Nama: Aldiandya Irsyad

Project: Simple Blog API

# DB

DB dibuat dengan automigrate() (GORM v2.0) , secara default tidak ada FK pada database level.

Diagram

Description automatically generated

Koneksi DB dihubungkan dengan models.connect\_DB.

Semua post dan put memiliki struct menegah didefinisikan di controllers.structinput

# Middleware

Semua /users/<id>, harus memiliki auth sesuai dengan <id> mereka sendiri, bahkan admin tidak diperbolehkan mengaksesnya. Namun admin dapat mengakses dengan jalur lain bila ingin mendelete sesuatu.

Terdapat middleware khusus Authentication.auth, yang bersifat OR dibanding dengan middleware default (bersifat AND). Parameternya menggunakan fungsi fungsi lainnya.

# POSTS

type Post struct {

    gorm.Model

    UserID     uint       `json:"user\_id"`

    Title      string     `json:"title"`

    Summary    string     `json:"summary"`

    Content    string     `json:"content" gorm:"type:text"`

    Categories []Category `gorm:"many2many:post\_categories"`

    Tags       []Tag      `gorm:"many2many:post\_tags"`

    Comments   []Comment

}

Post memiliki relasi many to many dengan categories dan tags, many to one dengan user, dan one to many dengan komen.

type createPostInput struct {

    Title      string   `json:"title"`

    Summary    string   `json:"summary"`

    Content    string   `json:"content"`

    Categories []string `json:"categories"`

    Tags       []string `json:"tags"`

}

Terdapat 6 methods yang dapat mengakses POST, seperti biasa semua PUT dan POST memiliki structure diatas.

server.GET("/posts", authentication.Login, controller.GetAllPosts)

server.GET("/posts/:postID", authentication.Login, controller.GetPost)

server.POST("/users/:userID/posts", authentication.Login, authentication.HandlerC

reatePost, controller.UserCreatePost)

server.PUT("/users/:userID/posts/:postID", authentication.Login, authentication.HandlerEditPost, controller.UpdatePost)

server.DELETE("/users/:userID/posts/:postID", authentication.Login, authentication.HandlerDeletePost, controller.DeletePost)

server.DELETE("posts/:postID", authentication.Login, authentication.HandlerDeletePost, controller.DeletePost)

# Comments

type Comment struct {

    gorm.Model

    UserID  uint   `json:"user\_id"`

    PostID  uint   `json:"post\_id"`

    Content string `json:"content" gorm:"type:text"`

}

Comments memiliki hubungan many to one dengan post dan users. Comments tidak dapat diedit. Hanya ada 3 methods

server.POST("/users/:userID/posts/:postID/comments", authentication.Login, controller.UserOnPostCreateComments)

server.DELETE("/users/:userID/posts/:postID/comments/:commentID", authentication.Login, authentication.HandlerDeleteComment, controller.UserOnPostDeleteComment)

server.DELETE("posts/:postID/comments/:commentID", authentication.Login, authentication.HandlerDeleteComment, controller.UserOnPostDeleteComment)

Original poster dapat mendelete comment yang dibuat di post nya namun tidak di post lainnya.

Struktur post nya sangat sederhana

type createCommentInput struct {

    Content string `json:"content" binding:"required"`

}

# Categories dan Tags

Mereka memiliki struktur yang sangat mirip pada implementasi, hanya berbeda secara high-level. Mereka menggunakan controller.createLabelInput sebagai struct sementara

type createLabelInput struct {

    Title       string `json:"title"`

    Description string `json:"description"`

}

type Tag struct {

    gorm.Model

    Title       string `json:"title" gorm:"unique"`

    Description string `json:"description"`

}

type Category struct {

    gorm.Model

    Title       string `json:"title" gorm:"unique"`

    Description string `json:"description"`

}

Mereka memiliki CRUD lengkap

server.GET("/categories/", authentication.Login, authentication.IsAdmin, controller.GetAllCategories)

server.POST("/categories", authentication.Login, authentication.IsAdmin, controller.CreateCategory)

server.PUT("/categories/:categoryID", authentication.Login, authentication.IsAdmin, controller.UpdateCategory)

server.DELETE("/categories/:categoryID", authentication.Login, authentication.IsAdmin, controller.DeleteCategory)

Namun hanya admin yang dapat mengaksesnya

# Users

User memiliki access level, 1 = admin, 2 = poster, 3 sama reader

type User struct {

    gorm.Model

    Username   string `json:"username"`

    Password   string `json:"password"`

    Privillege int64  `json:"privillege"`

    // Relation

    Posts    []Post

    Comments []Comment

}

type createUserInput struct {

    Username   string `json:"username"`

    Password   string `json:"password"`

    Privillege int64  `json:"privillege"`

}

Hanya admin yang dapat merubah akses privillege, mereka mempunyai 5 methods

server.GET("/users", authentication.Login, authentication.IsAdmin, controller.GetAllUsers)

server.POST("/users", controller.CreateUser)

server.PUT("/users/:userID/changePassword", authentication.Login, controller.ChangePassword)

server.PUT("/edituser/:id", authentication.Login, authentication.IsAdmin, controller.UpdateUser)

server.DELETE("/deleteuser/:id", authentication.Login, authentication.IsAdmin, controller.DeletePost)

Yang dapat diakses bukan admin hanya 2 yaitu changePassword dan createUser.