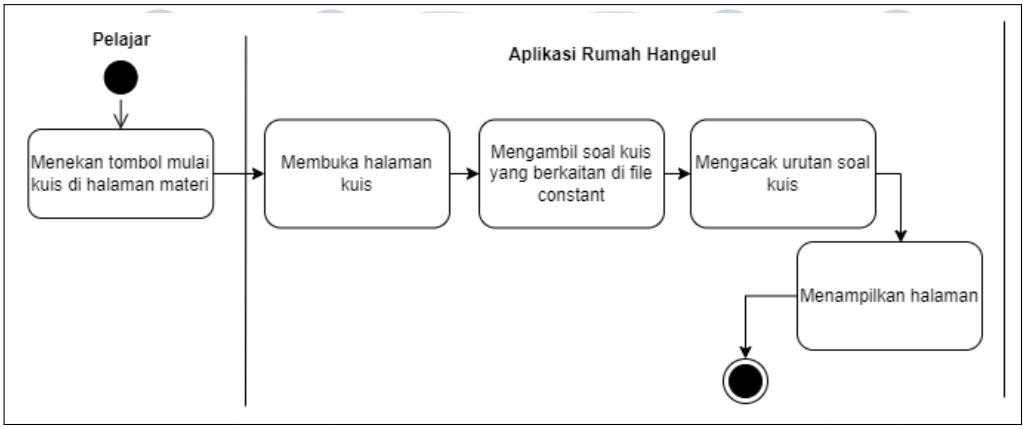
Gambar 3.5. Activity Diagram Halaman Toko

Pada *activity diagram* halaman pelajaran. Sistem akan mengambil materi pelajaran yang berkaitan di *file constant*. Lalu sistem akan mengacak urutan materi pelajaran dan menampilkan halaman pelajaran. *Activity diagram* halaman pelajaran ditunjukkan pada Gambar 3.6.



Gambar 3.6. Activity Diagram Halaman Pelajaran

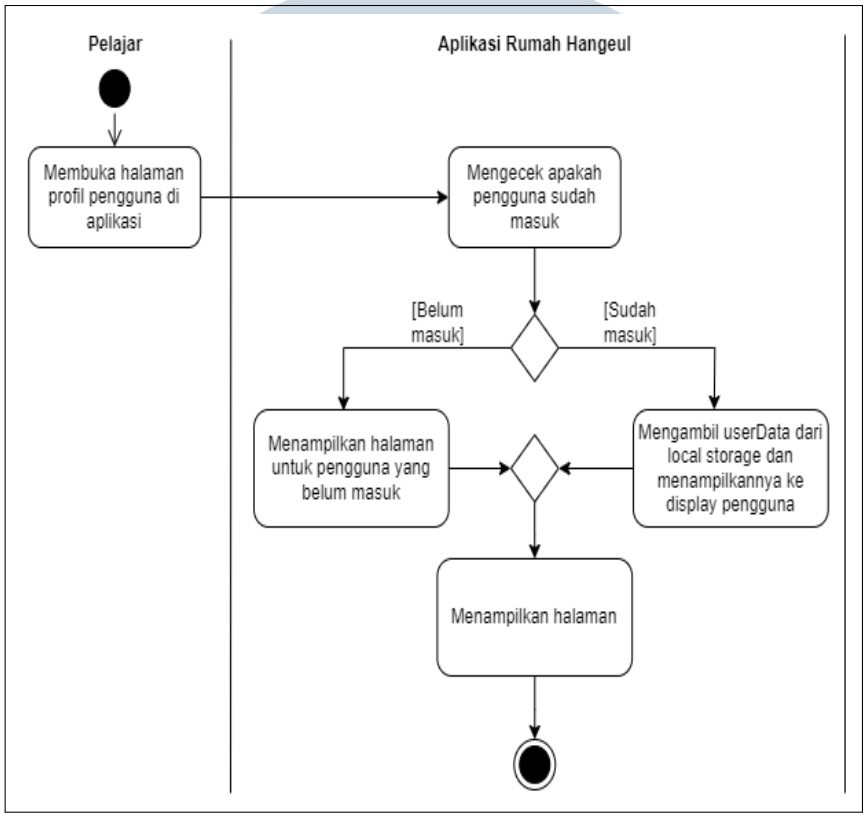
Pada *activity diagram* halaman kuis. Sistem akan mengambil soal kuis yang berkaitan di *file constant*. Lalu sistem akan mengacak urutan soal kuis dan menampilkan halaman kuis. *Activity diagram* halaman kuis ditunjukkan pada Gambar 3.7.



Gambar 3.7. Activity Diagram Halaman Kuis

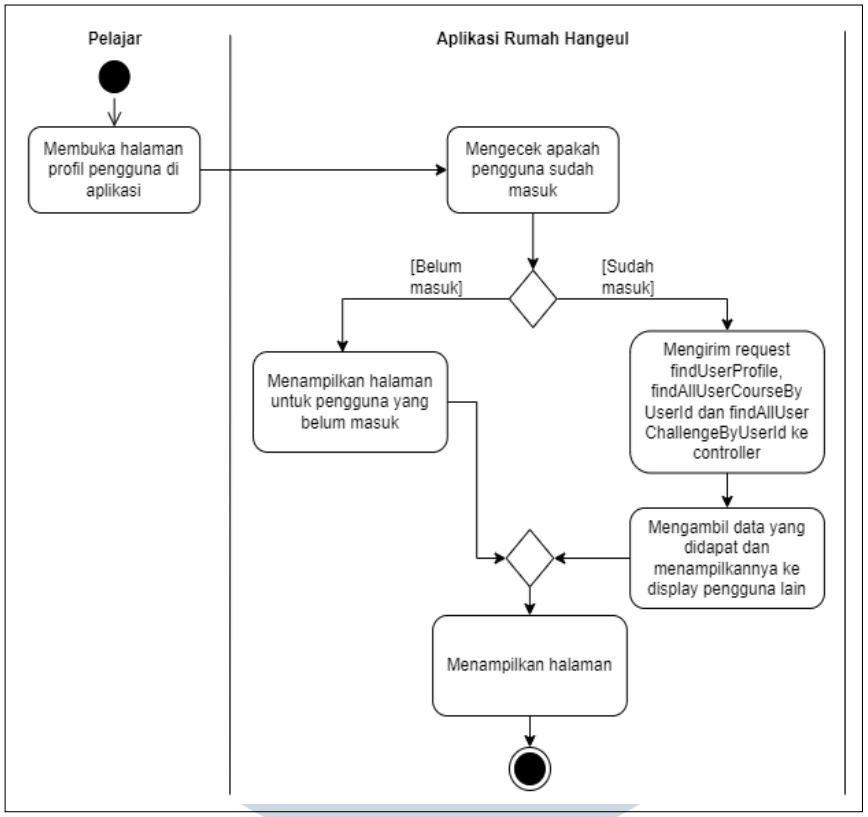
Pada *activity diagram* halaman profil pengguna. Sistem akan mengecek apakah pelajar sudah masuk atau belum jika belum maka sistem akan menampilkan halaman dengan gambar dan tulisan ”Masuk dulu ya!”. Jika sudah maka sistem akan mengambil userData dari *local storage* dan menampilkannya ke *display*

pengguna. Lalu sistem akan menampilkan halaman profil pengguna. *Activity diagram* halaman profil pengguna ditunjukkan pada Gambar 3.23.



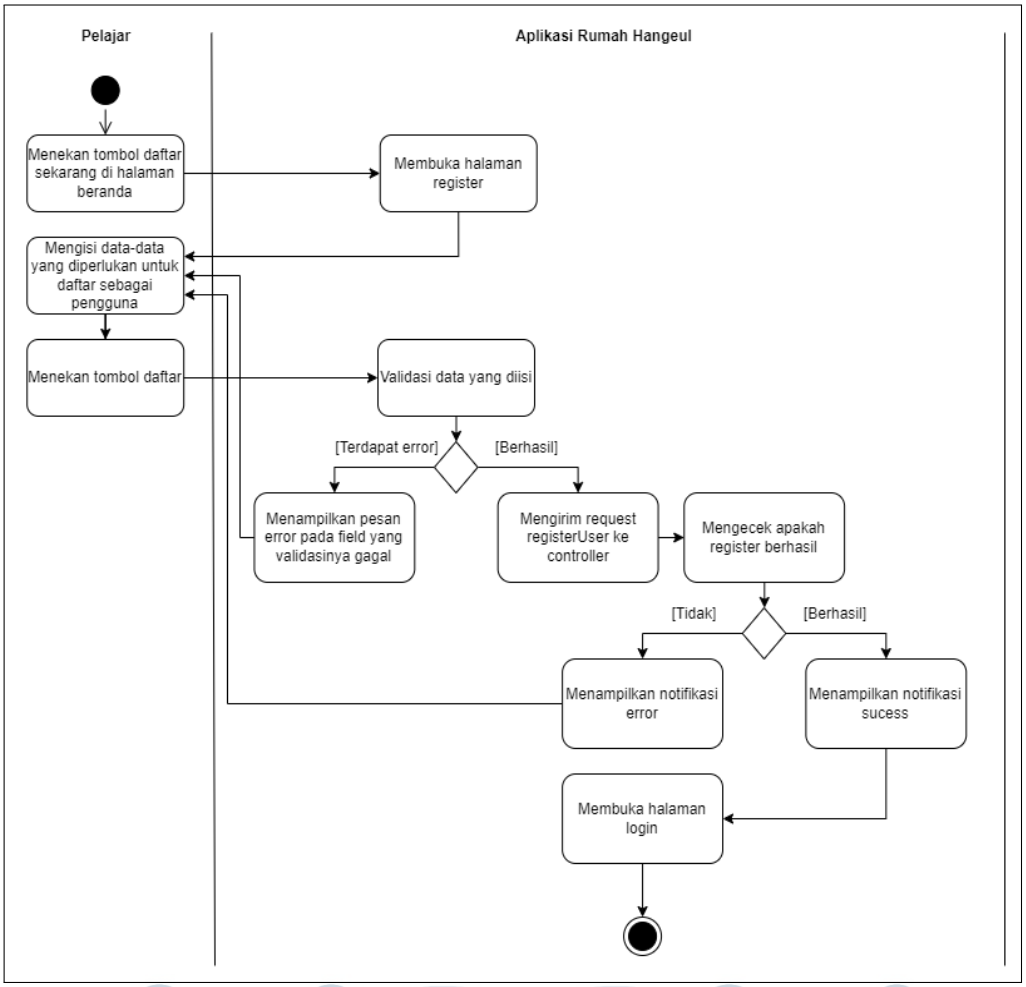
Gambar 3.8. Activity Diagram Halaman Profil Pengguna

Pada *activity diagram* halaman profil pengguna lain. Sistem akan mengecek apakah pelajar sudah masuk atau belum jika belum maka sistem akan menampilkan halaman dengan gambar dan tulisan ”Masuk dulu ya!”. Jika sudah maka sistem akan mengirimkan *request* findUserProfile, findUserChallengeByUserId dan findUserCourseByUserId ke *controller*. Setelah itu sistem akan menampilkan *display* pengguna sesuai dengan data yang dikembalikan saat mengirimkan *request*. Lalu sistem akan menampilkan halaman profil pengguna lain. *Activity diagram* halaman profil pengguna lain ditunjukkan pada Gambar 3.24.



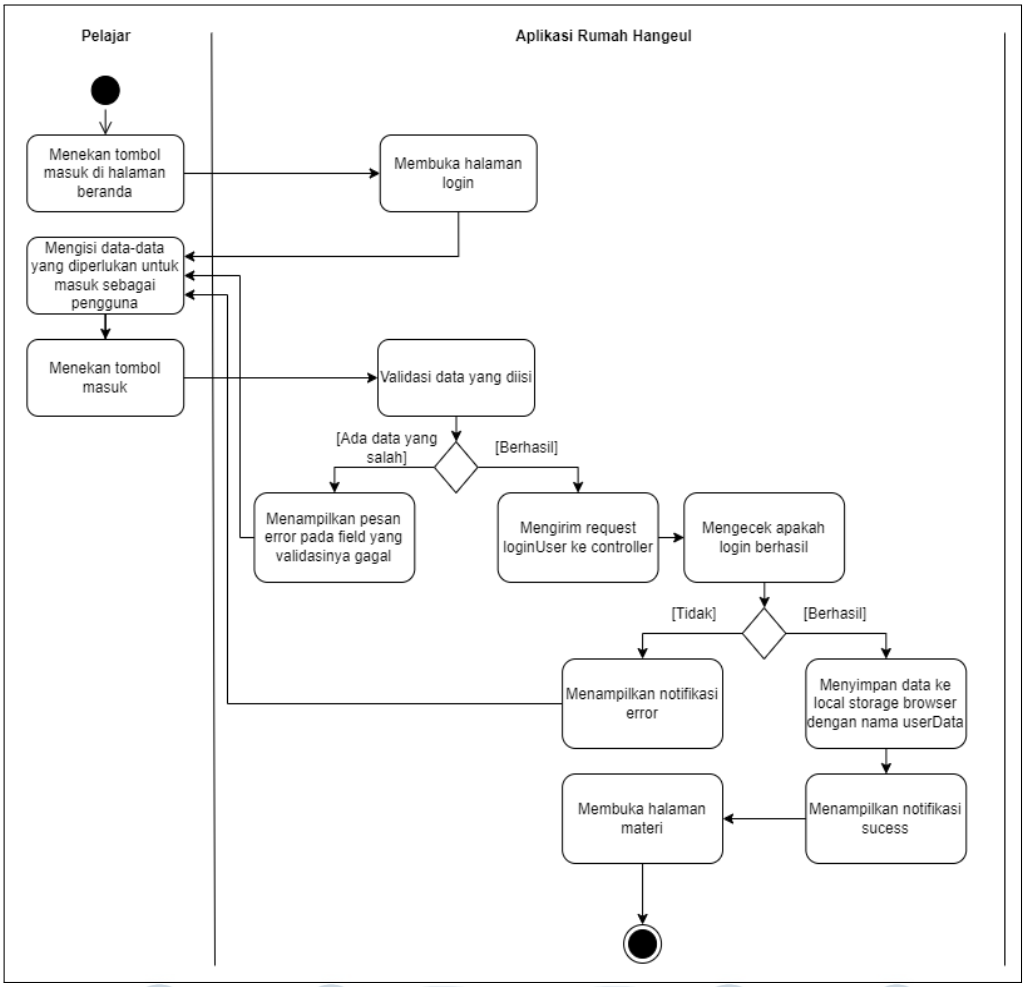
Gambar 3.9. Activity Diagram Halaman Profil Pengguna Lain

Pada *activity diagram register*, setelah pelajar menekan tombol daftar di halaman beranda maka sistem akan membawa pelajar ke halaman *register*. Lalu pelajar mengisi data-data yang diperlukan untuk daftar sebagai pengguna dan menekan tombol daftar. Kemudian sistem akan melakukan validasi pada data yang diisi oleh pelajar. Jika terdapat error maka sistem akan menampilkan pesan *error* pada *field* yang validasinya gagal. Jika berhasil maka sistem akan mengirimkan *request* registerUser ke *controller*. Setelah itu sistem akan mengecek apakah *register* berhasil atau tidak. Jika tidak maka sistem akan menampilkan notifikasi *error*. Sebaliknya jika berhasil sistem akan menampilkan notifikasi *success* dan membuka halaman *login*. *Activity diagram register* ditunjukkan pada Gambar 3.10.



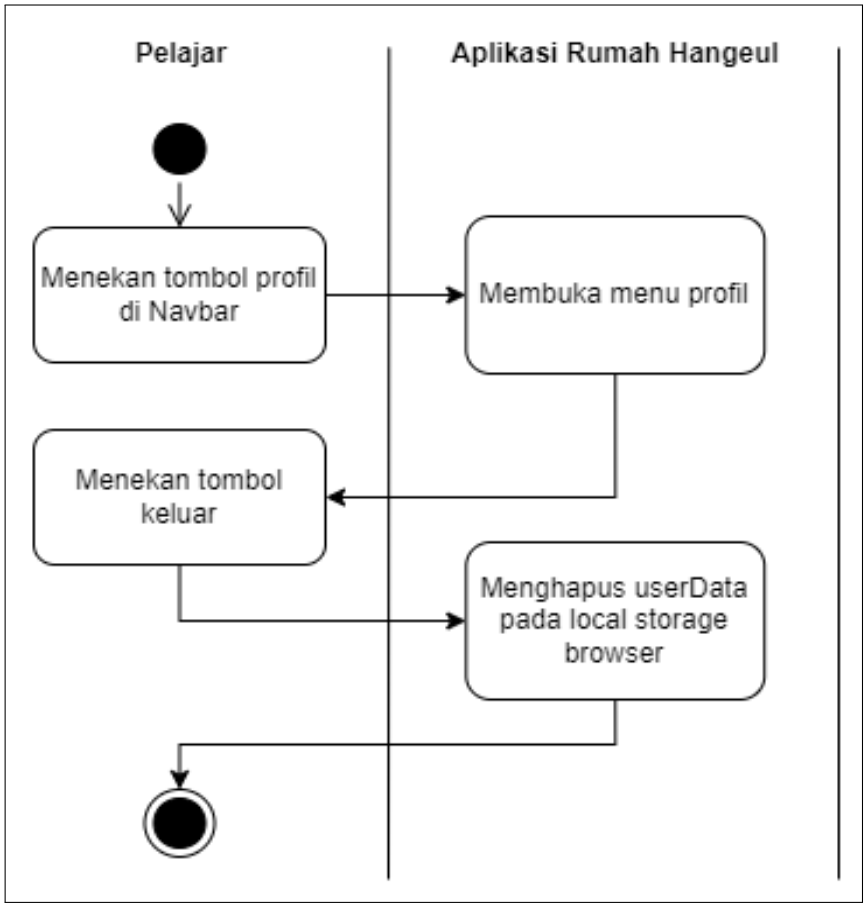
Gambar 3.10. Activity Diagram Register

Pada *activity diagram login*, setelah pelajar menekan tombol masuk di halaman beranda maka sistem akan membawa pelajar ke halaman *login*. Lalu pelajar mengisi data-data yang diperlukan untuk masuk sebagai pengguna dan menekan tombol masuk. Kemudian sistem akan melakukan validasi pada data yang diisi oleh pelajar. Jika terdapat error maka sistem akan menampilkan pesan *error* pada *field* yang validasinya gagal. Jika berhasil maka sistem akan mengirimkan *request* loginUser ke *controller*. Setelah itu sistem akan mengecek apakah *register* berhasil atau tidak. Jika tidak maka sistem akan menampilkan notifikasi *error*. Sebaliknya jika berhasil sistem akan menampilkan notifikasi *success* dan menyimpan data yang dikembalikan saat mengirimkan *request* ke *local storage* dengan nama userData . Setelah itu sistem akan membuka halaman materi. *Activity diagram login* ditunjukkan pada Gambar 3.11.



