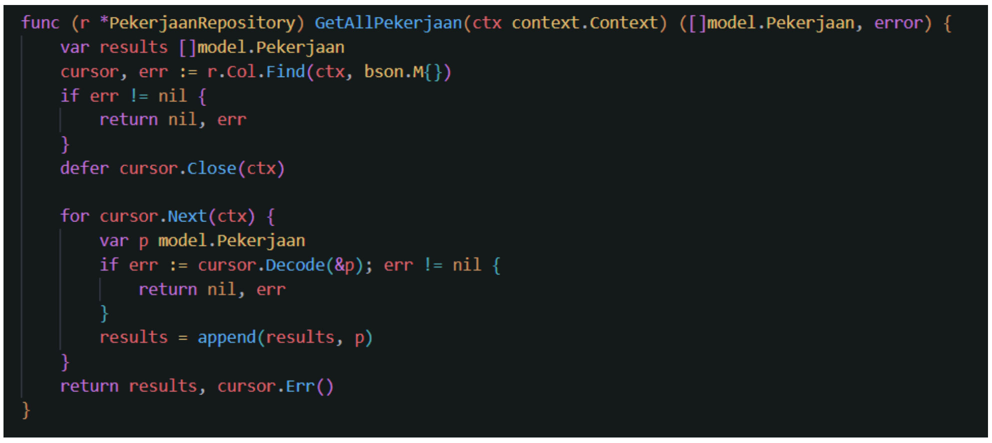
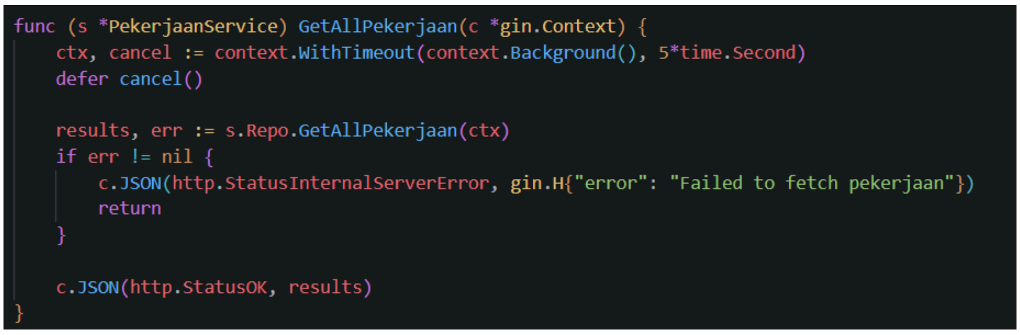
Nama: Daffa Surya Arrayan
kelas: C5
Nim: 434231042

LAPORAN BACKEND PRAKTIKUM MONGODB

❖ Ambil Semua data pekerjaan alumni (admin)

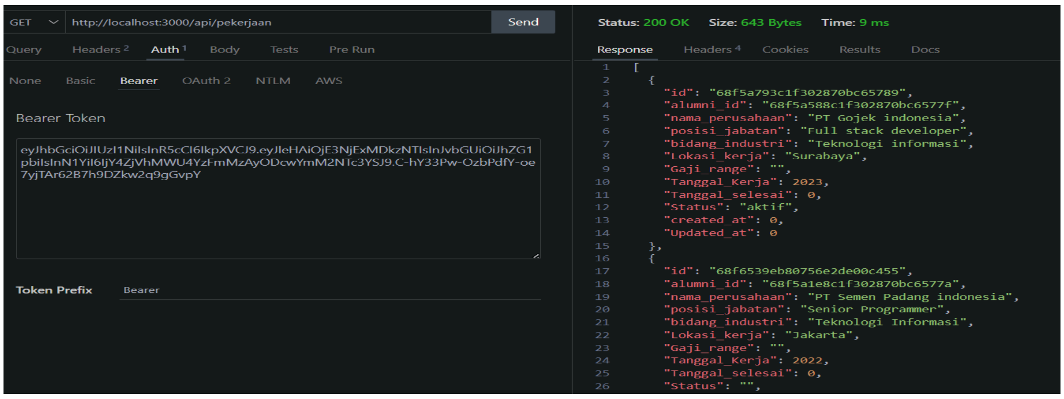


```go
func (r *PekerjaanRepository) GetAllPekerjaan(ctx context.Context) ([]model.Pekerjaan, error) {
    var results []model.Pekerjaan
    cursor, err := r.Col.Find(ctx, bson.M{})
    if err != nil {
        return nil, err
    }
    defer cursor.Close(ctx)

    for cursor.Next(ctx) {
        var p model.Pekerjaan
        if err := cursor.Decode(&p); err != nil {
            return nil, err
        }
        results = append(results, p)
    }
    return results, cursor.Err()
}
```



```go
func (s *PekerjaanService) GetAllPekerjaan(c *gin.Context) {
    ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
    defer cancel()

    results, err := s.Repo.GetAllPekerjaan(ctx)
    if err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Failed to fetch pekerjaan"})
        return
    }

    c.JSON(http.StatusOK, results)
}
```

