**Quiz App Documentation**

aplikasi quiz app yang saya buat menggunakan arsitektur DDD, Clean Architecture, dan Hexagonal Pattern. Untuk tech stacknya adalah Dotnet Core Versi 9.0.100, kemudian dotnet blazor WASM. Untuk message queue yang saya pilih adalah RabbitMQ, Dengan tujuan mengirimkan data message saat seorang admin menambahkan pertanyaan ke quiz yang akan dibuat dan saat participant memilih jawaban dari soal, tujuannya untuk auto save. Saya juga ingin mengimplementasikan redis cache untuk beberapa data dari database yang bersifat statis agar lebih cepat diakses.

Beberapa efisiensi yang saya pikirkan adalah:

* Dari sisi data, saya menggunakan NoSQL dan SQL untuk jenis-jenis data tertentu, contohnya seperti soal dan jawaban itu No SQL dikarenakan datanya bisa saja memiliki gambar. Kemudian untuk hasil quiz juga saya buat dalam bentuk NoSQL dikarenakan Sebagian besar hanya di read datanya.
* Dari sisi aplikasi backend, saya menggunakan micro service dengan memecah aplikasi menjadi 5 service terpisah yaitu User Service, Question Service, Quiz Service, Schedule Service, dan Quiz Result Service. Dengan tujuan agar setiap service bisa discale secara horizontally jika dibutuhkan dan tidak mengganggu fungsi service lainnya, sehingga tidak semua service menjadi lemot jika banyak user yang hanya menggunakan salah satu servicenya saja.
* Kemudian caching juga ingin saya gunakan, tapi belum sempat diimplementasikan, tetapi caching ini saya ingin tambahkan untuk mengambil dan menyimpan data-data statis seperti question status, quiz status dan semacamnya.
* Dari sisi frontend, saya ingin membuat sebuah UI yang Dimana load data itu by komponen sehingga saat user mengload sebuah halaman, yang diload itu hanya komponen yang akan ditampilkan. Implementasi pagination juga dilakukan.
* Saya juga menambahkan JWT untuk menghandle authorization setiap user. Harusnya menggunakan option HttpOnly yang saya taruh di cookie, tapi dikarenakan saya salah dalam membuat validasi JWT dibackend dikarenakan menggunakan Authorization Bearer jadi saya membuat HttpOnlynya jadi false saat ditaruh di cookie.
* Efisiensi

Berikut adalah detail dari data dan pilihan database yang saya buat:

Entity

1. User

- Id (UUID)

- Username (String)

- Description (String)

- Email (String)

- Password (String)

- DOB (Date)

- IsDeleted (Boolean)

- TotalTest (Int)

- TotalTestPassed (Int)

- TotalTestFailed (Int)

- LastLoginAt (Date)

- CreatedAt (Date)

- UpdatedAt (Date)

- DeletedAt (Date)

2. UserRole

- UserId (UUID)

- RoleId (UUID)

- IsDeleted (Date)

- CreatedAt (Date)

- UpdatedAt (Date)

- DeletedAt (Date)

3. Role

- Id (UUID)

- Name (String) "super\_admin", "admin\_quiz\_questioner", "admin\_scheduler", "admin\_qc", "admin\_quiz\_manager", "admin\_grading", "participant".

4. Question

- Id (UUID)

- Status (string)

- AnswerType (string)

- QuestionContent ([Object?])

- Point (Double)

- Answers

- CorrectAnswer

- IsDeleted (Boolean)

- CreatedBy (UUID) (Id User)

- CreatedAt (Date)

- UpdatedAt (Date)

- DeletedAt (Date)

5. QuestionStatus

- Name (String) "Draft", "Review", "Reject", "Finalized"

6. AnswerType

- Name (String) "multiple\_choice", "checkbox", "essay", "image\_upload", "mix"

7. Quiz

- Id (UUID)

- DifficutlyId (UUID)

- QuizStatusId (UUID)

- TagId (UUID)

- Name (String)

- Description (String)

- MinimumGrade (Double)

- TestDuration (Int)