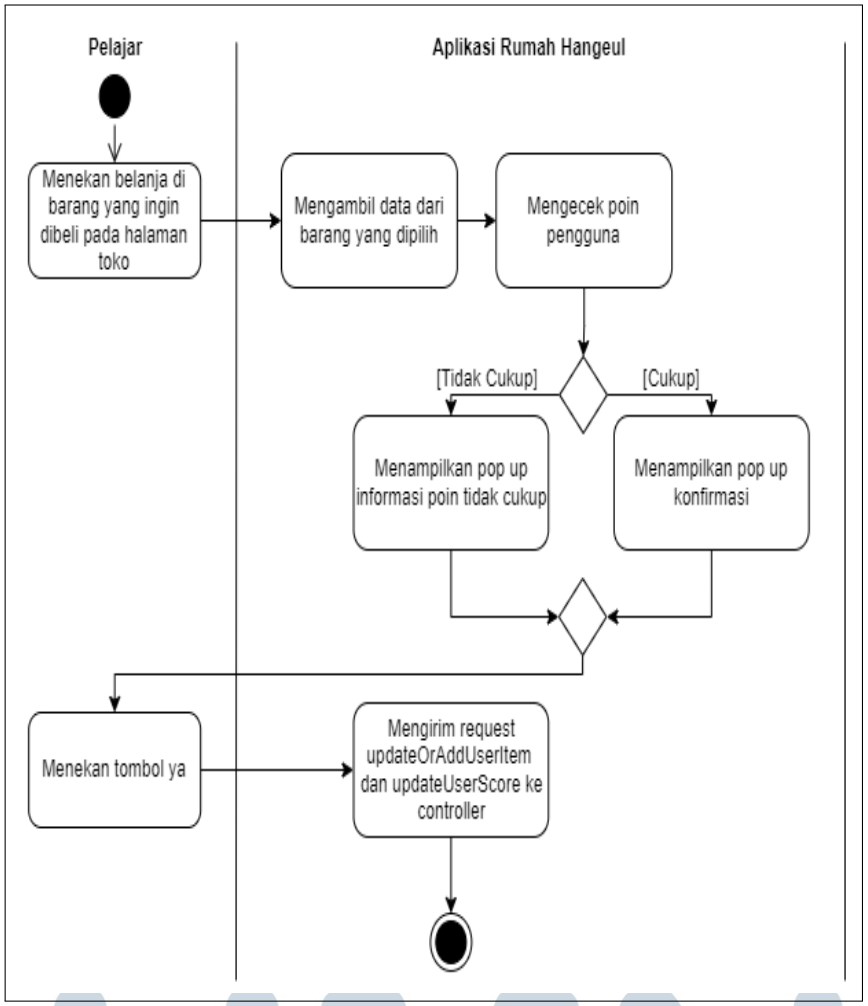
Gambar 3.11. Activity Diagram Login

Pada *activity diagram logout*, setelah pelajar menekan tombol profil di Navbar. Sistem akan membuka *menu profile*. Selanjutkan pelajar akan menekan tombol keluar maka sistem akan menghapus data ”userData” yang disimpan pada *local storage*. *Activity diagram logout* ditunjukkan pada Gambar 3.12.



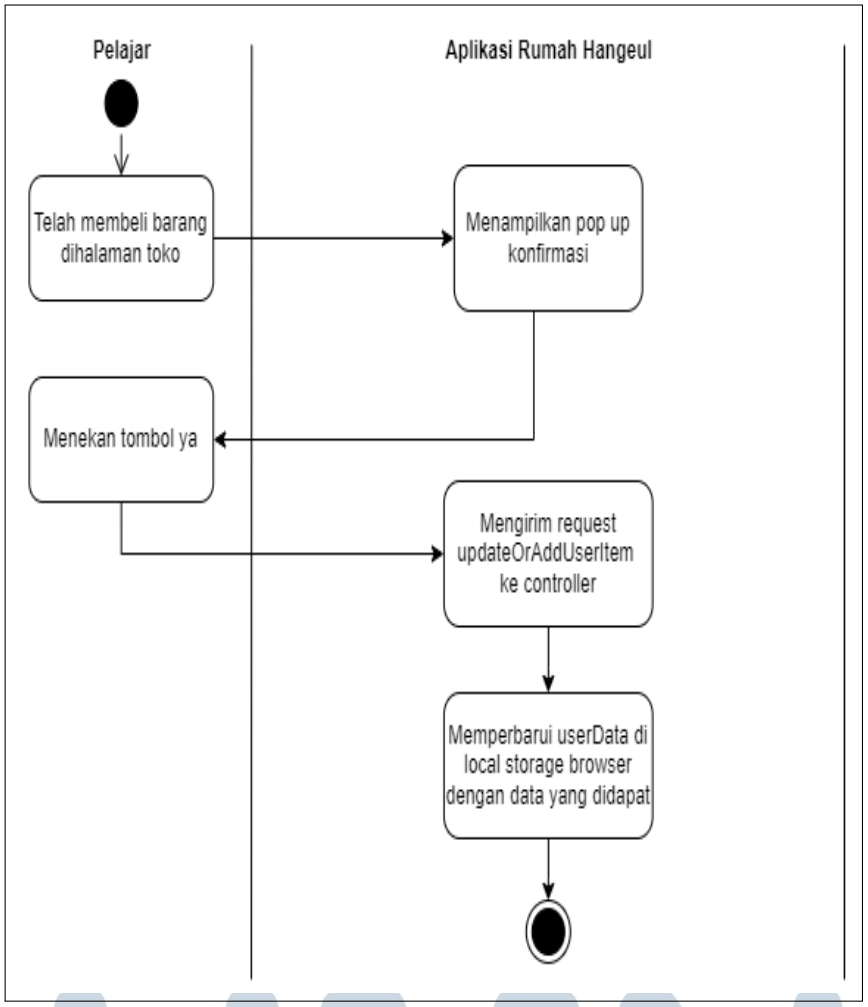
Gambar 3.12. Activity Diagram Logout

Pada *activity diagram* beli barang, setelah pelajar menekan belanja pada barang yang ingin dibeli pada halaman toko. Sistem akan mengambil data dari barang yang dipilih dan mengecek poin pengguna. Apabila poin tidak mencukupi maka sistem akan menampilkan *pop up* informasi point tidak cukup sedangkan jika mencukupi maka sistem akan menampilkan *pop up* konfirmasi. Setelah itu pelajar akan menekan tombol ya dan sistem akan mengirim *request* updateOrAddUserItem ke *controller*. *Activity diagram* beli barang ditunjukkan pada Gambar 3.13.



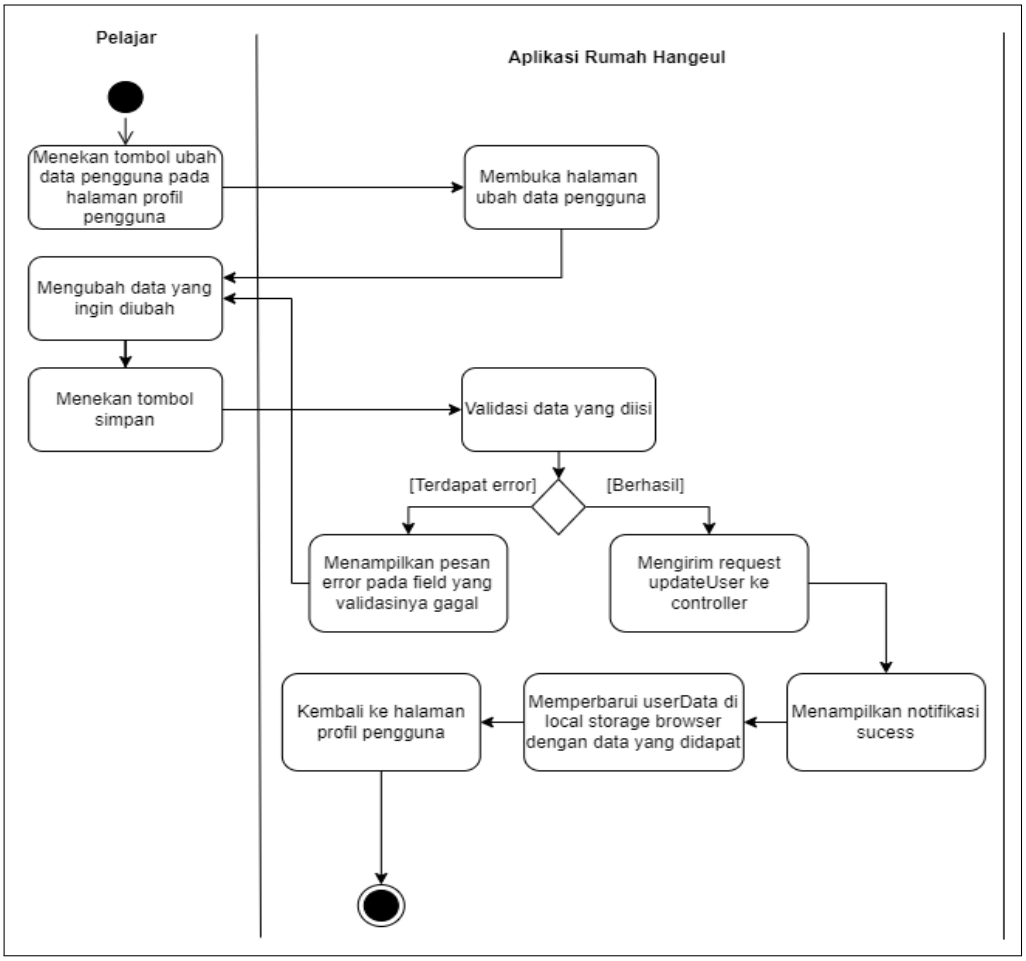
Gambar 3.13. Activity Diagram Beli Barang

Pada *activity diagram* pakai barang, setelah membeli barang pada halaman toko. Sistem akan menampilkan *pop up* konfirmasi. Kemudian pelajar akan menekan tombol ya dan sistem akan mengirim *request* updateUserProfile ke *controller* dan memperbarui userData di *local storage* dengan data yang dikembalikan saat mengirimkan *request*. *Activity diagram* pakai barang ditunjukkan pada Gambar 3.14.



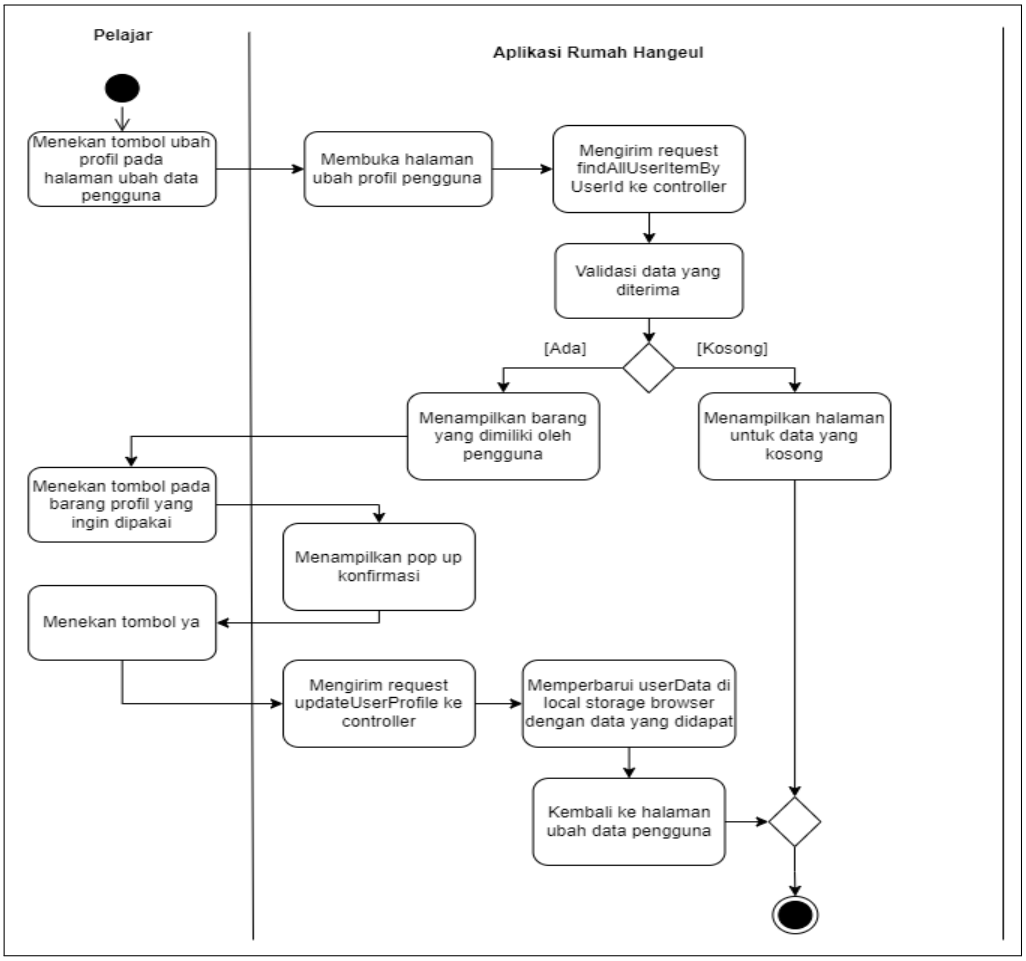
Gambar 3.14. Activity Diagram Pakai Barang

Pada *activity diagram* ubah data pengguna, setelah pelajar menekan tombol ubah data pengguna di halaman profil pengguna maka sistem akan membawa pelajar ke halaman ubah data pengguna. Lalu pelajar mengubah data-data yang ingin diubah dan menekan tombol simpan. Kemudian sistem akan melakukan validasi pada data yang diisi oleh pelajar. Jika terdapat error maka sistem akan menampilkan pesan *error* pada *field* yang validasinya gagal. Jika berhasil maka sistem akan mengirimkan *request* updateUser ke *controller*. Kemudian sistem akan memperbarui userData di *local storage* dengan data yang dikembalikan saat mengirimkan *request*. Setelah itu sistem akan membuka halaman profil pengguna. *Activity diagram* ubah data pengguna ditunjukkan pada Gambar 3.15.



Gambar 3.15. Activity Diagram Ubah Data Pengguna

Pada *activity diagram* ubah profil pengguna, setelah pelajar menekan tombol ubah profil pengguna di halaman ubah data pengguna maka sistem akan membawa pelajar ke halaman ubah profil pengguna. Setelah itu sistem akan mengirim *request* findAllUserItemByUserId ke *controller*. Kemudian sistem akan melakukan validasi pada data yang dikembalikan saat mengirim *request*. Apabila data tersebut kosong maka sistem akan menampilkan halaman untuk data yang kosong pada halaman ubah profil pengguna. Jika data tersebut tidak kosong maka sistem akan menampilkan barang apa saja yang dimiliki oleh pelajar. Kemudian pelajar menekan tombol barang yang ingin dipakai. Setelah itu sistem akan menampilkan *pop up* konfirmasi dan pelajar menekan tombol ya. Lalu sistem akan mengirim *request* updateUserProfile ke *controller* dan memperbarui userData di *local storage* dengan data yang dikembalikan saat mengirimkan *request*. Kemudian sistem akan membuka halaman ubah data pengguna. *Activity diagram* ubah profil pengguna ditunjukkan pada Gambar 3.16.

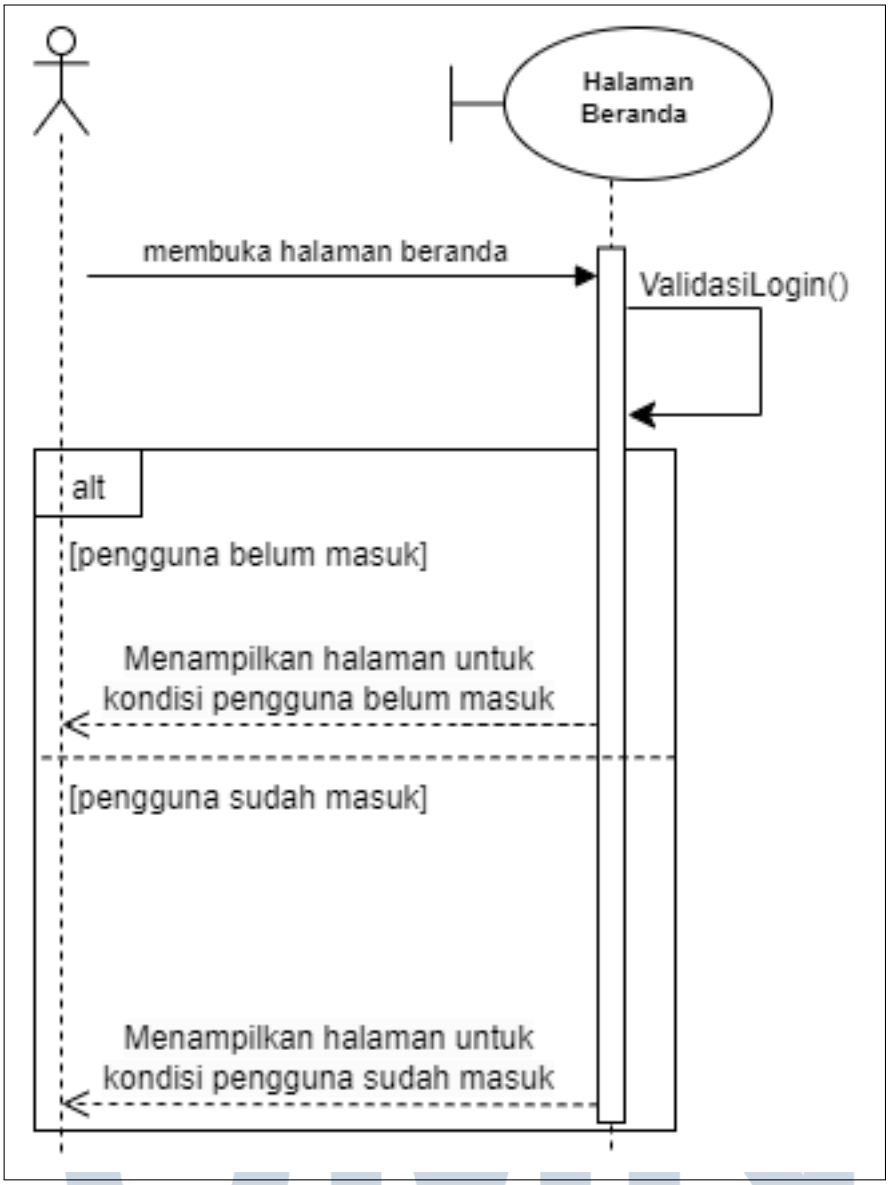


Gambar 3.16. Activity Diagram Ubah Profil Pengguna

C Sequence Diagram

Sequence diagram adalah sebuah diagram yang bertujuan untuk menjelaskan dan memperlihatkan interaksi dan komunikasi antara objek-objek yang ada pada sebuah sistem [18].