Result:



```
GET http://localhost:3000/api/pekerjaan
```

Bearer Token

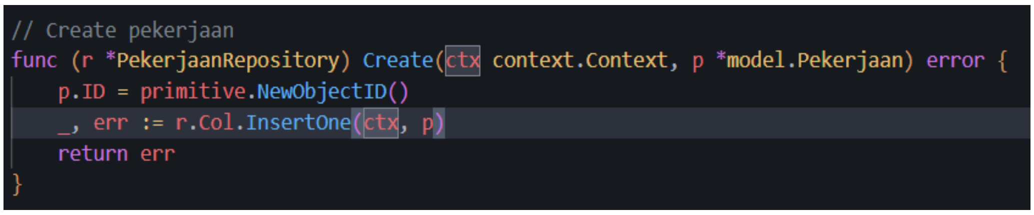
```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3NjExMDkzNTIsInJvbGUiOiJhZG1
pbiIsInN1YiI6IjY4ZjVhMWU4YzFmMzAyODcwYmM2NTc3YSJ9.C-hY33Pw-OzbPdfY-oe
7yjTAr62B7h9DZkw2q9gGvpY
```

Token Prefix: Bearer

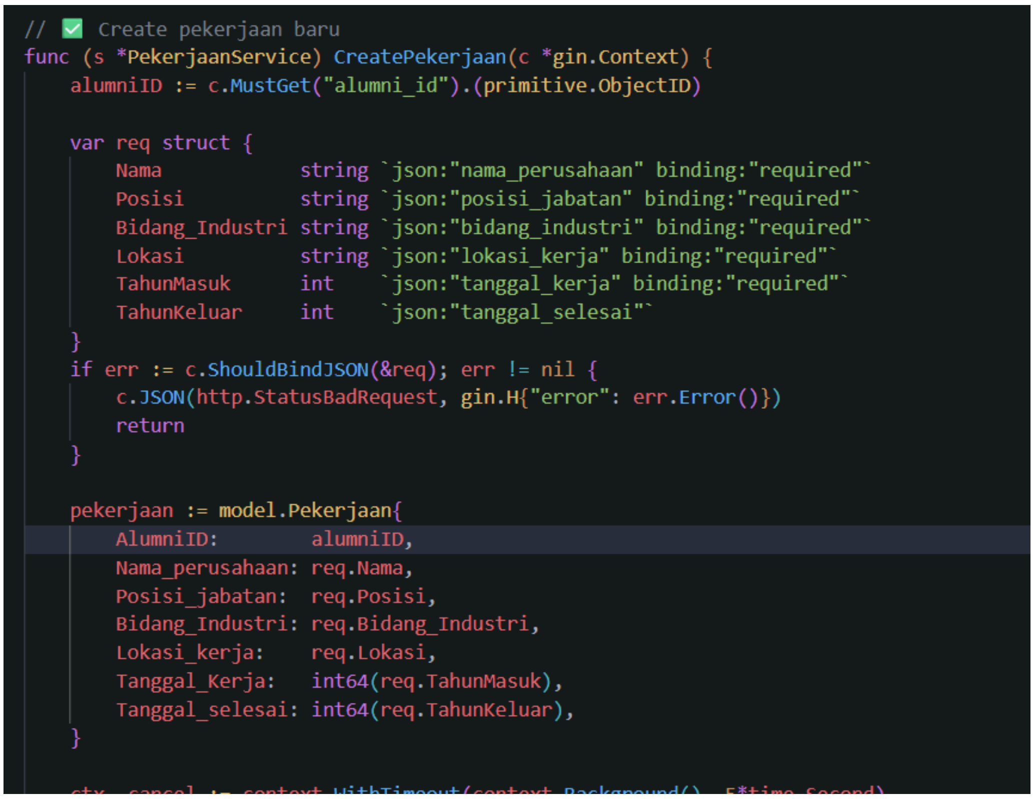
Status: 200 OK Size: 643 Bytes Time: 9 ms

```json
[
  {
    "id": "68f5a793c1f302870bc65789",
    "alumni_id": "68f5a588c1f302870bc6577f",
    "nama_perusahaan": "PT Gojek indonesia",
    "posisi_jabatan": "Full stack developer",
    "bidang_industri": "Teknologi informasi",
    "Lokasi_kerja": "Surabaya",
    "Gaji_range": "",
    "Tanggal_Kerja": 2023,
    "Tanggal_selesai": 0,
    "Status": "aktif",
    "created_at": 0,
    "Updated_at": 0
  },
  {
    "id": "68f6539eb80756e2de00c455",
    "alumni_id": "68f5a1e8c1f302870bc6577a",
    "nama_perusahaan": "PT Semen Padang indonesia",
    "posisi_jabatan": "Senior Programmer",
    "bidang_industri": "Teknologi Informasi",
    "Lokasi_kerja": "Jakarta",
    "Gaji_range": "",
    "Tanggal_Kerja": 2022,
    "Tanggal_selesai": 0,
    "Status": "",
```