- TotalQuestion (Int)

- IsDeleted (Boolean)

- CreatedBy (UUID)

- CreatedAt (Date)

- UpdatedAt (Date)

- DeletedAt (Date)

8. QuizStatus

- Id (UUID)

- Name (String) "Draft", "Review", "Reject", "Finalized"

9. QuizDetail

- QuizId (UUID)

- QuestionId (UUID)

- IsDeleted (Boolean)

- CreatedBy (UUID)

- CreatedAt (Date)

- UpdatedAt (Date)

- DeletedAt (Date)

10. FinalizedQuizDetail

- QuizId (UUID)

- Name (String)

- Description (String)

- MinimumGrade (Double)

- TestDuration (Int)

- TotalQuestion (Int)

- CreatedAt (Date)

- Questions [Question]

11. Tag

- Id (UUID)

- Name (String) "Math", "Science", "Communicating", "DKK"

12. Difficulty

- Id (UUID)

- Name (String) "beginner", "intermediate", "expert"

13. Schedule

- Id (UUID)

- QuizId (UUID)

- StatusId (UUID)

- StartAt (Date)

- EndAt (Date)

- MaxParticipant (Int)

- IsDeleted (Boolean)

- CreatedBy (UUID)

- CreatedAt (Date)

- UpdatedAt (Date)

- DeletedAt (Date)

14. ScheduleStatus

- Id (UUID)

- Name (String) "Not Started", "Registration", "Active", "Completed", "Cancelled"

15. ScheduleDetail

- ScheduleId (UUID)

- ParticipantId (UUID)

- IsDeleted (Boolean)

- CreatedAt (Date)

- UpdatedAt (Date)

- DeletedAt (Date)

16. QuizResultAnswer

- ParticipantId (UUID)

- ScheduleId (UUID)

- QuizId (UUID)

- QuestionId (UUID)

- Answers ([Object?])

- IsCorrect (Boolean)

17. QuizResult

- Id (UUID)

- ParticipantId (UUID)

- GradedBy (UUID)

- Schedule

- Id

- Status

- StartAt

- EndAt

- Quiz

- Id

- Difficulty

- Status

- Tag

- Name

- Description

- MinimumGrade

- TestDuration

- TotalQuestion

- QuizResultStatus (String)

- Score (Double)

- Reason (String)

- IsDeleted (Boolean)

- CreatedAt (Date)

- UpdatedAt (Date)

- DeletedAt (Date)

19. QuizResultQuestion

- ParticipantId

- ScheduleId

- QuizId

- Questions

- Id

- QuestionContent

- Point

- Answers

20. QuizResultStatus

- Name (String) "In-Check", "Failed", "Passed"

Database Type:

- User -> SQL

- UserRole -> SQL

- Role -> SQL

- Question -> No SQL

- QuestionStatus -> No SQL

- AnswerType -> No SQL

- Quiz -> SQL

- QuizStatus -> SQL

- QuizDetail -> SQL

- FinalizedQuizDetail -> No SQL

- Tag -> SQL

- Difficulty -> SQL

- Schedule -> SQL

- ScheduleStatus -> SQL

- ScheduleDetail -> SQL

- QuizResult -> No SQL

- QuizResultDetail -> No SQL

- QuizResultStatus -> No SQL

Database:

- PostgreSQL

- MongoDB

- Redis (Caching)

Dan ini adalah detail per micro servicenya:

Services

- User Service

Database Entity = User, UserRole, Role (Full SQL DB) -> 1 DB

- Question Service

Database Entity = Question, QuestionStatus, AnswerType (Full No SQL DB) -> 1 DB

- Quiz Service

Database Entity = Quiz, QuizStatus, QuizDetail, FinalizedQuizDetail, Tag, Difficulty (SQL and No SQL DB) -> 2 DB

- Schedule Service

Database Entity = Schedule, ScheduleStatus, ScheduleDetail (Full SQL DB) -> 1 DB

- QuizResult Service

Database Entity = QuizResult, QuizResultDetail, QuizResultStatus (Full No SQL DB) -> 1 DB