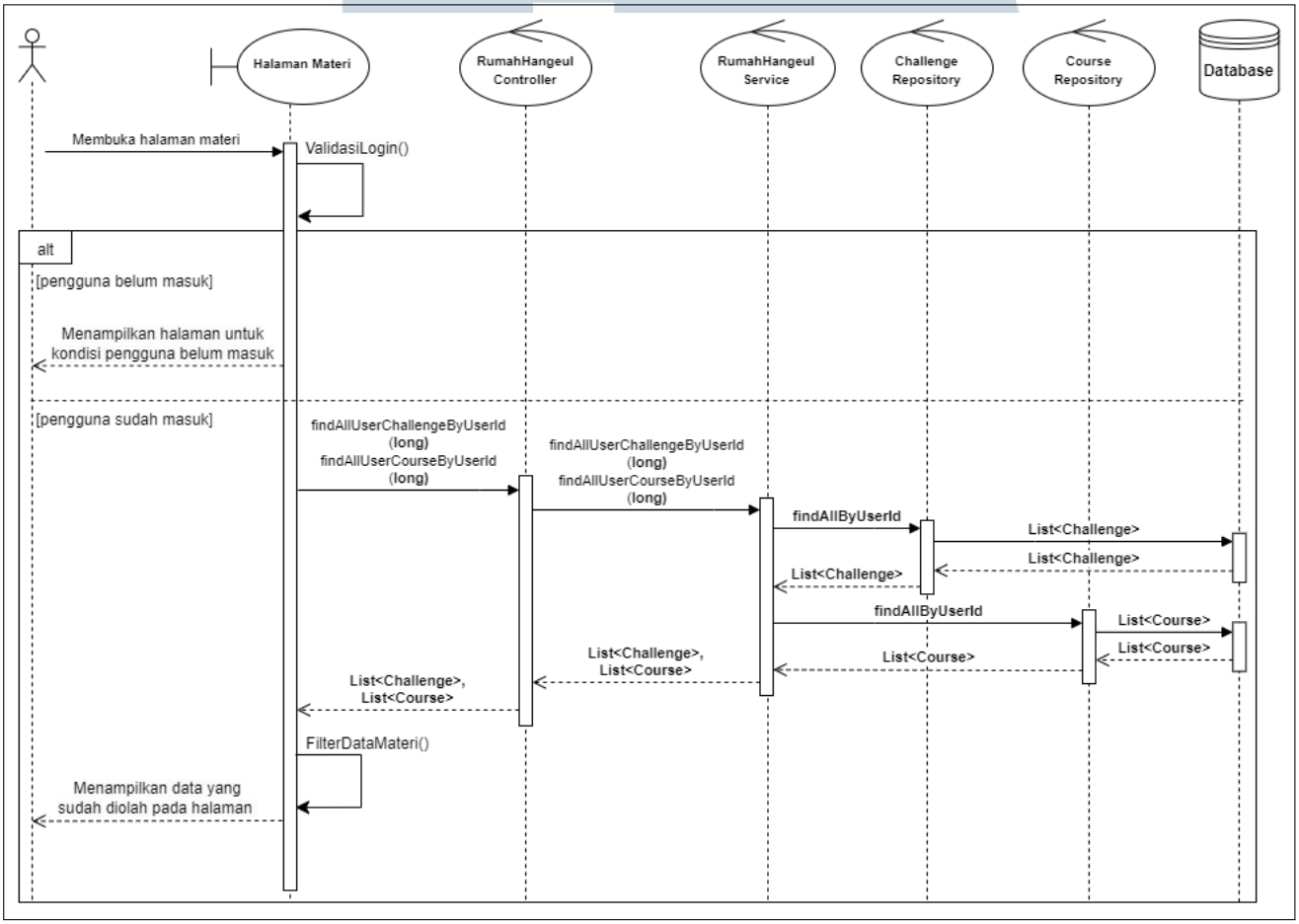
Pada *sequence diagram* halaman beranda. Sistem akan melakukan validasiLogin() untuk mengecek apakah pelajar sudah masuk atau belum. Kemudian sistem akan menampilkan halaman untuk kondisi pengguna belum masuk atau untuk kondisi pengguna sudah masuk. *Sequence diagram* halaman beranda ditunjukkan pada Gambar 3.17.



Gambar 3.17. Sequence Diagram Halaman Beranda

Pada *sequence diagram* halaman materi. Sistem akan melakukan validasiLogin() untuk mengecek apakah pelajar sudah masuk atau belum. Jika pelajar belum masuk maka sistem akan menampilkan halaman untuk kondisi pengguna belum masuk. Jika pelajar sudah masuk maka sistem akan mengirim *request* findAllUserCourseByUserId(long) dan findAllUserChallengeByUserId(long) dengan parameter *long* yang berupa *id user*. Kemudian *controller* akan memanggil *service* untuk menjalakan *function* findAllUserCourseByUserId(long) dan findAllUserChallengeByUserId(long). Setelah itu *service* akan memanggil *repository course* dan *challenge* dan

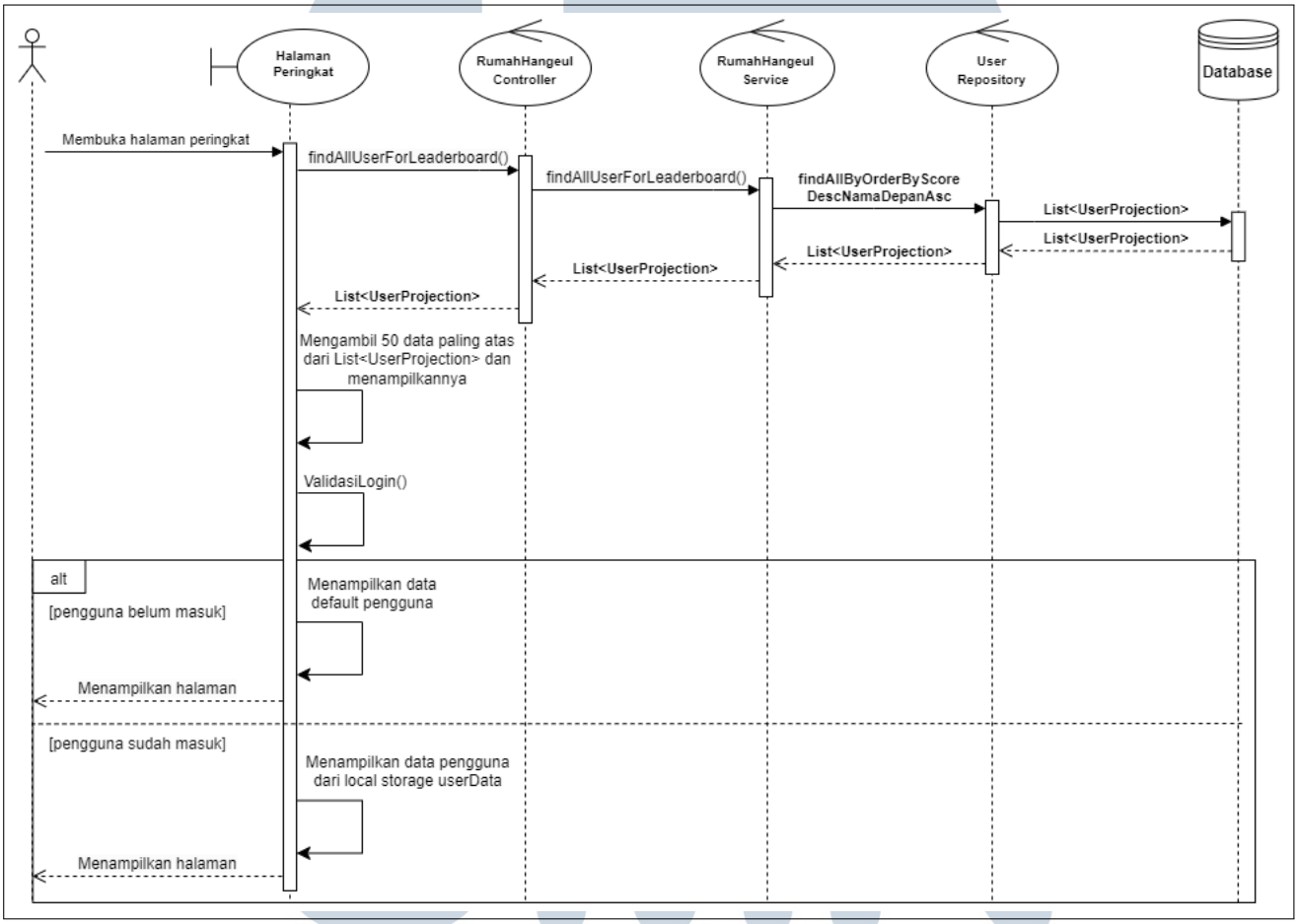
menjalankan *method query* findAllByUserId untuk mengambil data *list course* dan *list challenge* dari *database* dan mengembalikannya. Setelah itu sistem akan melakukan filterDataMateri() dengan cara menghapus data *default* pada *list challenge* dan *course* jika terdapat data yang sama dengan data yang dikembalikan saat mengirimkan *request*. Lalu sistem akan menampilkan halaman dengan data yang sudah diolah tersebut pada halaman materi. *Sequence diagram* halaman materi ditunjukkan pada Gambar 3.18.



Gambar 3.18. Sequence Diagram Halaman Materi

Pada *sequence diagram* halaman peringkat. Sistem akan mengirim *request* findAllUserForLeaderboard(). Kemudian *controller* akan memanggil *service* untuk menjalakan *function* findAllUserForLeaderboard(). Setelah itu *service* akan memanggil *repository user* dan menjalankan *method query* findAllByOrderByScoreDescNamaDepanAsc untuk mengambil data *user projection* dari *database* dan mengembalikannya. Lalu sistem akan mengambil 50 data paling atas dari data yang dikembalikan saat mengirimkan *request*. Lalu

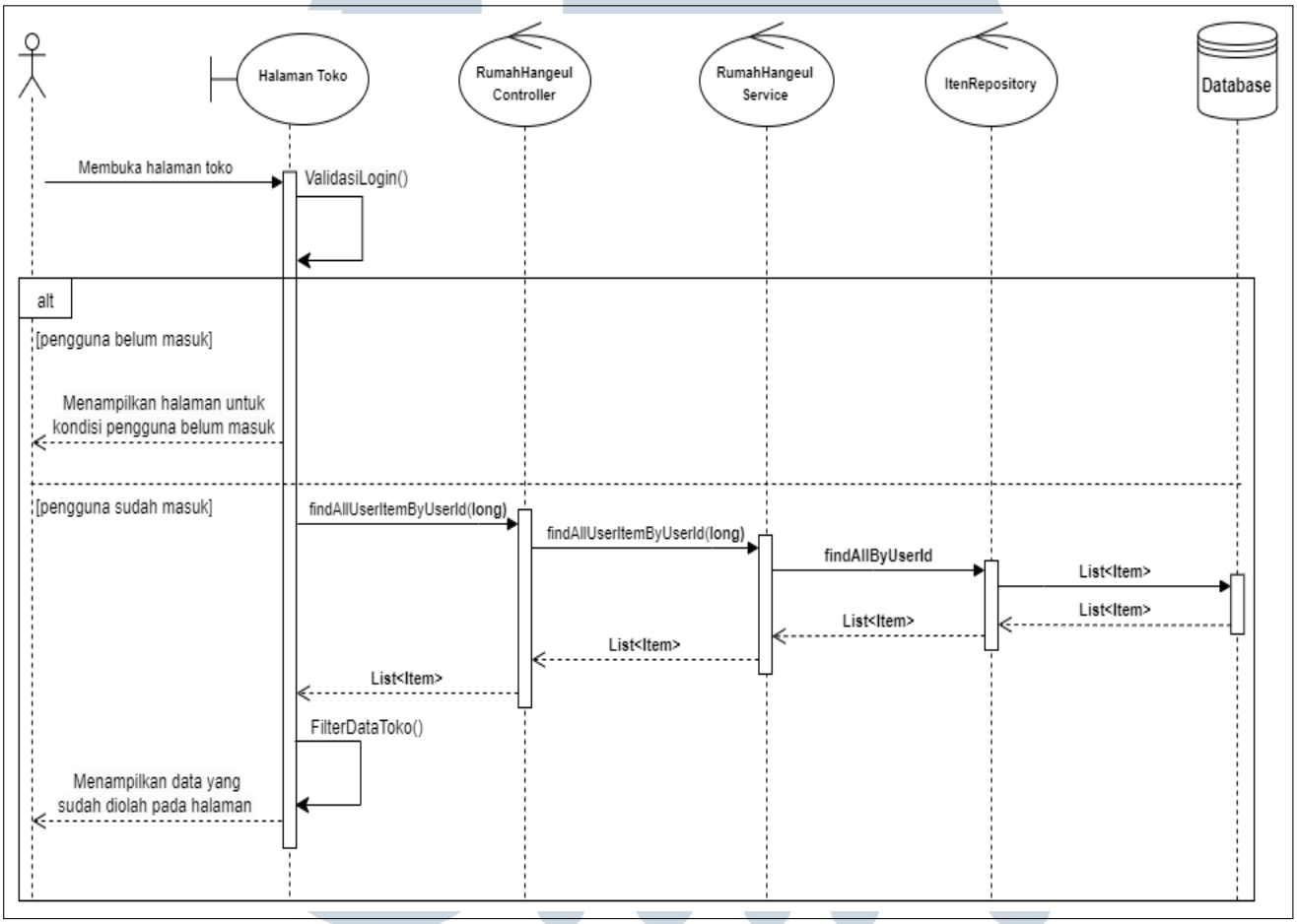
sistem akan melakukan validasiLogin() untuk mengecek apakah pelajar sudah masuk atau belum jika belum maka akan sistem akan menampilkan data *default* berupa ”Anda belum Masuk” sebagai nama depan dan nilai 0 untuk skor pada *display* pengguna. Jika sudah maka sistem akan menampilkan nama depan dan skor sesuai dengan data yang didapatkan dari *local storage* pada *display* pengguna. *Sequence diagram* halaman peringkat ditunjukkan pada Gambar 3.19.



Gambar 3.19. Sequence Diagram Halaman Peringkat

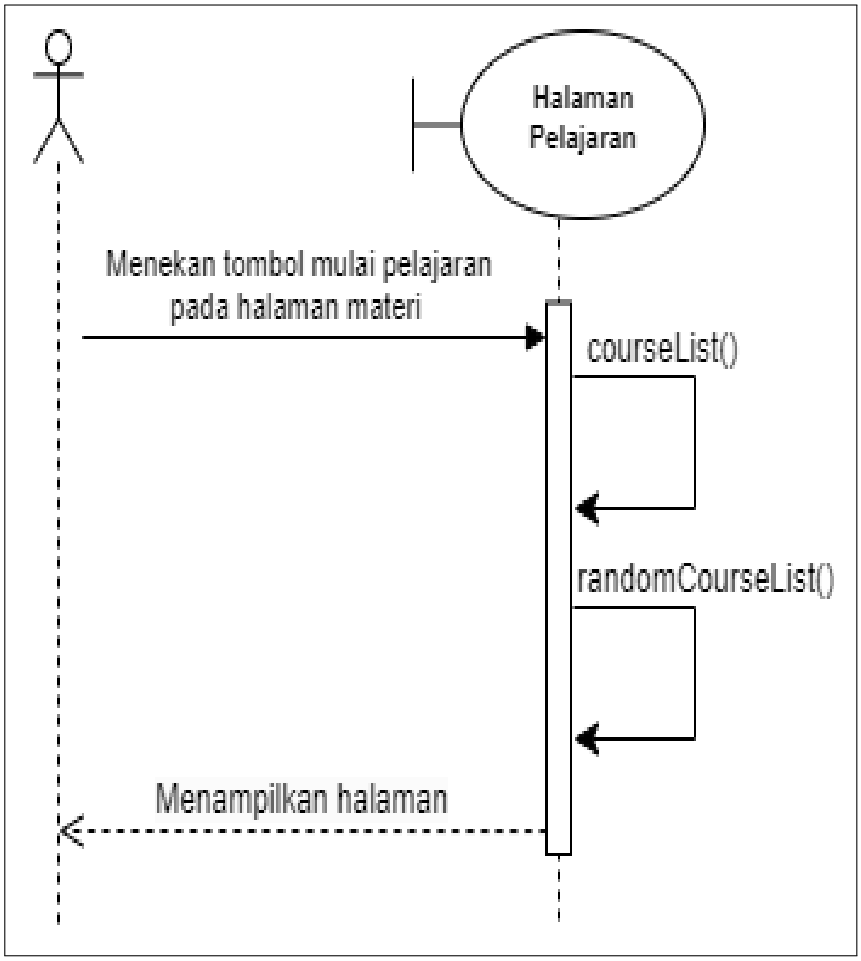
Pada *sequence diagram* halaman toko. Sistem akan melakukan validasiLogin() untuk mengecek apakah pelajar sudah masuk atau belum. j Jika pelajar belum masuk maka sistem akan menampilkan halaman untuk kondisi pengguna belum masuk. Jika sudah maka sistem akan mengirim *request* findAllUserItemByUserId(long) dengan parameter *long* yang berupa *id user*. Kemudian *controller* akan memanggil *service* untuk menjalakan *function* findAllUserItemByUserId(long). Setelah itu *service* akan memanggil *repository item* dan menjalankan *method query* findAllByUserId untuk mengambil data *list*

item dari *database* dan mengembalikannya. Setelah itu sistem akan melakukan filterDataToko() dengan menghapus data *default* pada *list* barang yang dijual pada halaman toko jika terdapat data yang sama dengan data yang dikembalikan saat mengirimkan *request*. Kemudian sistem akan menampilkan data yang sudah diolah pada halaman toko untuk pengguna yang sudah masuk. *Sequence diagram* halaman toko ditunjukkan pada Gambar 3.20.