❖ Tambah Pekerjaan Baru (admin)

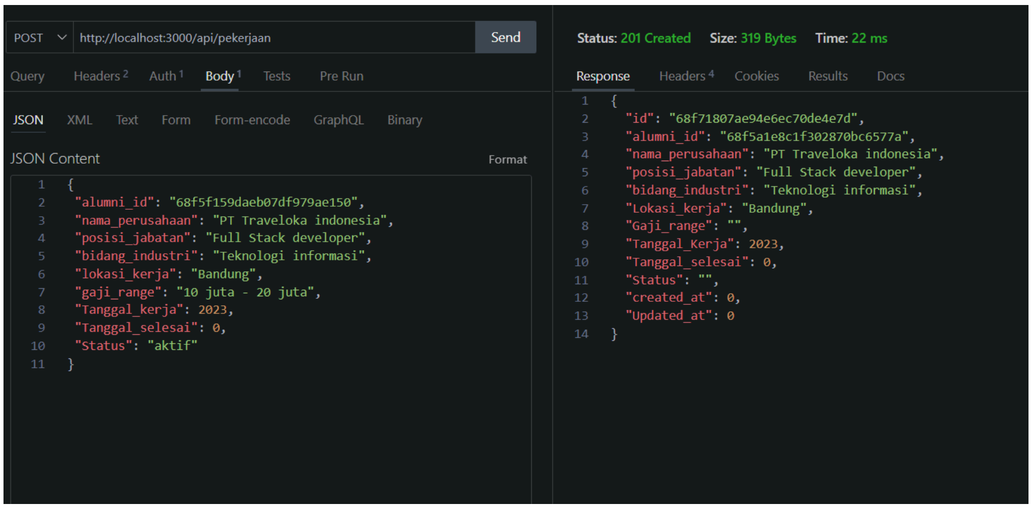


```go
// Create pekerjaan
func (r *PekerjaanRepository) Create(ctx context.Context, p *model.Pekerjaan) error {
    p.ID = primitive.NewObjectID()
    _, err := r.Col.InsertOne(ctx, p)
    return err
}
```



```go
// ✅ Create pekerjaan baru
func (s *PekerjaanService) CreatePekerjaan(c *gin.Context) {
    alumniID := c.MustGet("alumni_id").(primitive.ObjectID)

    var req struct {
        Nama            string `json:"nama_perusahaan" binding:"required"`
        Posisi          string `json:"posisi_jabatan" binding:"required"`
        Bidang_Industri string `json:"bidang_industri" binding:"required"`
        Lokasi          string `json:"lokasi_kerja" binding:"required"`
        TahunMasuk      int    `json:"tanggal_kerja" binding:"required"`
        TahunKeluar     int    `json:"tanggal_selesai"`
    }
    if err := c.ShouldBindJSON(&req); err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": err.Error()})
        return
    }

    pekerjaan := model.Pekerjaan{
        AlumniID:        alumniID,
        Nama_perusahaan: req.Nama,
        Posisi_jabatan:  req.Posisi,
        Bidang_Industri: req.Bidang_Industri,
        Lokasi_kerja:    req.Lokasi,
        Tanggal_Kerja:   int64(req.TahunMasuk),
        Tanggal_selesai: int64(req.TahunKeluar),
    }

    ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
```