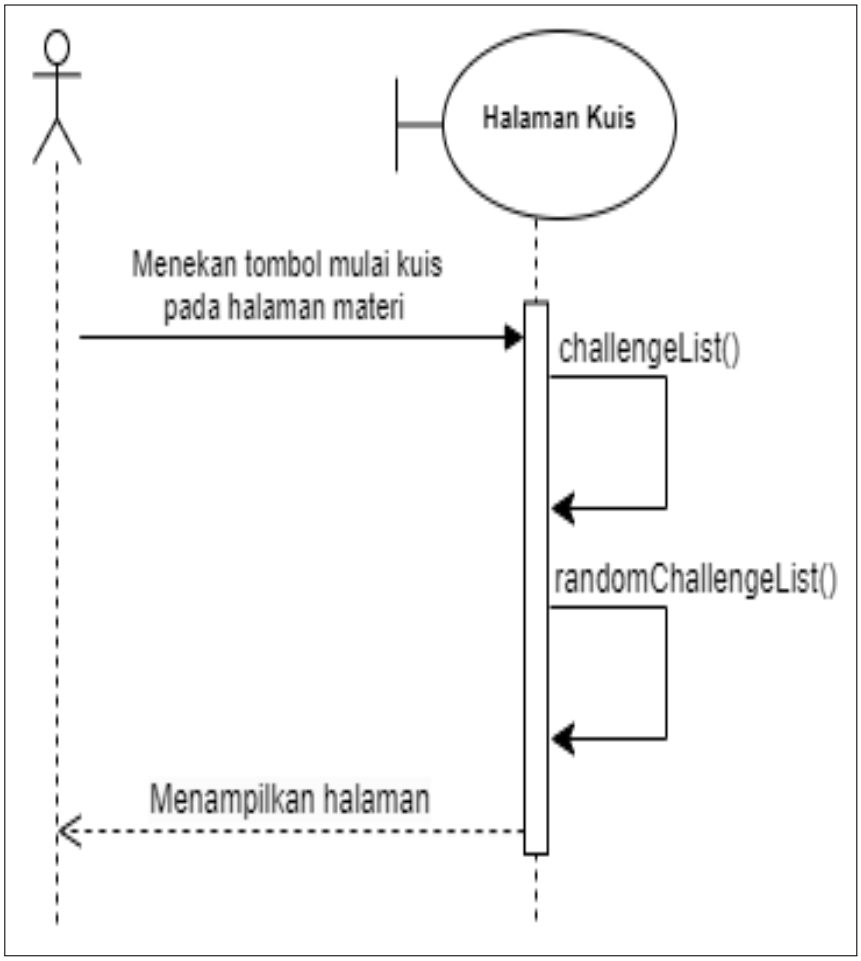
Gambar 3.20. Sequence Diagram Halaman Toko

Pada *sequence diagram* halaman pelajaran. Sistem akan melakukan courseList() untuk mengambil materi pelajaran yang berkaitan di *file constant*. Lalu sistem akan melakukan randomCourseList() untuk mengacak urutan materi pelajaran dan menampilkan halaman pelajaran. *Sequence diagram* halaman pelajaran ditunjukkan pada Gambar 3.21.



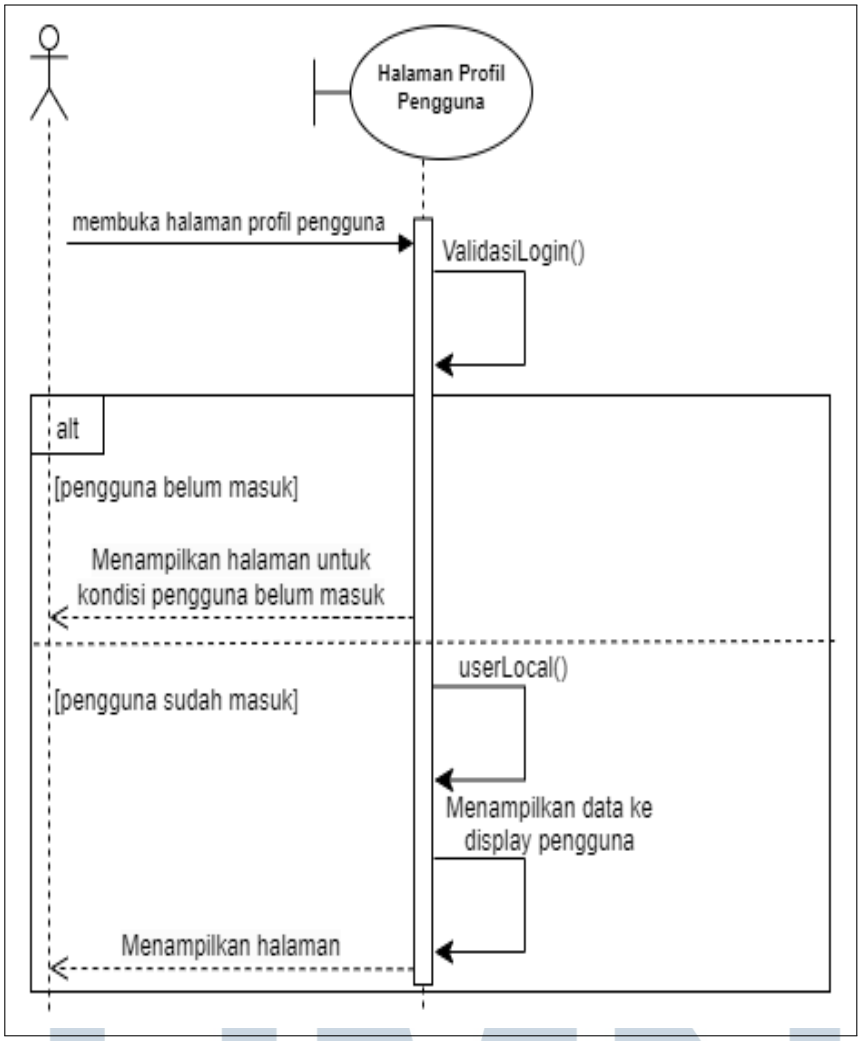
Gambar 3.21. Sequence Diagram Halaman Pelajaran

Pada *sequence diagram* halaman kuis. Sistem akan melakukan challengeList() untuk mengambil soal kuis yang berkaitan di *file constant*. Lalu sistem akan melakukan randomChallengeList() untuk mengacak urutan soal kuis dan menampilkan halaman kuis. *Sequence diagram* halaman kuis ditunjukkan pada Gambar 3.22.



Gambar 3.22. Sequence Diagram Halaman Kuis

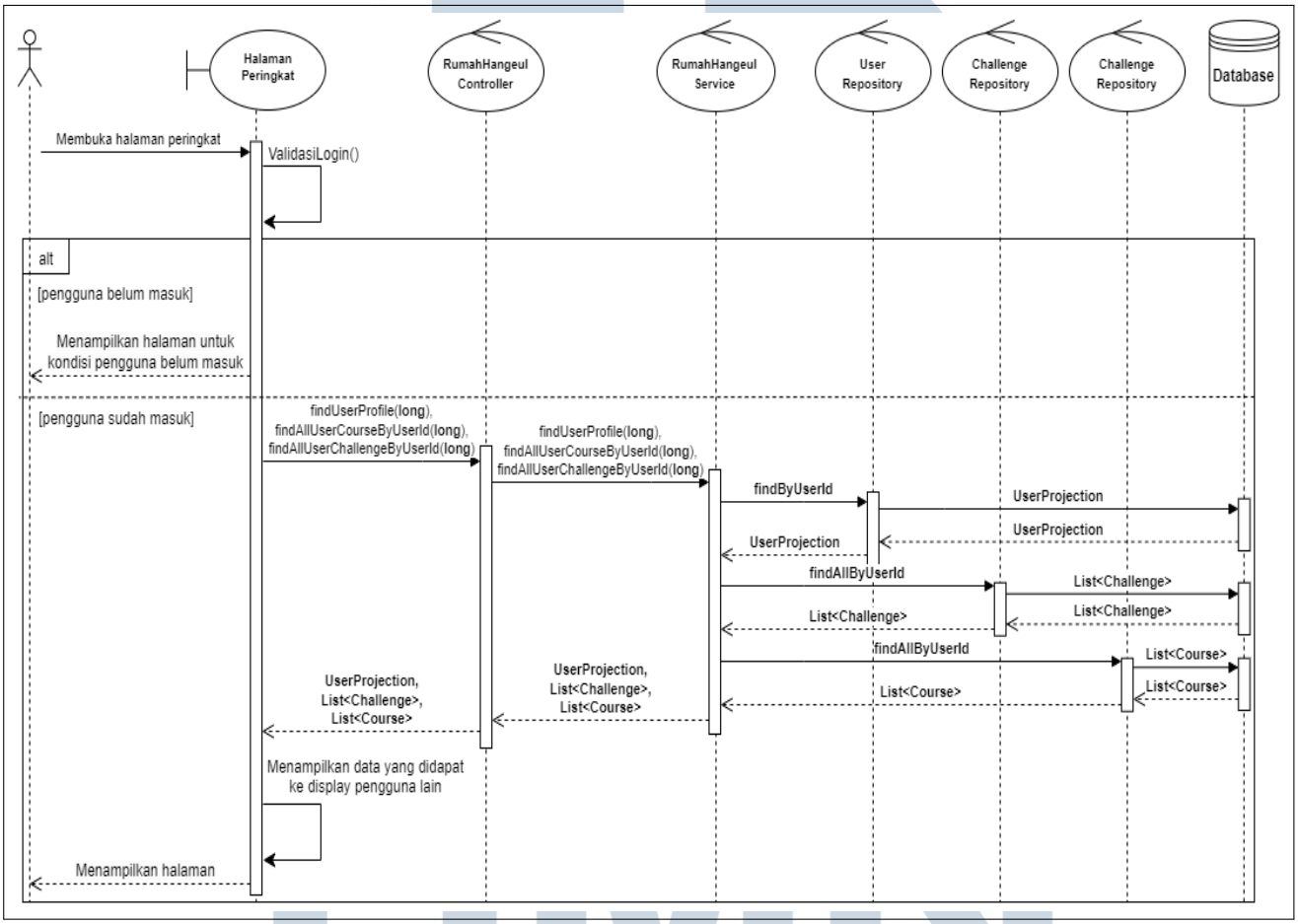
Pada *sequence diagram* halaman profil pengguna. Sistem akan melakukan validasiLogin() untuk mengecek apakah pelajar sudah masuk atau belum. Jika pelajar belum masuk maka sistem akan menampilkan halaman untuk kondisi pengguna belum masuk. Jika sudah maka sistem akan menjalankan userLocal() untuk mengambil userData dari *local storage* dan menampilkannya ke *display* pengguna. Lalu sistem akan menampilkan halaman profil pengguna untuk pengguna yang sudah masuk. *Sequence diagram* halaman profil pengguna ditunjukkan pada Gambar 3.23.



Gambar 3.23. Sequence Diagram Halaman Profil Pengguna

Pada *sequence diagram* halaman profil pengguna lain. Sistem akan melakukan validasiLogin() untuk mengecek apakah pelajar sudah masuk atau belum. Jika pelajar belum masuk maka sistem akan menampilkan halaman untuk kondisi pengguna belum masuk. Jika pelajar sudah masuk maka sistem akan mengirim *request* , findUserProfile(long), findAllUserCourseByUserId(long) dan findAllUserChallengeByUserId(long) dengan parameter *long* yang berupa *id user*. Kemudian *controller* akan memanggil *service* untuk menjalakan *function* findUserProfile(long), findAllUserCourseByUserId(long) dan findAllUserChallengeByUserId(long). Setelah itu *service* akan memanggil *repository user, course* dan *challenge* dan menjalankan *method query* findByUserId untuk *user* serta findAllByUserId untuk *course* dan *challenge* untuk mengambil data *user projection, list course* dan *list challenge* dari *database*

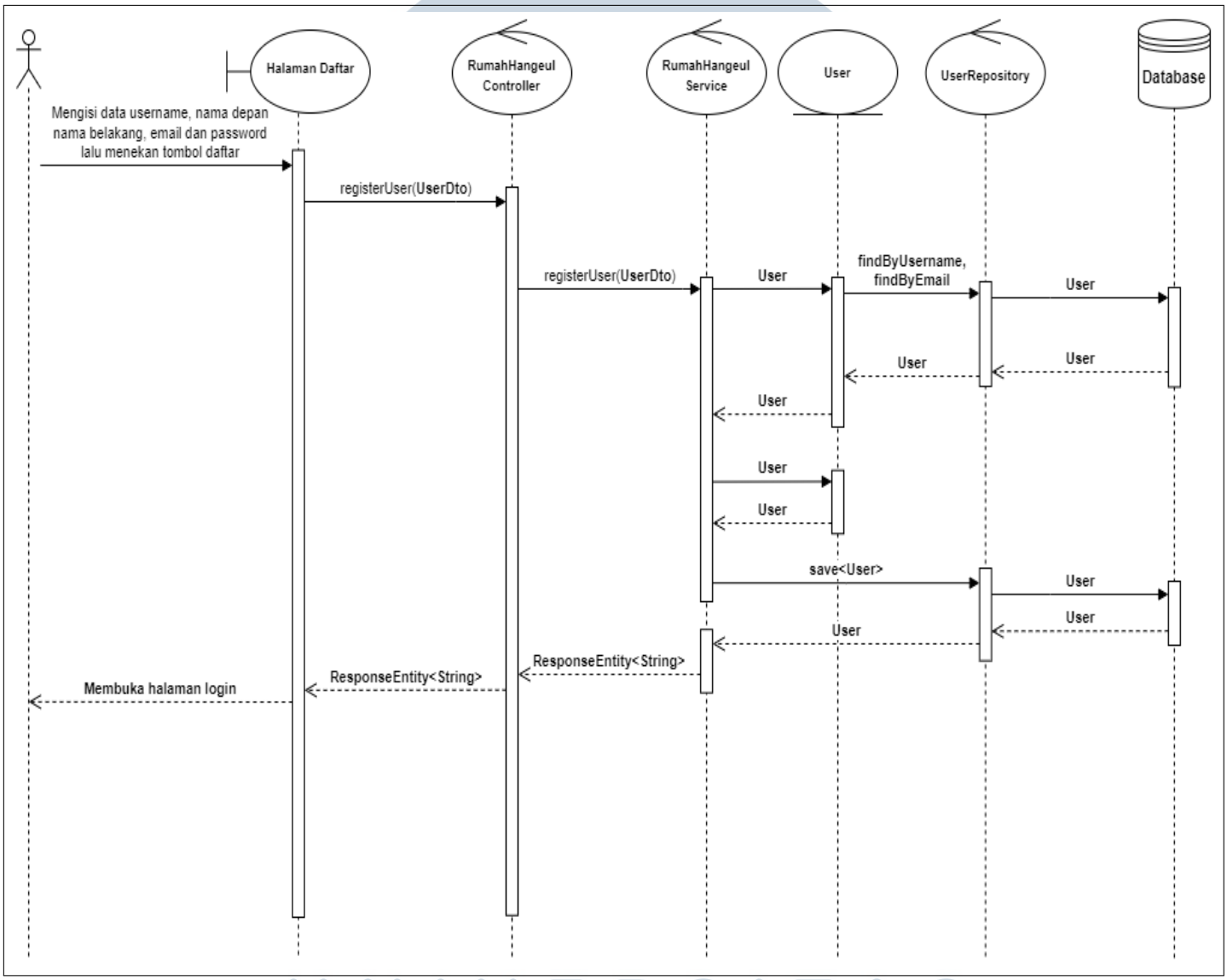
dan mengembalikannya. Setelah itu sistem akan menampilkan *display* pengguna sesuai dengan data yang dikembalikan saat mengirimkan *request*. Lalu sistem akan menampilkan halaman profil pengguna lain. *Sequence diagram* halaman profil pengguna lain ditunjukkan pada Gambar 3.24.