Result:



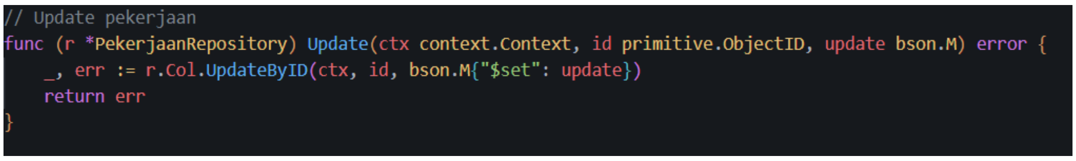
POST http://localhost:3000/api/pekerjaan

```json
{
 "alumni_id": "68f5f159daeb07df979ae150",
 "nama_perusahaan": "PT Traveloka indonesia",
 "posisi_jabatan": "Full Stack developer",
 "bidang_industri": "Teknologi informasi",
 "lokasi_kerja": "Bandung",
 "gaji_range": "10 juta - 20 juta",
 "Tanggal_kerja": 2023,
 "Tanggal_selesai": 0,
 "Status": "aktif"
}
```

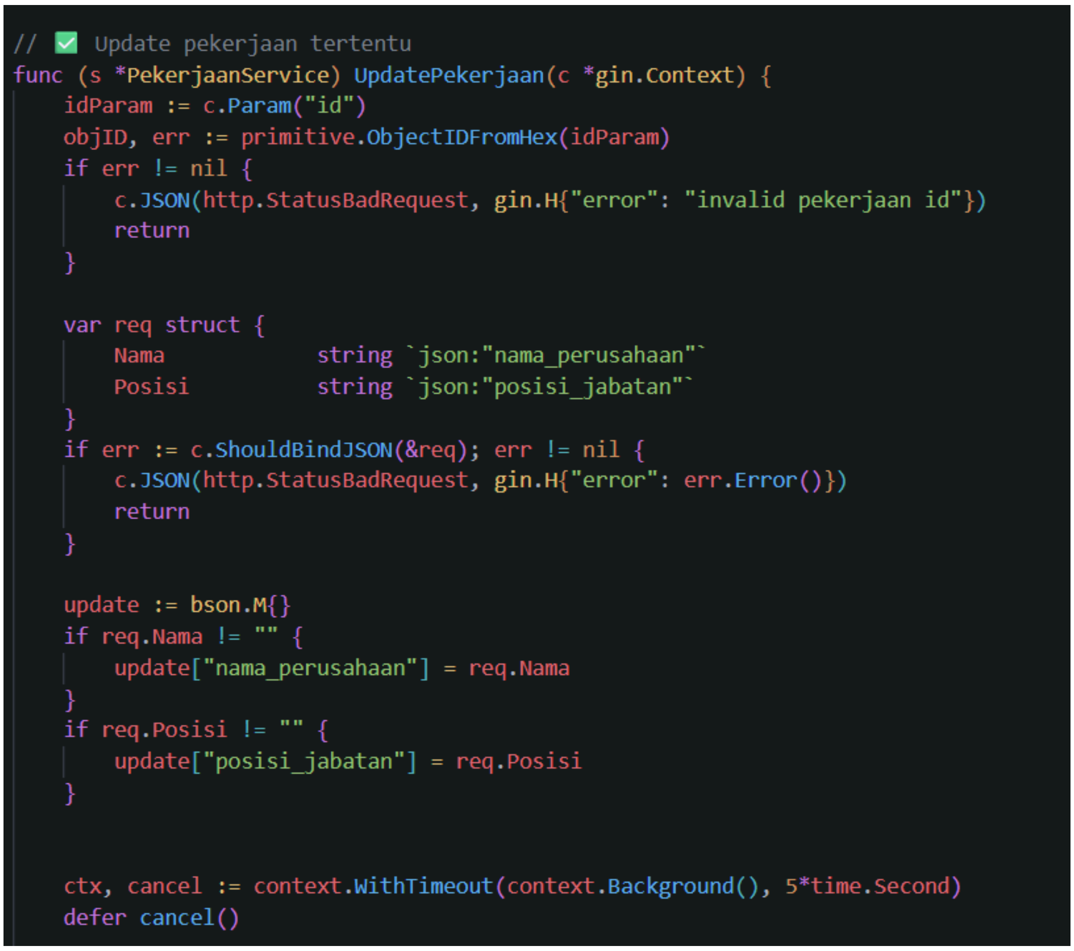
Status: 201 Created   Size: 319 Bytes   Time: 22 ms

```json
{
  "id": "68f71807ae94e6ec70de4e7d",
  "alumni_id": "68f5a1e8c1f302870bc6577a",
  "nama_perusahaan": "PT Traveloka indonesia",
  "posisi_jabatan": "Full Stack developer",
  "bidang_industri": "Teknologi informasi",
  "Lokasi_kerja": "Bandung",
  "Gaji_range": "",
  "Tanggal_Kerja": 2023,
  "Tanggal_selesai": 0,
  "Status": "",
  "created_at": 0,
  "Updated_at": 0
}
```