Gambar 3.24. Sequence Diagram Halaman Profil Pengguna Lain

Pada *sequence diagram register*, *controller* akan menerima *request* registerUser dengan *parameter* UserDto dan menjalankan *function* registerUser(UserDto) dengan memanggil *service*. Setelah itu *service* akan menjalankan *function* registerUser(UserDto). Lalu *service* akan mendeklarasi *optional user* baru dan memanggil *user repository* yang akan mencari data *user* di *database* berdasarkan *username* dan email yang diterima pada UserDto. Jika ada maka akan dikembalikan *response error* dengan pesan bahwa *username* atau email sudah digunakan. Jika tidak maka *service* akan mendeklarasi *user* baru dan mengisi datanya dengan data UserDto dan memanggil *repository* untuk menyimpan data tersebut ke database. Setelah itu *service* akan mengembalikan

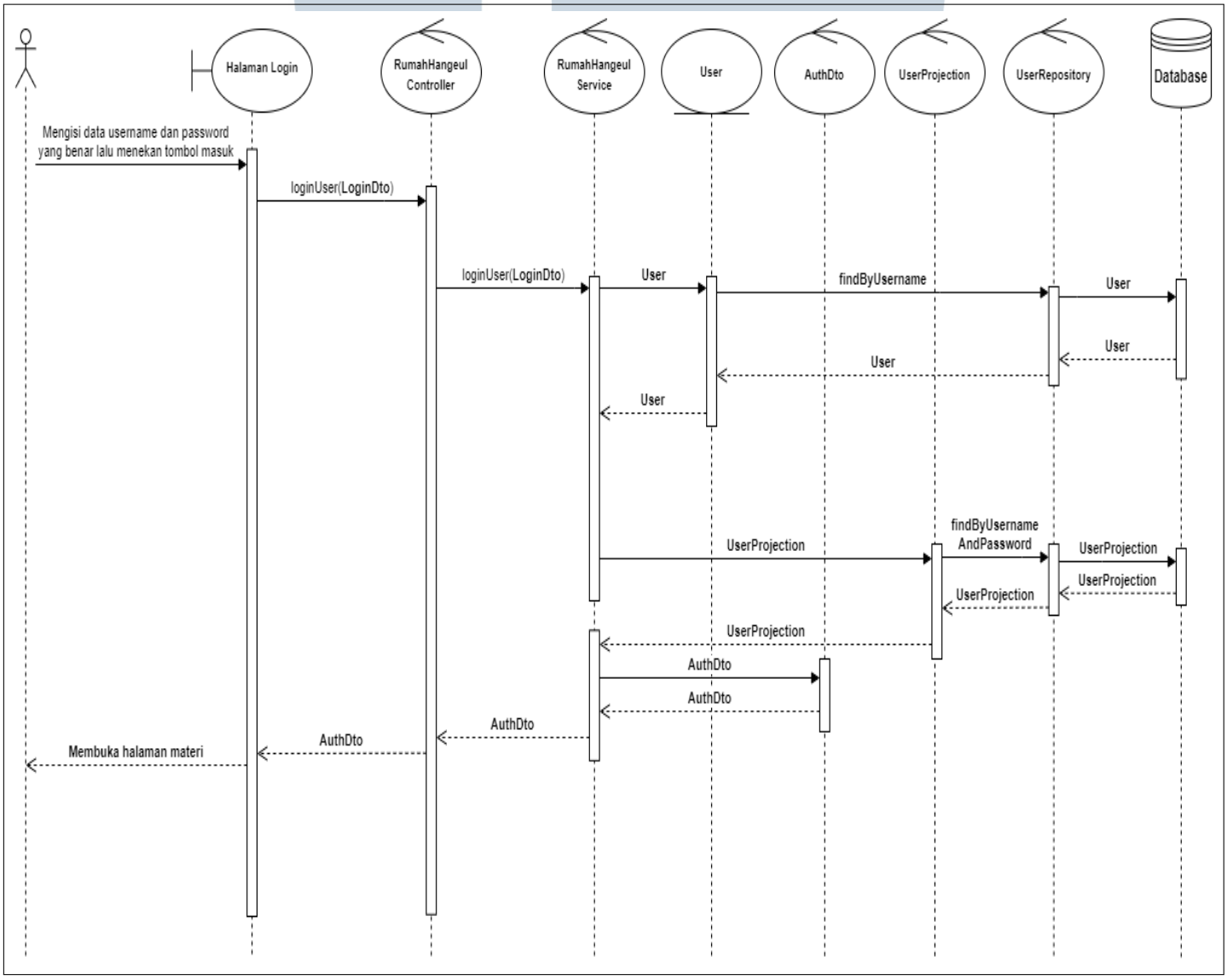
response success dengan pesan bahwa pengguna berhasil didaftarkan. Kemudian sistem akan membuka halaman *login* setelah pelajar berhasil melakukan *register*. *Sequence diagram register* ditunjukkan pada Gambar 3.25.



Gambar 3.25. Sequence Diagram Register

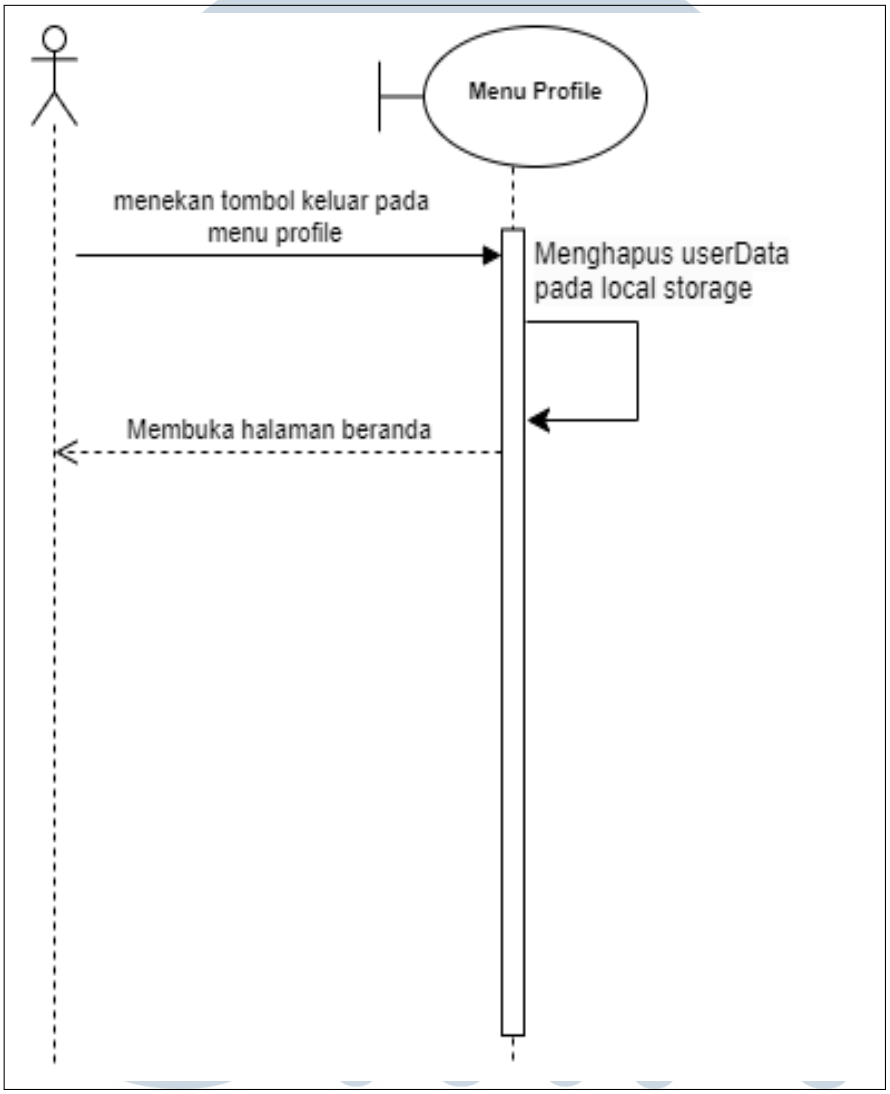
Pada *sequence diagram login*, *controller* akan menerima *request* loginUser dengan *parameter* LoginDto dan menjalankan *function* loginUser(LoginDto) dengan memanggil *service*. Setelah itu *service* akan menjalankan *function* loginUser(LoginDto). Lalu *service* akan mendeklarasi *optional user* baru dan memanggil *user repository* yang akan mencari data *user* di *database* berdasarkan *username* yang diterima pada LoginDto. Jika tidak ada maka *service* akan

mengembalikan *response error* dengan pesan *username* atau *password* dimasukkan salah. Jika terdapat maka data tersebut akan diambil lalu akan dilakukan pencocokan data *password* LoginDto dengan data dari *database*. Jika tidak cocok maka *service* akan mengembalikan *response error* dengan pesan *username* atau *password* dimasukkan salah. Jika cocok maka *service* akan mendeklarasi User Projection dan memanggil *repository* untuk mendapatkan data di *database* berdasarkan data dari LoginDto. Setelah itu *service* akan membuat AuthDto *builder* dan memasukan data User Projection yang didapat lalu akan dikembalikan ke halaman *login*. Kemudian sistem akan membuka halaman materi setelah pelajar berhasil melakukan *login*. *Sequence diagram login* ditunjukkan pada Gambar 3.26.



Gambar 3.26. Sequence Diagram Login

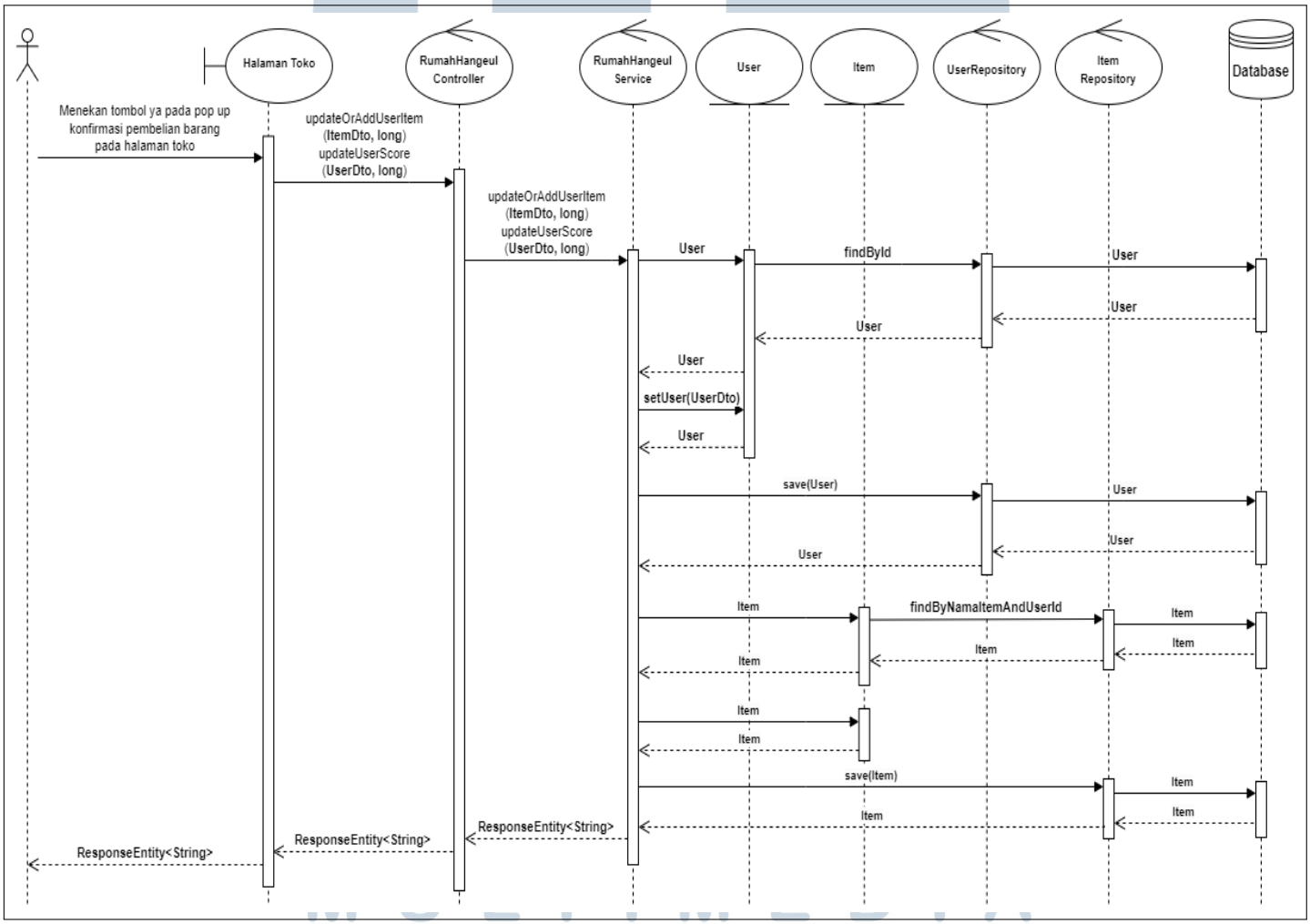
Pada *sequence diagram* logout. Sistem akan menghapus userData pada *local storage*. Kemudian sistem akan membuka halaman beranda. *Sequence diagram* logout ditunjukkan pada Gambar 3.27.



Gambar 3.27. Sequence Diagram Logout

Pada *sequence diagram* beli barang, *controller* akan menerima *request* updateOrAddUserItem dengan *parameter* ItemDto serta *long* yang merupakan *id user* dan menjalankan *function* updateOrAddUserItem(ItemDto, long) dengan memanggil *service*. Setelah itu *service* akan menjalankan *function* updateOrAddUserItem(ItemDto, long). Lalu *service* akan memanggil *repository* dan menjalankan *method query* findById untuk mengecek apakah terdapat pengguna dengan parameter yang dikirim pada *database*. Jika tidak ada maka akan

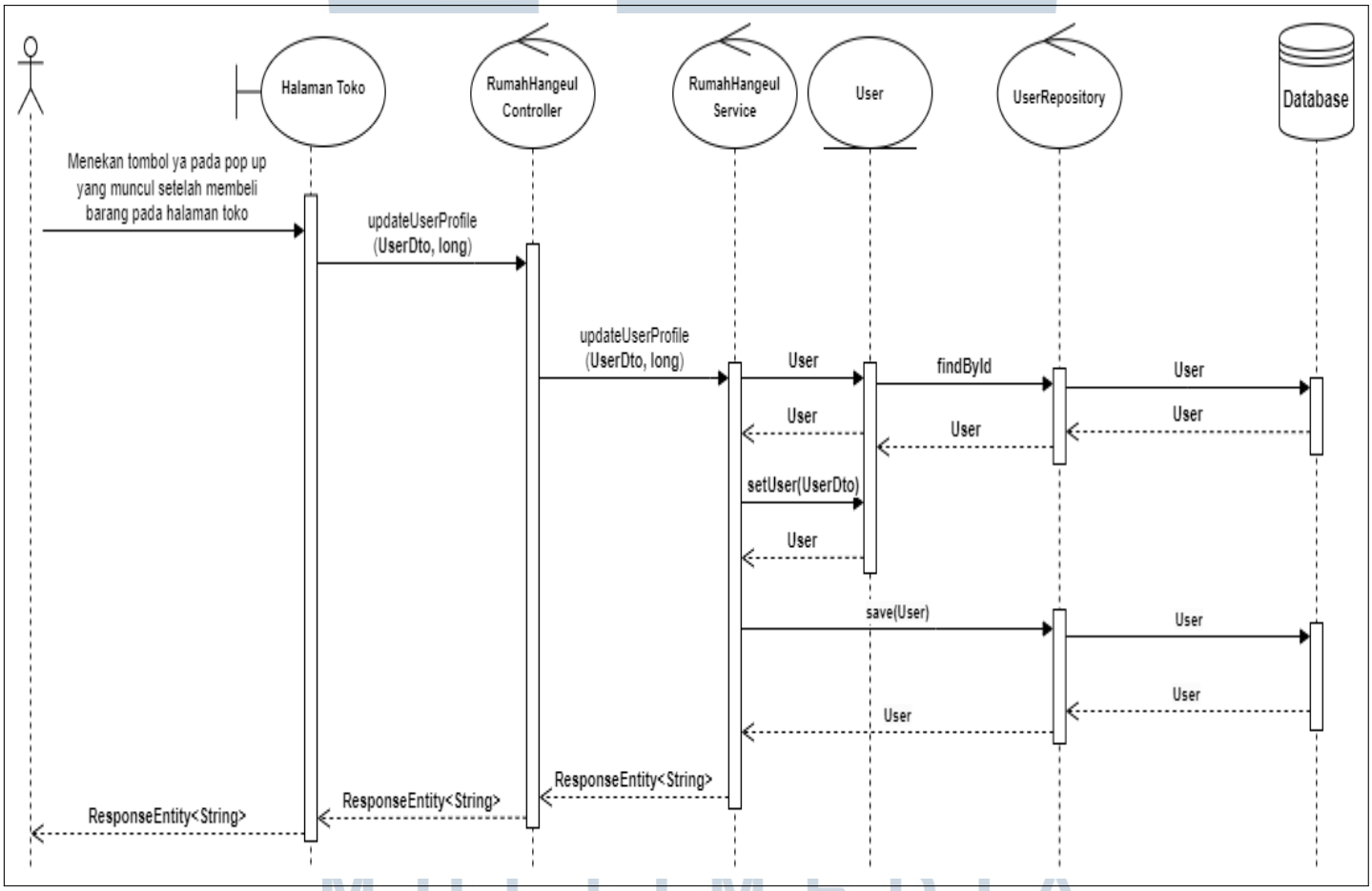
dikembalikan *response error* dengan pesan bahwa pengguna tidak ditemukan. Jika ada maka *service* akan melakukan hal yang sama untuk mengecek apakah sudah terdapat *item* dengan nama *item* dan *id user* yang ada pada ItemDto di *database*. Jika ada maka akan dikembalikan *response ok* dengan pesan yang ditentukan. Jika tidak ada maka *service* akan mendeklarasi *item* baru dan mengisi datanya dengan data ItemDto dan memanggil *repository* untuk menyimpan data tersebut ke *database*. Setelah itu *service* akan mengembalikan *response success* dengan pesan bahwa barang berhasil dibeli. *Sequence diagram* beli barang ditunjukkan pada Gambar 3.28.



Gambar 3.28. Sequence Diagram Beli Barang

Pada *sequence diagram* pakai barang, *controller* akan menerima *request* updateUserProfile dengan *parameter* UserDto serta *long* yang merupakan *id user* dan menjalankan *function* updateUserProfile(UserDto, long) dengan memanggil

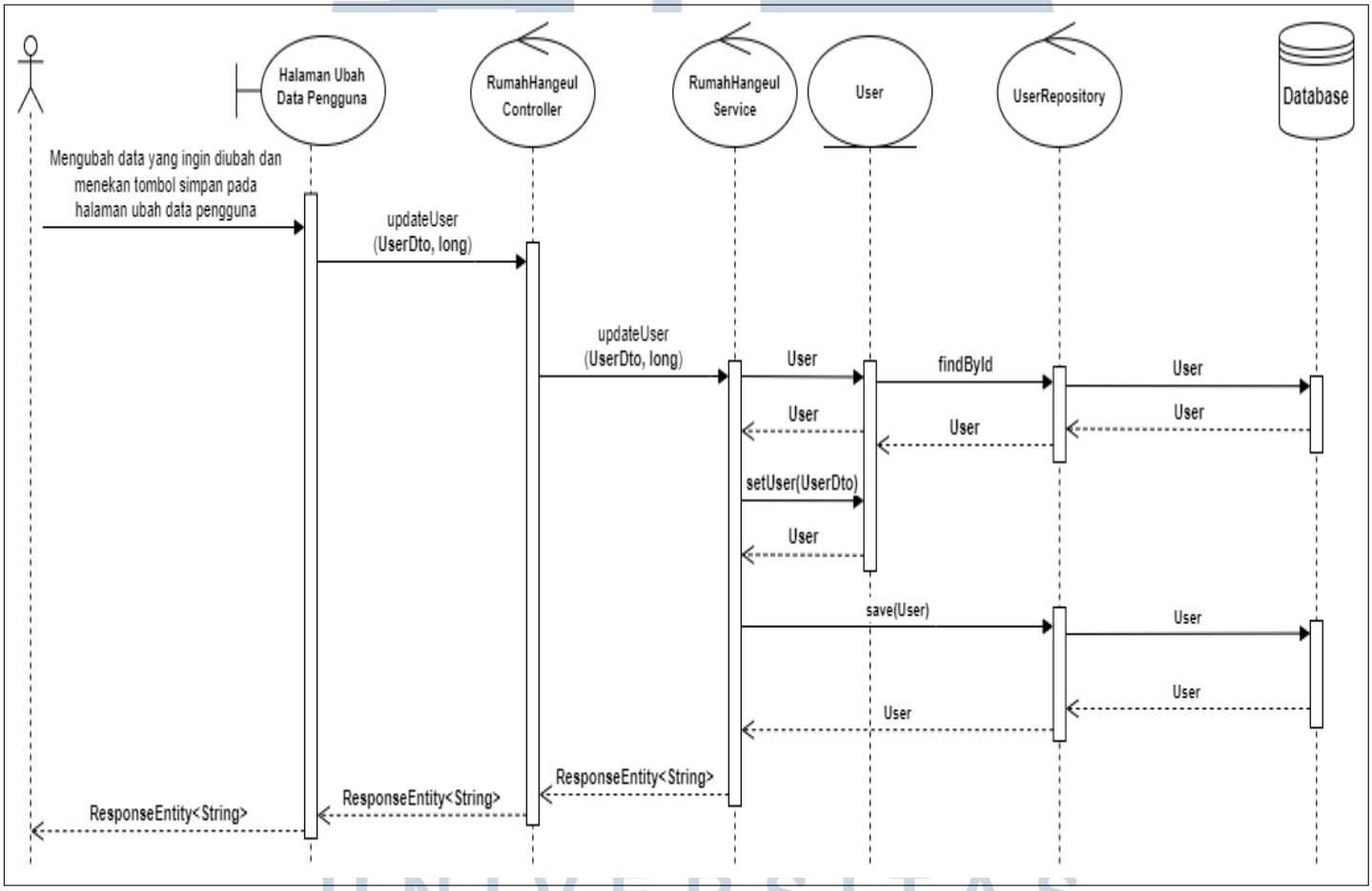
service. Setelah itu *service* akan menjalankan *function* updateUserProfile(UserDto, long). Lalu *service* akan memanggil *repository* dan menjalankan *method query* findById untuk mengecek apakah terdapat pengguna dengan parameter yang dikirim pada *database*. Jika tidak ada maka akan dikembalikan *response error* dengan pesan bahwa pengguna tidak ditemukan. Jika ada maka *service* akan melakukan *update* pada *field* yang ditetapkan dengan data dari UserDto lalu memanggil *repository* untuk menyimpan data tersebut ke *database*. Setelah itu *service* akan mengembalikan *response success* dengan pesan profil pengguna berhasil diperbarui. *Sequence diagram* pakai barang ditunjukkan pada Gambar 3.29.



Gambar 3.29. Sequence Diagram Pakai Barang

Pada *sequence diagram* ubah data pengguna, *controller* akan menerima *request* updateUser dengan *parameter* UserDto serta *long* yang merupakan *id user* dan menjalankan *function* updateUser(UserDto, long) dengan memanggil *service*. Setelah itu *service* akan menjalankan *function* updateUser(UserDto, long).

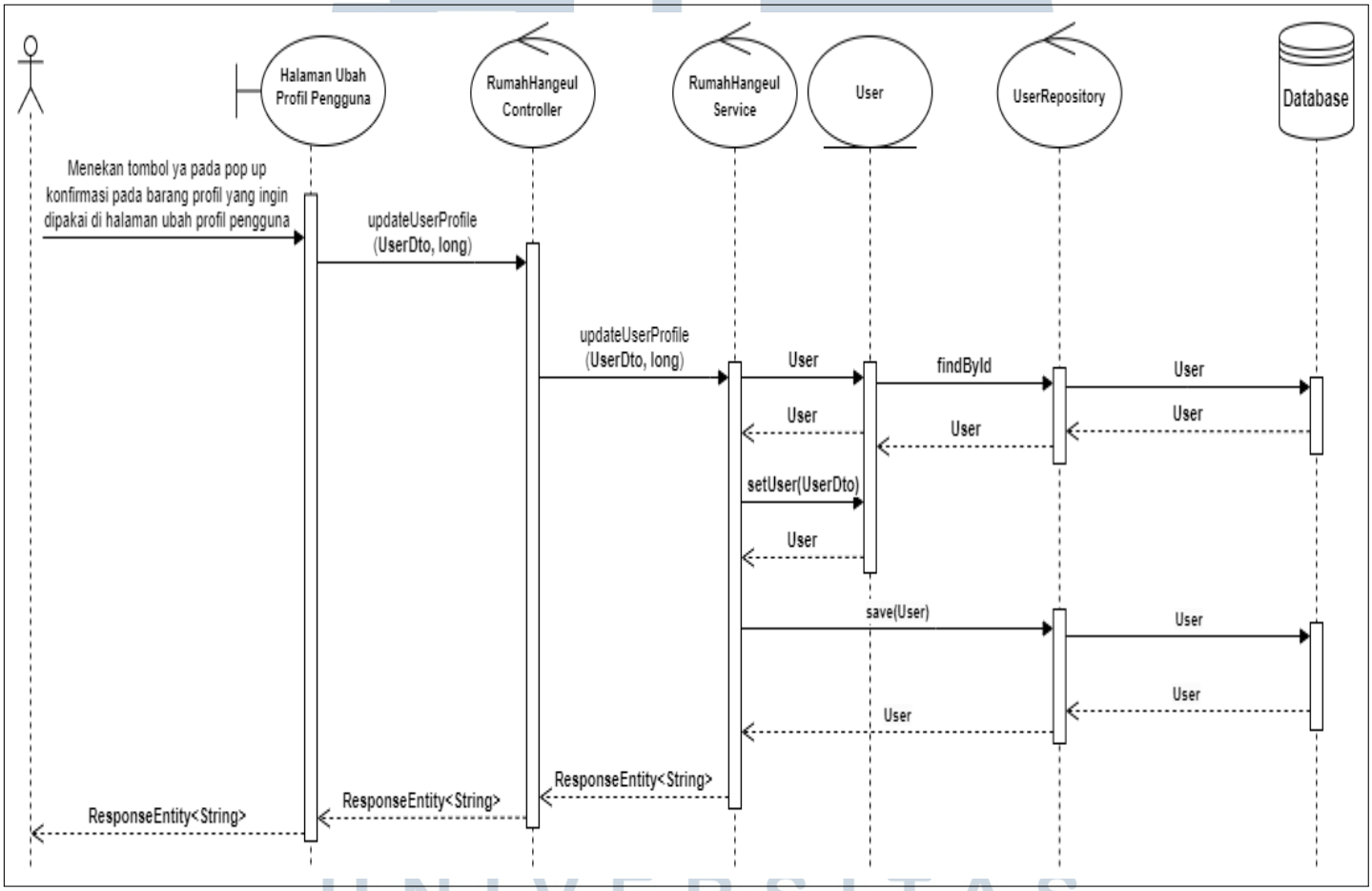
Lalu *service* akan memanggil *repository* dan menjalankan *method query* findById untuk mengecek apakah terdapat pengguna dengan parameter yang dikirim pada *database*. Jika tidak ada maka akan dikembalikan *response error* dengan pesan bahwa pengguna tidak ditemukan. Jika ada maka *service* akan melakukan *update* pada *field* yang ditetapkan dengan data dari UserDto lalu memanggil *repository* untuk menyimpan data tersebut ke *database*. Setelah itu *service* akan mengembalikan *response success* dengan pesan berhasil memperbarui pengguna. *Sequence diagram* ubah data pengguna ditunjukkan pada Gambar 3.30.



Gambar 3.30. Sequence Diagram Ubah Data Pengguna

Pada *sequence diagram* ubah profil pengguna, hal yang terjadi sama dengan *sequence diagram* pakai barang dimana *controller* akan menerima *request* updateUserProfile dengan *parameter* UserDto serta *long* yang merupakan *id user* dan menjalankan *function* updateUserProfile(UserDto, long) dengan memanggil *service*. Setelah itu *service* akan menjalankan *function* updateUserProfile(UserDto, long). Lalu *service* akan memanggil *repository* dan menjalankan *method query*

findById untuk mengecek apakah terdapat pengguna dengan parameter yang dikirim pada *database*. Jika tidak ada maka akan dikembalikan *response error* dengan pesan bahwa pengguna tidak ditemukan. Jika ada maka *service* akan melakukan *update* pada *field* yang ditetapkan dengan data dari UserDto lalu memanggil *repository* untuk menyimpan data tersebut ke *database*. Setelah itu *service* akan mengembalikan *response success* dengan pesan profil pengguna berhasil diperbarui. *Sequence diagram* ubah profil pengguna ditunjukkan pada Gambar 3.31.

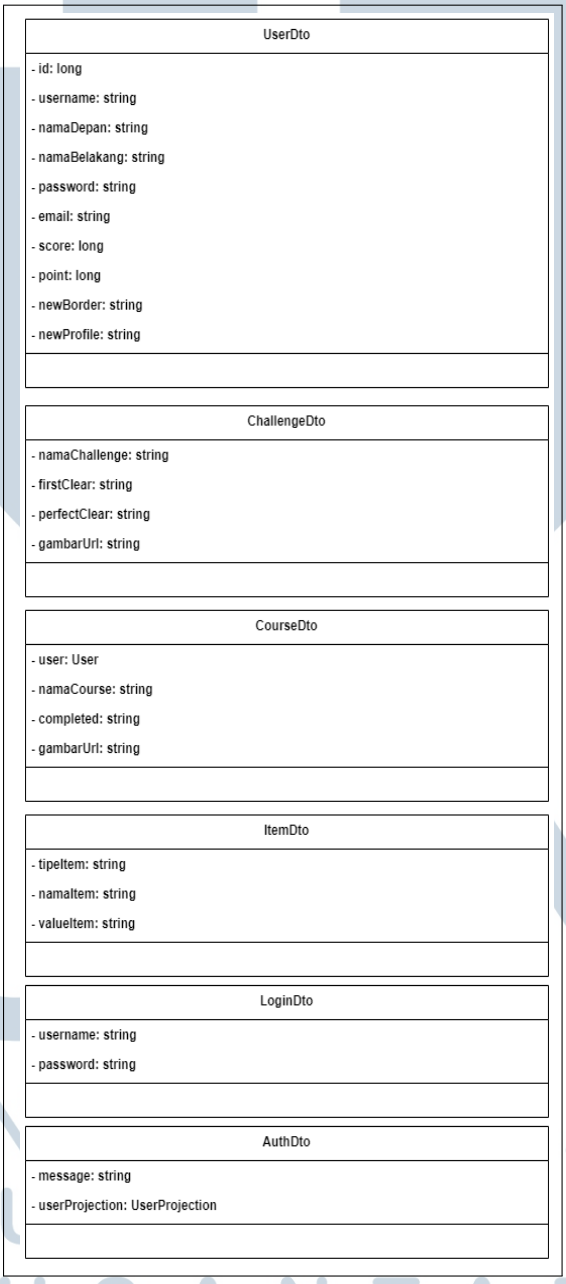


Gambar 3.31. Sequence Diagram Ubah Profil Pengguna

D *Class Diagram*

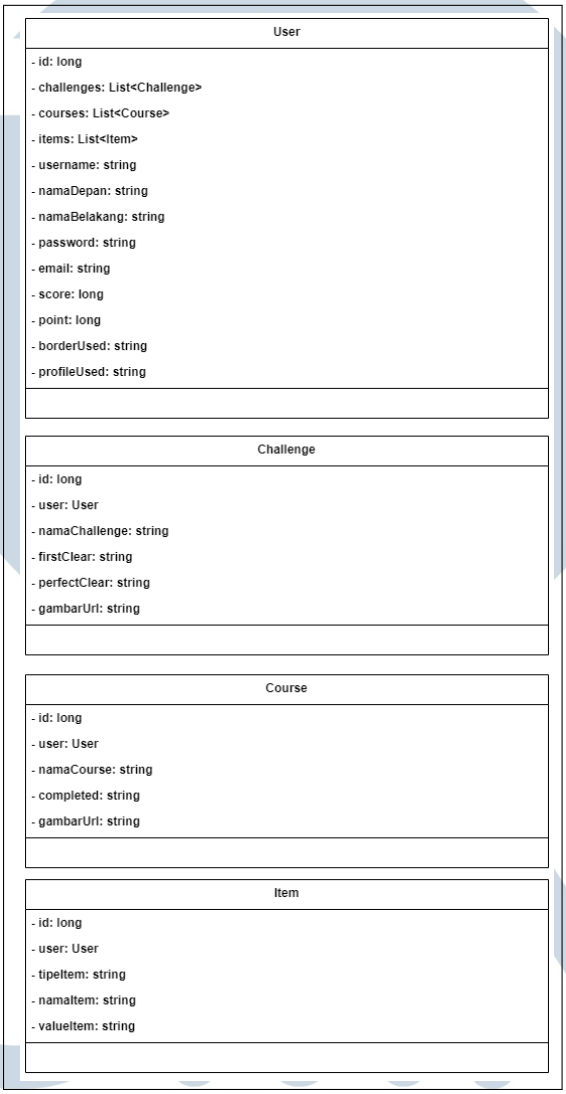
Class diagram merupakan diagram yang digunakan untuk menggambarkan dengan jelas objek yang ada. *Class diagram* menjelaskan atribut, metode, dan hubungan objek dalam suatu sistem [19].

DTO berguna sebagai objek yang menampung data yang dikirim oleh pengguna ataupun sebagai data yang akan dikirim ke pengguna. Gambar 3.32 menunjukkan *class diagram* untuk DTO UserDto, ChallengeDto, CourseDto, ItemDto, LoginDto dan AuthDto.



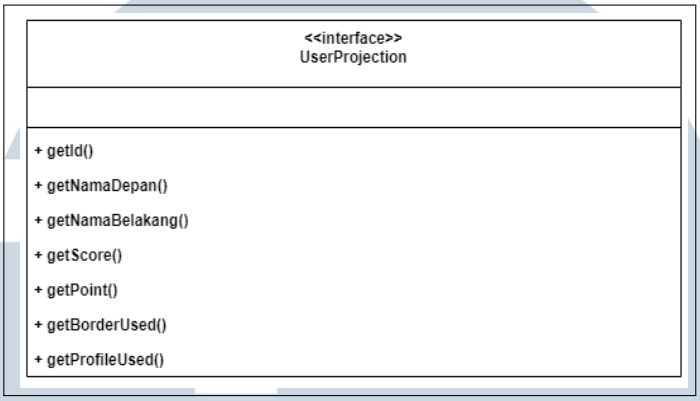
Gambar 3.32. Class Diagram DTO Aplikasi Rumah Hangeul

Entity pada aplikasi Rumah Hangeul mempunyai atribut dengan tipe datanya yang beragam. Gambar 3.34 menunjukkan *class diagram* untuk *entity* User, Challenge, Course, dan Item.



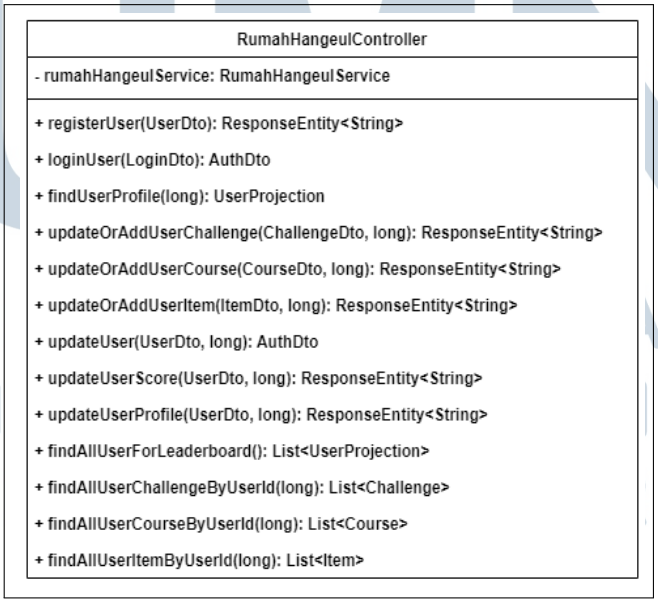
Gambar 3.33. Class Diagram Entity Aplikasi Rumah Hangeul

Interface pada aplikasi Rumah Hangeul digunakan untuk menghasilkan proyeksi data yang akan ditampilkan pada pengguna. Gambar 3.34 menunjukkan *class diagram* untuk *interface* UserProjection.



Gambar 3.34. Class Diagram Interface Aplikasi Rumah Hangeul

Controller berguna sebagai perantara API dan program untuk mengirimkan request melalui *path* URL yang ditentukan dengan *method* yang dipanggil dari *service*. *Controller* pada aplikasi Rumah Hangeul memiliki *method* untuk registeruser, loginUser, findUserProfile, findAllUsersForLeaderboard, updateUser, updateUserScore, updateUserProfile. Terdapat juga *method* updateOrAddUserItems dan findUserItemByUserId. Entity Course dan Challenge juga memiliki kedua *method* tersebut.