❖ Update Pekerjaan (admin)

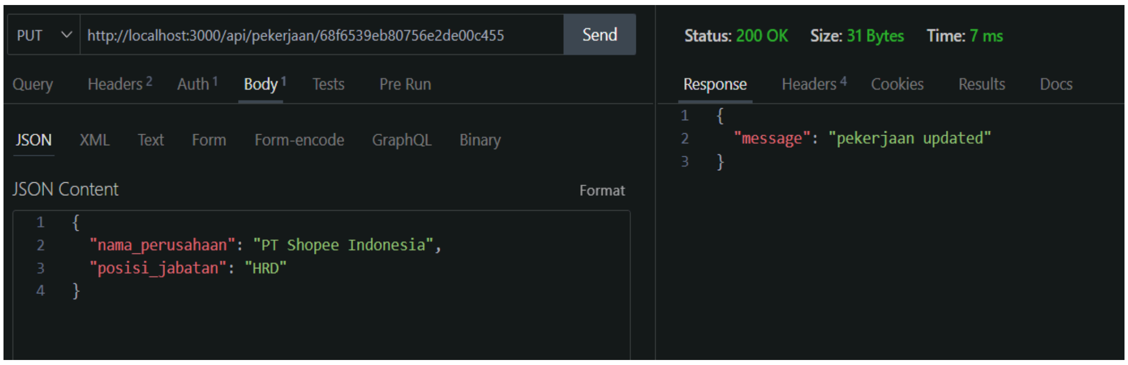


```go
// Update pekerjaan
func (r *PekerjaanRepository) Update(ctx context.Context, id primitive.ObjectID, update bson.M) error {
    _, err := r.Col.UpdateByID(ctx, id, bson.M{"$set": update})
    return err
}
```



```go
// ✅ Update pekerjaan tertentu
func (s *PekerjaanService) UpdatePekerjaan(c *gin.Context) {
    idParam := c.Param("id")
    objID, err := primitive.ObjectIDFromHex(idParam)
    if err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "invalid pekerjaan id"})
        return
    }

    var req struct {
        Nama            string `json:"nama_perusahaan"`
        Posisi          string `json:"posisi_jabatan"`
    }
    if err := c.ShouldBindJSON(&req); err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": err.Error()})
        return
    }

    update := bson.M{}
    if req.Nama != "" {
        update["nama_perusahaan"] = req.Nama
    }
    if req.Posisi != "" {
        update["posisi_jabatan"] = req.Posisi
    }


    ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
    defer cancel()
```

Result:



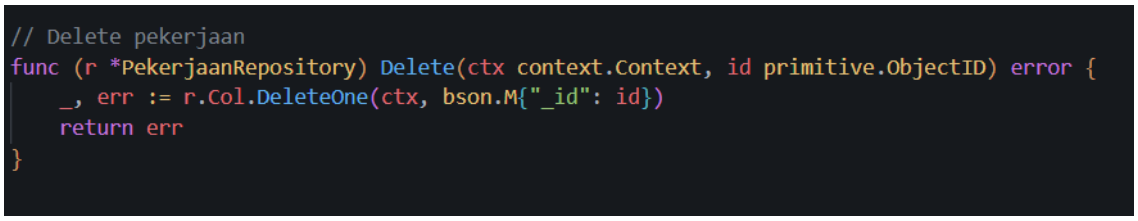
PUT http://localhost:3000/api/pekerjaan/68f6539eb80756e2de00c455 Send

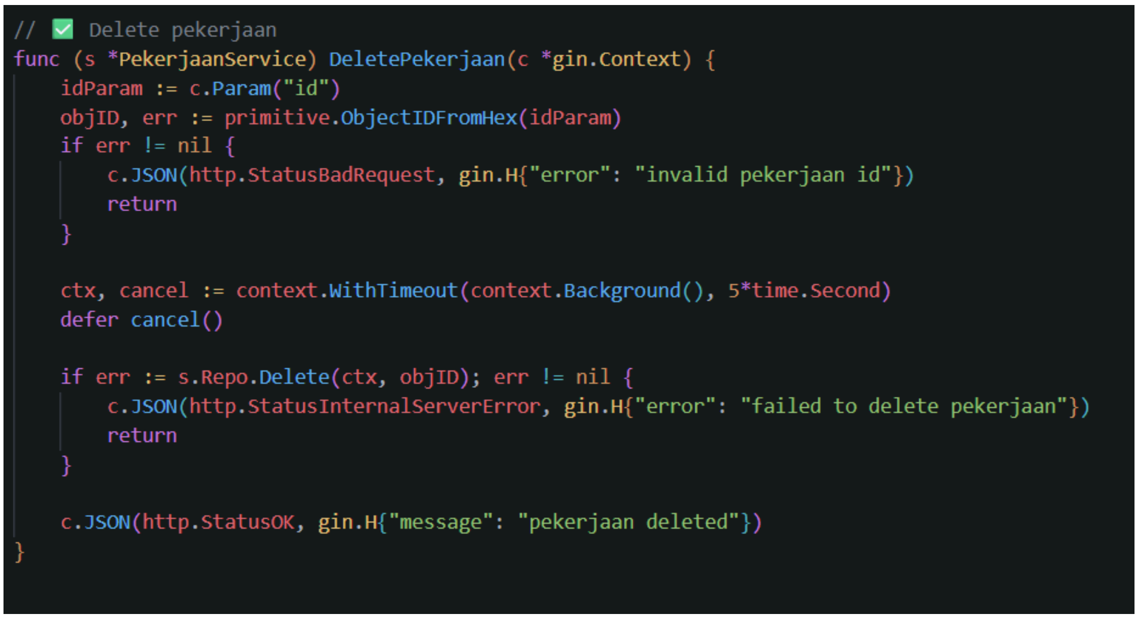
Status: 200 OK Size: 31 Bytes Time: 7 ms

```json
{
  "nama_perusahaan": "PT Shopee Indonesia",
  "posisi_jabatan": "HRD"
}
```

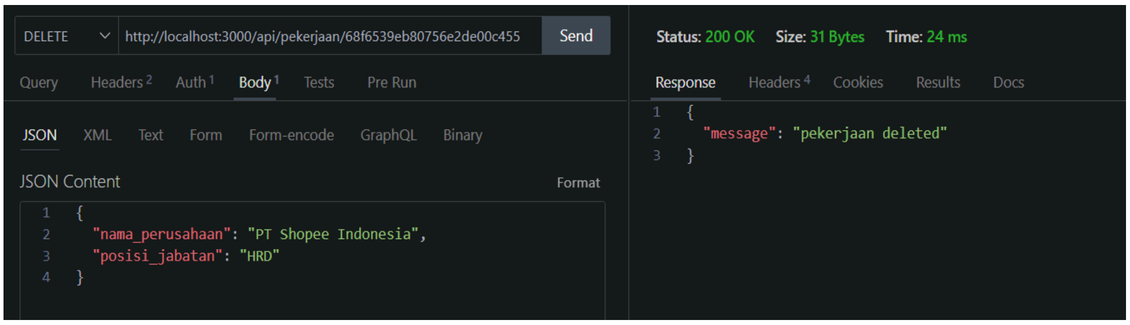
```json
{
  "message": "pekerjaan updated"
}
```