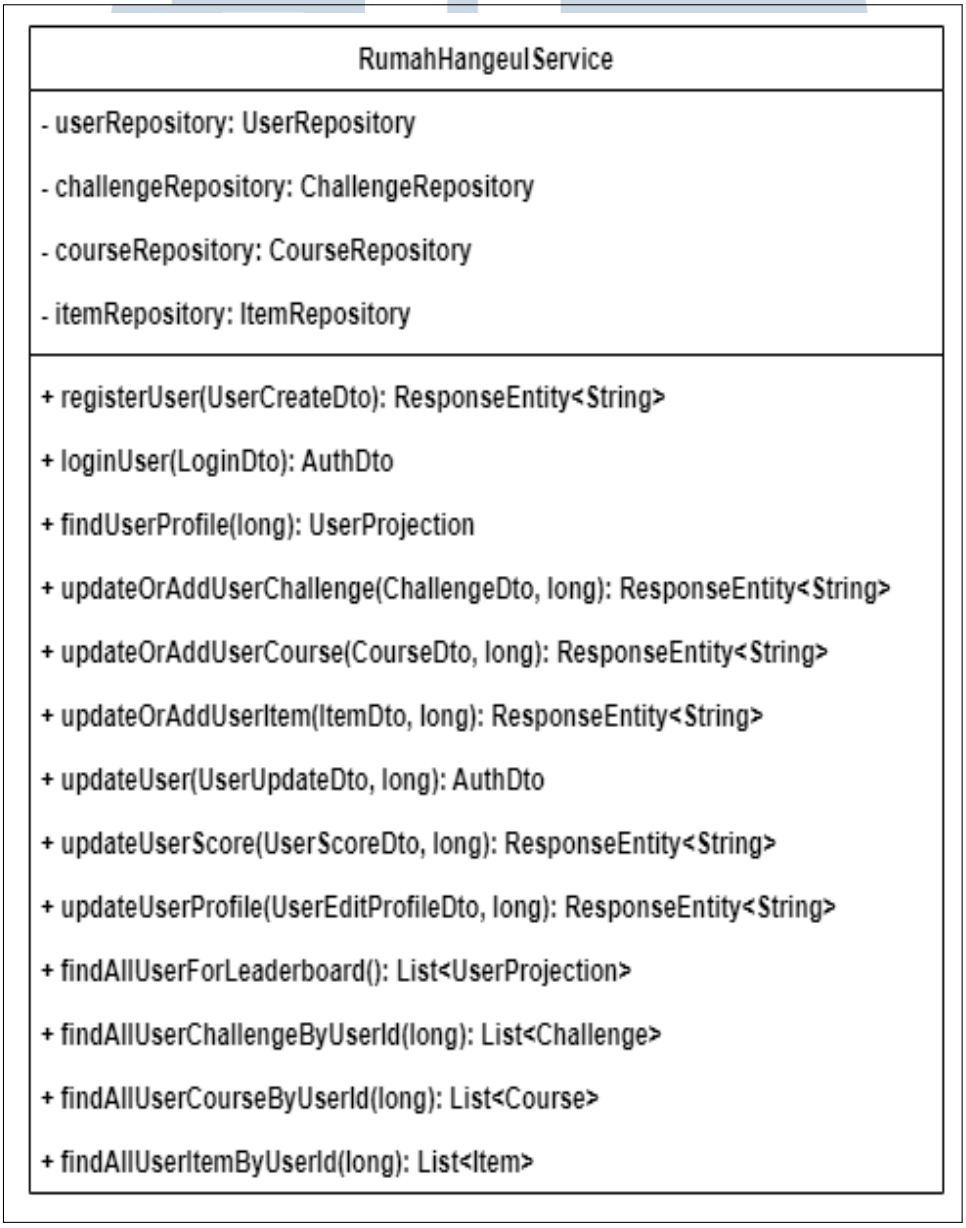


Gambar 3.35. Class Diagram Controller Aplikasi Rumah Hangeul

Repository pada aplikasi Rumah Hangeul digunakan untuk melakukan *method query database* dan melakukan komunikasi antara sistem dengan *database* yang dipanggil melalui *service*. *Method* yang terdapat pada *repository* yaitu *method save* untuk menyimpan objek pada *database* dan *method find* untuk mencari data dari *database*. Gambar 3.37 menunjukkan *class diagram* untuk RumahHangeulRepository.

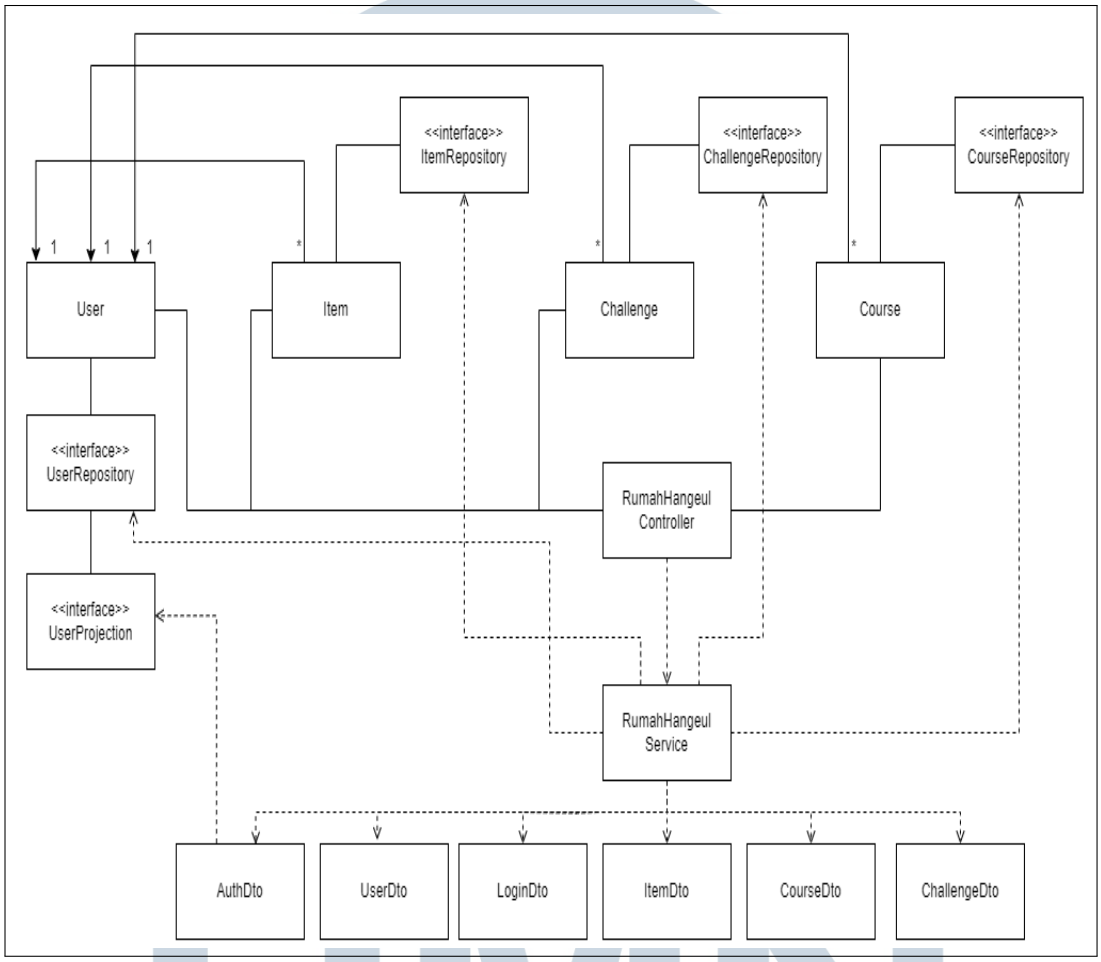
Gambar 3.36. Class Diagram Repository Aplikasi Rumah Hangeul

Service berguna untuk menjalankan *method* yang akan dipanggil dan akan dikembalikan melalui *controller* kepada pengguna. Pada *service*, diperlukan deklarasi *repository* untuk menggunakan *method query* untuk mengakses *database*. *Service* pada aplikasi Rumah Hangeul memiliki *method* untuk registeruser, authenticateUser, findByUserId, findAllUsersForLeaderboard, updateUser, updateUserScore, updateUserProfile. Terdapat juga *method* updateOrAddUserItems dan findUserItemByUserId. Entity Course dan Challenge juga memiliki kedua *method* tersebut.



Gambar 3.37. Class Diagram Service Aplikasi Rumah Hangeul

Relationship pada class diagram yang ada pada aplikasi Rumah Hangeul dapat dilihat pada Gambar 3.38.



Gambar 3.38. Relationship Class Diagram Aplikasi Rumah Hangeul

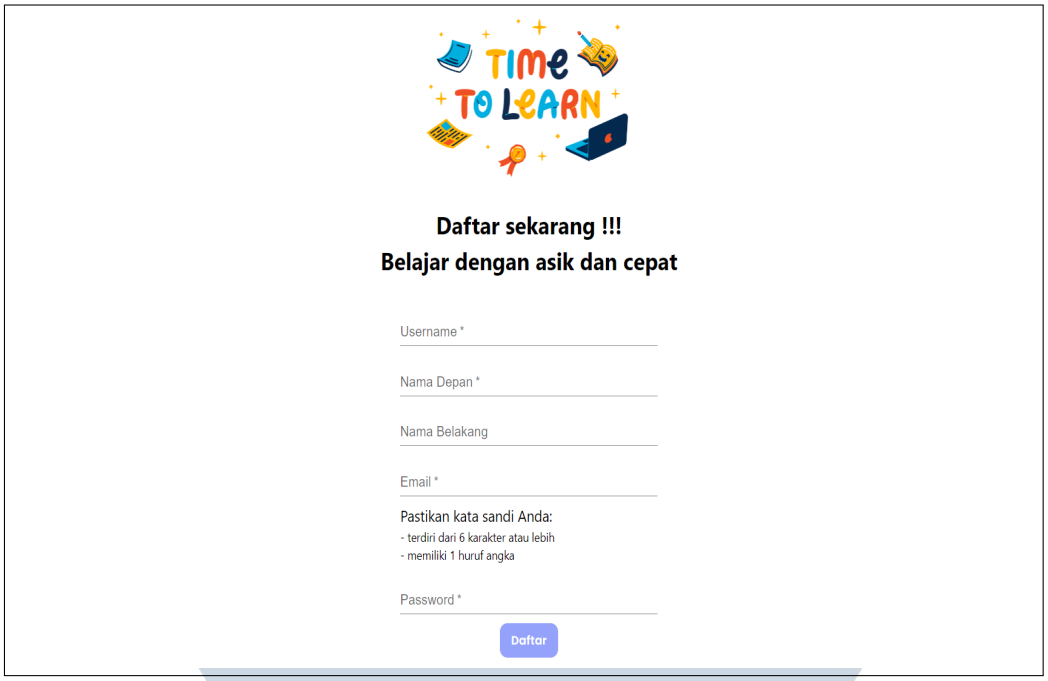
RumahHangeulController memiliki sifat *dependency* terhadap RumahHangeulService karena untuk menjalankan *method* yang ada pada *controller* perlu memanggil *service*, hal yang sama juga berlaku terhadap RumahHangeulService dengan UserRepository, ChallengeRepository, CourseRepository dan ItemRepository, AuthDto dengan UserProjection dan RumahHangeulService dengan UserDto, LoginDto, ItemDto, CourseDto, ChallengeDto dan AuthDto. Setiap *entity* dan *repository* memiliki hubungan *simple association*, hal yang sama juga berlaku terhadap semua *entity* dengan RumahHangeulController dan UserProjection dengan UserRepository. User juga memiliki hubungan *cardinality one to many* dengan Challenge, Course dan Item.

3.2.4 Desain antarmuka

Desain antarmuka halaman beranda dditampilkan pada Gambar 3.39. Pada gambar tersebut dapat dilihat terdapat kalimat sapaan pada halaman beranda dan juga terdapat tombol mulai sekarang yang akan membawa pengguna ke halaman *register*.

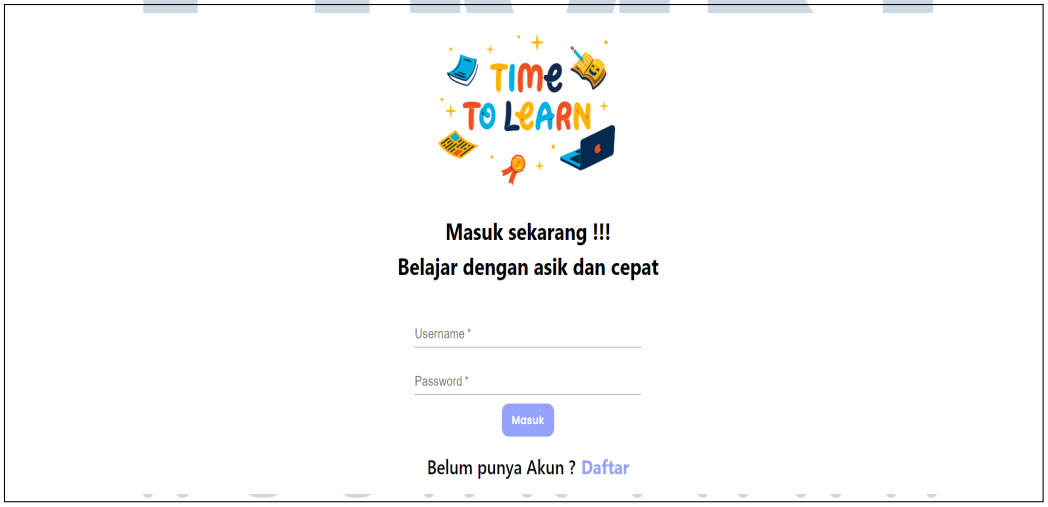
Gambar 3.39. Desain Antarmuka Halaman Beranda

Desain antarmuka halaman *register* ditampilkan pada Gambar 3.40. Pada gambar tersebut dapat dilihat terdapat *field* yang harus diisi oleh pengguna untuk mendaftar antara lain *username*, nama depan, nama belakang, email dan *password*. Selain itu juga terdapat tombol daftar yang akan mendaftarkan pengguna setelah pengguna memasukkan data-data yang diperlukan.



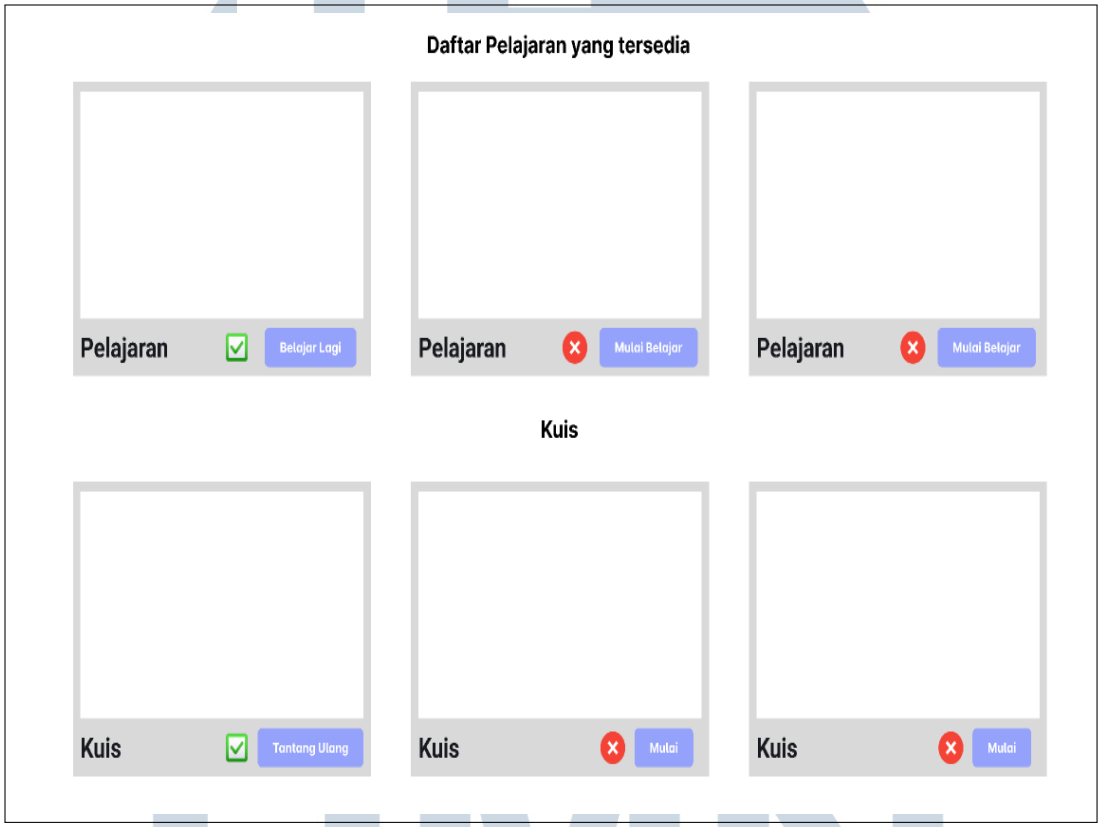
Gambar 3.40. Desain Antarmuka Halaman Register

Desain antarmuka halaman *login* ditampilkan pada Gambar 3.41. Pada gambar tersebut dapat dilihat terdapat *field* yang harus diisi oleh pengguna untuk masuk yaitu *username* dan *password*. Selain itu juga terdapat tombol masuk yang dapat ditekan untuk masuk ke aplikasi Rumah Hangeul dan tombol daftar yang akan membawa pengguna ke halaman *register*.