❖ Delete Pekerjaan (admin)


```go
// Delete pekerjaan
func (r *PekerjaanRepository) Delete(ctx context.Context, id primitive.ObjectID) error {
    _, err := r.Col.DeleteOne(ctx, bson.M{"_id": id})
    return err
}
```


```go
// ✅ Delete pekerjaan
func (s *PekerjaanService) DeletePekerjaan(c *gin.Context) {
    idParam := c.Param("id")
    objID, err := primitive.ObjectIDFromHex(idParam)
    if err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "invalid pekerjaan id"})
        return
    }

    ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
    defer cancel()

    if err := s.Repo.Delete(ctx, objID); err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "failed to delete pekerjaan"})
        return
    }

    c.JSON(http.StatusOK, gin.H{"message": "pekerjaan deleted"})
}
```

Result:



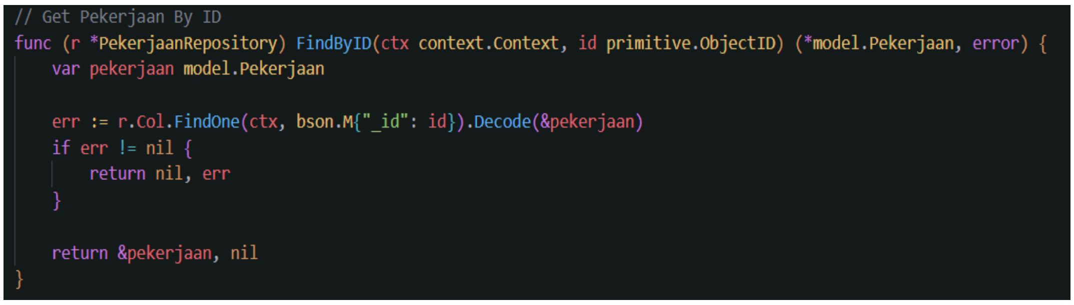
DELETE http://localhost:3000/api/pekerjaan/68f6539eb80756e2de00c455

JSON Content
```json
{
  "nama_perusahaan": "PT Shopee Indonesia",
  "posisi_jabatan": "HRD"
}
```

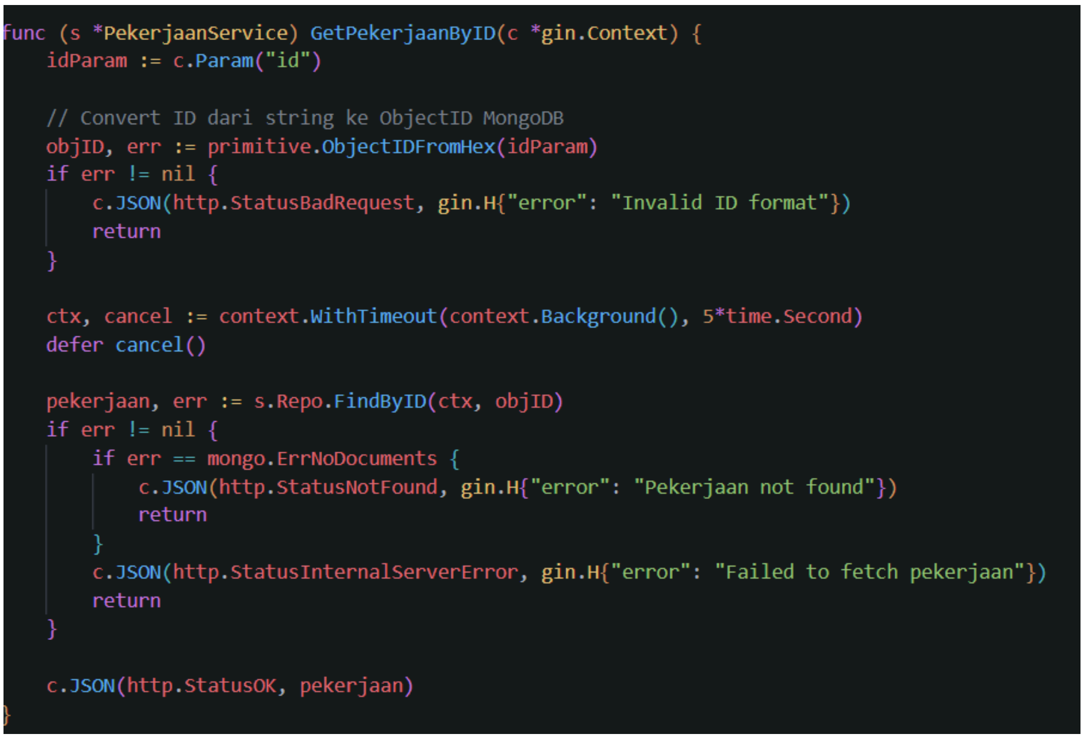
Status: 200 OK   Size: 31 Bytes   Time: 24 ms

```json
{
  "message": "pekerjaan deleted"
}
```