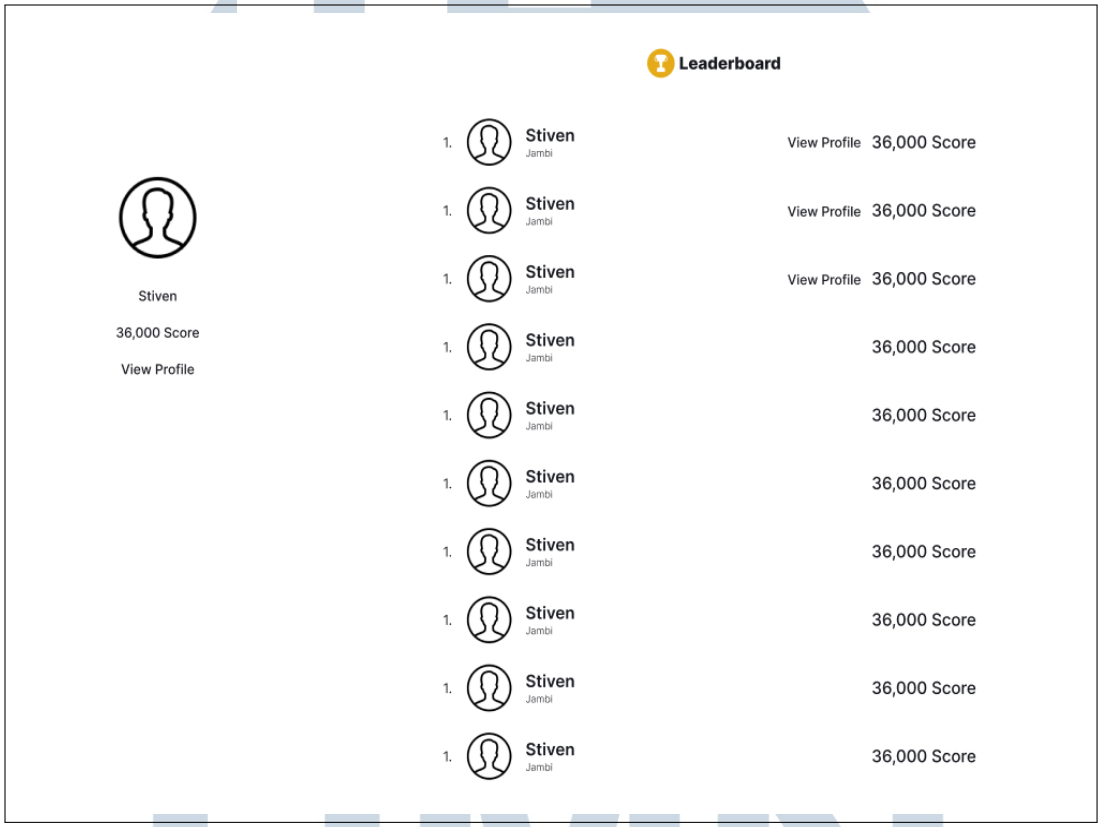
Gambar 3.41. Desain Antarmuka Halaman Login

Desain antarmuka halaman materi ditampilkan pada Gambar 3.42. Pada gambar tersebut dapat dilihat terdapat daftar pelajaran dan kuis yang dapat dikerjakan oleh pengguna dengan menekan tombol. Selain itu juga terdapat *icon* yang menandakan apakah pelajaran atau kuis tersebut sudah diselesaikan oleh pengguna atau belum.



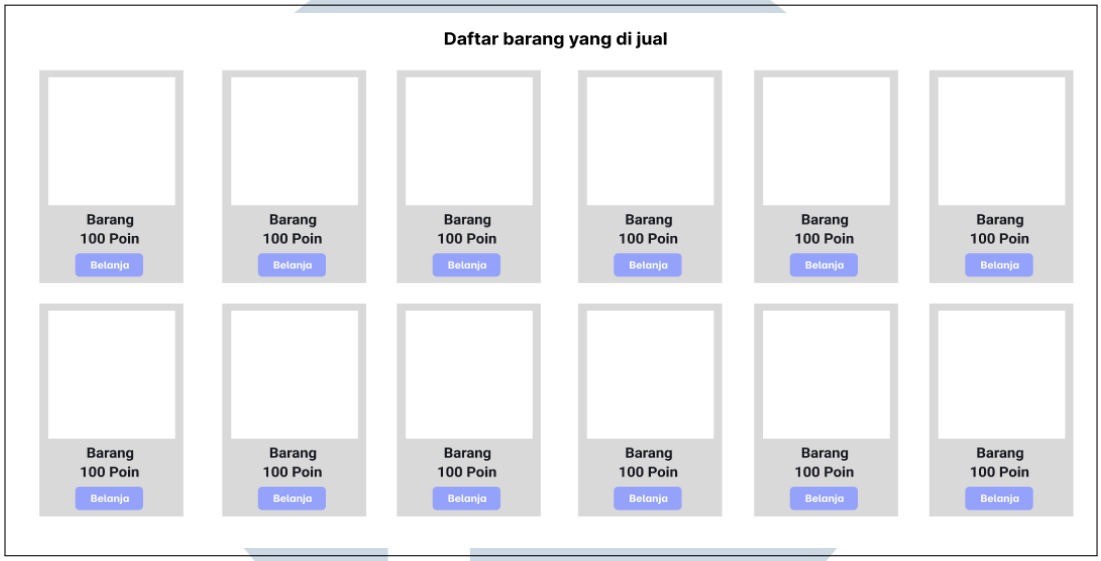
Gambar 3.42. Desain Antarmuka Halaman Materi

Desain antarmuka halaman peringkat ditampilkan pada Gambar 3.43. Pada gambar tersebut dapat dilihat terdapat informasi pengguna dan daftar pengguna yang memasuki peringkat di aplikasi Rumah Hangeul. Selain itu juga terdapat tombol yang dapat digunakan untuk melihat halaman profil pengguna dan profil pengguna peringkat satu sampai tiga.



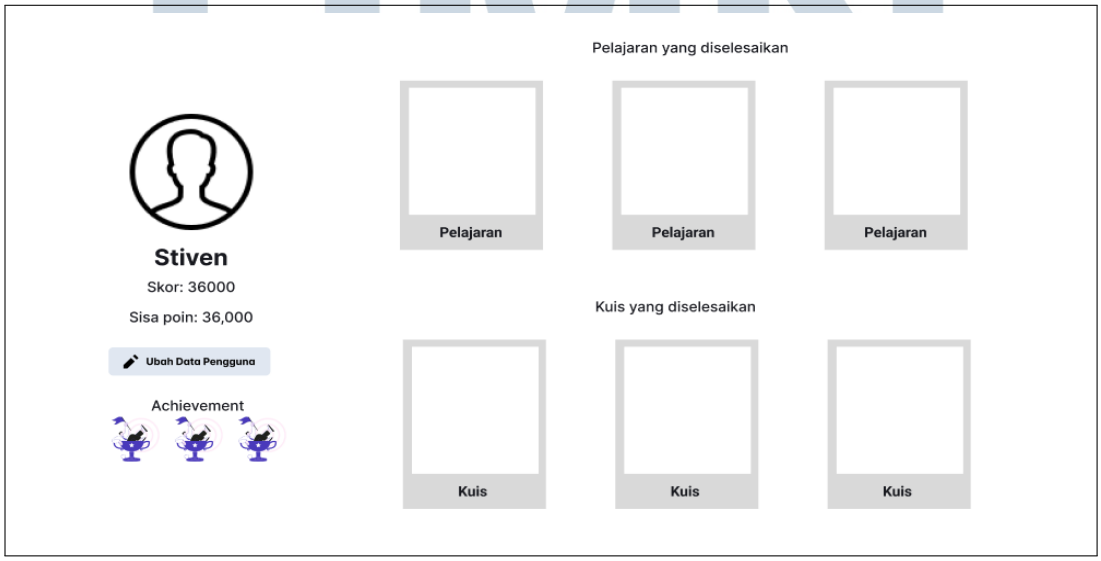
Gambar 3.43. Desain Antarmuka Halaman Peringkat

Desain antarmuka halaman toko ditampilkan pada Gambar 3.44. Pada gambar tersebut dapat daftar barang yang dijual berserta harganya pada halaman toko. Pengguna dapat menekan tombol belanja pada barang yang ingin dibeli.



Gambar 3.44. Desain Antarmuka Halaman Toko

Desain antarmuka halaman profil pengguna ditampilkan pada Gambar 3.45. Pada gambar tersebut dapat informasi mengenai pengguna berupa nama pengguna, skor, poin, *achievement*, pelajaran dan kuis yang telah diselesaikan oleh pengguna. Terdapat juga tombol ubah profil yang akan membawa pengguna ke halaman ubah data pengguna.

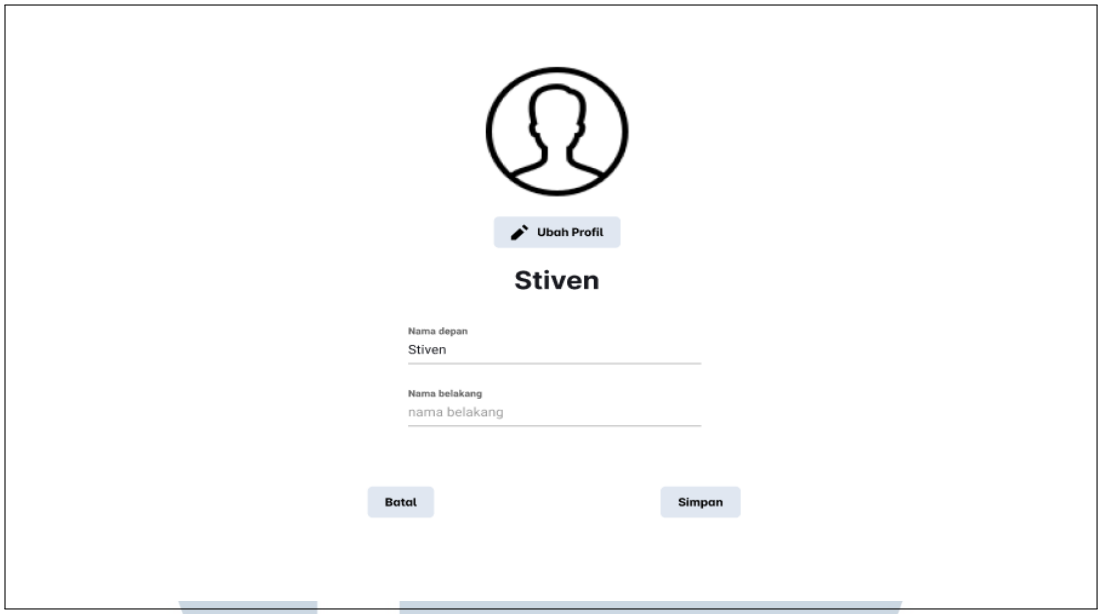


Gambar 3.45. Desain Antarmuka Halaman Profil Pengguna

Desain antarmuka halaman profil pengguna lain ditampilkan pada Gambar 3.46. Pada gambar tersebut dapat informasi mengenai pengguna lain berupa nama pengguna , skor, *achievement*, pelajaran dan kuis yang telah diselesaikan oleh pengguna lain.

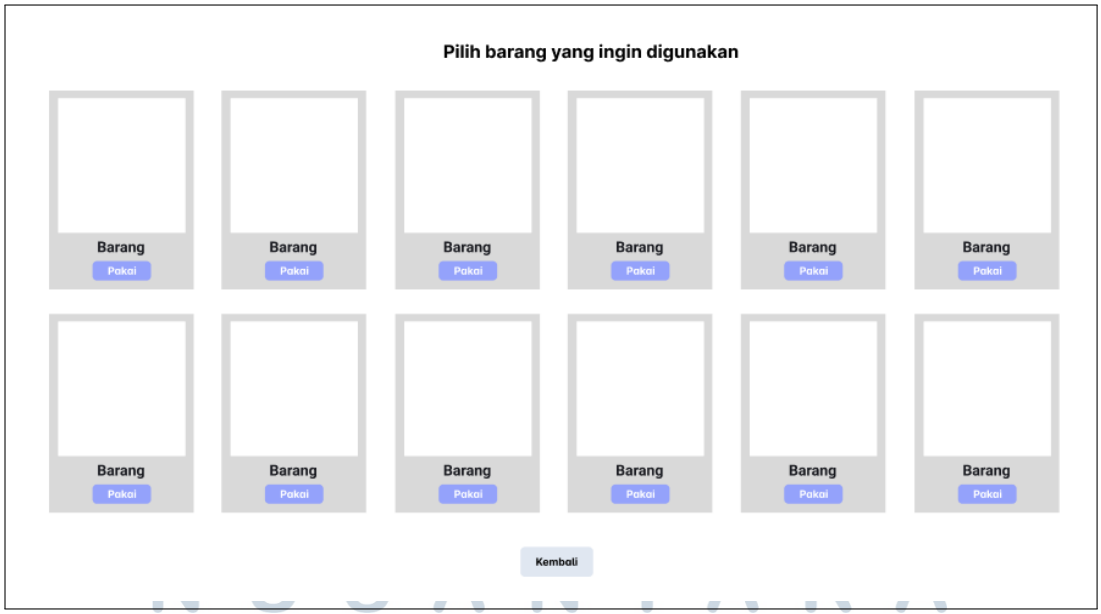
Gambar 3.46. Desain Antarmuka Halaman Profil Pengguna Lain

Desain antarmuka halaman ubah data pengguna ditampilkan pada Gambar 3.47. Pada gambar tersebut dapat informasi mengenai pengguna berupa nama pengguna. Terdapat juga *field* nama depan dan nama belakang yang dapat diisi oleh pengguna untuk mengubah nama mereka. Pada halaman ubah data pengguna juga terdapat tiga tombol yaitu tombol batal untuk kembali ke halaman profil pengguna, tombol simpan untuk menyimpan data yang telah diubah dan tombol ubah profil yang akan membawa pengguna ke halaman ubah profil pengguna.



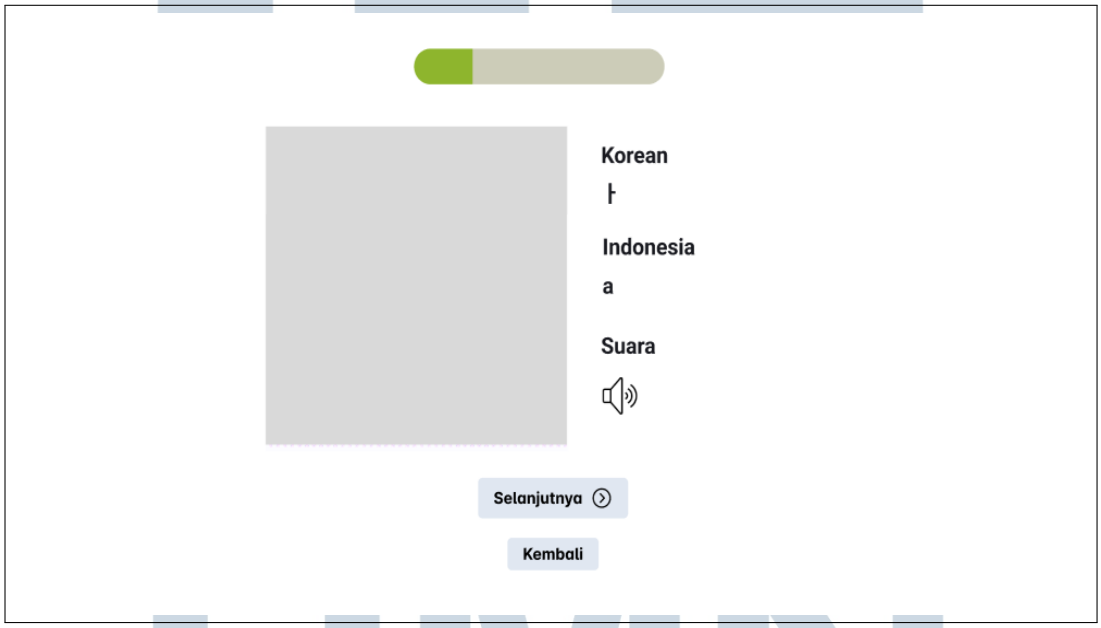
Gambar 3.47. Desain Antarmuka Halaman Ubah Data Pengguna

Desain antarmuka halaman ubah profil pengguna ditampilkan pada Gambar 3.48. Pada gambar tersebut dapat daftar barang yang dimiliki oleh pengguna dan tombol pakai untuk memakai barang yang dipilih oleh pengguna tersebut. Terdapat juga tombol kembali untuk kembali ke halaman ubah data pengguna.



Gambar 3.48. Desain Antarmuka Halaman Ubah Profil Pengguna

Desain antarmuka halaman pelajaran ditampilkan pada Gambar 3.49. Pada gambar tersebut dapat dilihat terdapat tulisan bahasa Korea serta terjemahannya dalam bahasa Indonesia. Pada halaman kuis terdapat juga *progress bar* yang menujukkan *progress* pengguna saat melakukan pembelajaran. Selain itu terdapat juga *icon* suara yang akan membunyikan cara membaca bahasa Korea yang ditampilkan tersebut jika ditekan. Terdapat juga tombol selanjutnya yang akan mengganti materi pelajaran bahasa Korea ke materi selanjutnya dan tombol kembali yang akan membawa pengguna ke halaman materi. Persegi abu-abu akan menampilkan gambar dari pelajaran yang sedang dilakukan.



Gambar 3.49. Desain Antarmuka Halaman Pelajaran

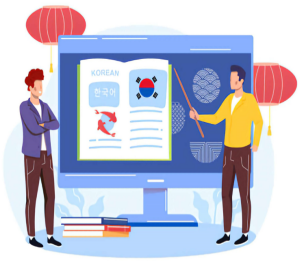
Desain antarmuka halaman kuis ditampilkan pada Gambar 3.50. Pada gambar tersebut dapat dilihat terdapat tulisan bahasa Korea serta empat pilihan bahasa Indonesia dimana salah satu pilihan tersebut merupakan terjemahan bahasa Indonesia yang benar. Saat pengguna menekan salah satu pilihan maka soal berikutnya akan ditampilkan. Pada halaman kuis terdapat juga *progress bar* yang menujukkan *progress* pengguna saat mengerjakan kuis dan sisa nyawa pengguna. Terdapat juga tombol kembali yang akan membawa pengguna ke halaman materi. *register*.

Gambar 3.50. Desain Antarmuka Halaman Kuis

3.2.5 Pemilihan Aset

Terdapat beberapa aset berupa gambar dan *icon* yang digunakan dalam desain antarmuka aplikasi Rumah Hangeul yang dapat dilihat pada.3.1

Tabel 3.1. Tabel Kegiatan Kerja Magang tiap Minggu

Aset	Deskripsi	Sumber
	Illustrasi pada halaman beranda	Icons8
	Illustrasi *achievement* pada halaman profil pengguna	Icons8
	Illustrasi *time to learn* pada halaman *register* dan *login*	Icons8
	*Icon* pada tombol daftar sekarang dan selanjutnya	Icons8
	*Icon* pada tombol ubah data pengguna dan ubah profil pengguna	Icons8
Lanjut pada halaman berikutnya		

Aset	Deskripsi	Sumber
	*Icon* suara pada halaman pelajaran	Icons8
	Illustrasi *progress bar* pada halaman pelajaran	Icons8
	*Icon* piala pada halaman peringkat	Icons8
	*Icon* yang menandakan pelajaran atau kuis sudah diselesaikan pada halaman materi	Icons8
	*Icon* yang menandakan pelajaran atau kuis belum diselesaikan pada halaman materi	Icons8