❖ Get Pekerjaan By ID (admin)

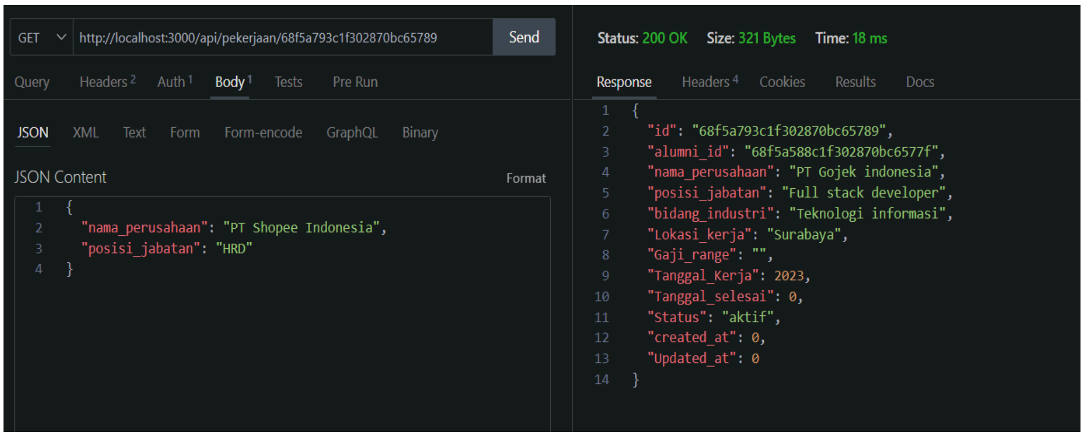


```go
// Get Pekerjaan By ID
func (r *PekerjaanRepository) FindByID(ctx context.Context, id primitive.ObjectID) (*model.Pekerjaan, error) {
    var pekerjaan model.Pekerjaan

    err := r.Col.FindOne(ctx, bson.M{"_id": id}).Decode(&pekerjaan)
    if err != nil {
        return nil, err
    }

    return &pekerjaan, nil
}
```



```go
func (s *PekerjaanService) GetPekerjaanByID(c *gin.Context) {
    idParam := c.Param("id")

    // Convert ID dari string ke ObjectID MongoDB
    objID, err := primitive.ObjectIDFromHex(idParam)
    if err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "Invalid ID format"})
        return
    }

    ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
    defer cancel()

    pekerjaan, err := s.Repo.FindByID(ctx, objID)
    if err != nil {
        if err == mongo.ErrNoDocuments {
            c.JSON(http.StatusNotFound, gin.H{"error": "Pekerjaan not found"})
            return
        }
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Failed to fetch pekerjaan"})
        return
    }

    c.JSON(http.StatusOK, pekerjaan)
}
```

Result:



http://localhost:3000/api/pekerjaan/68f5a793c1f302870bc65789

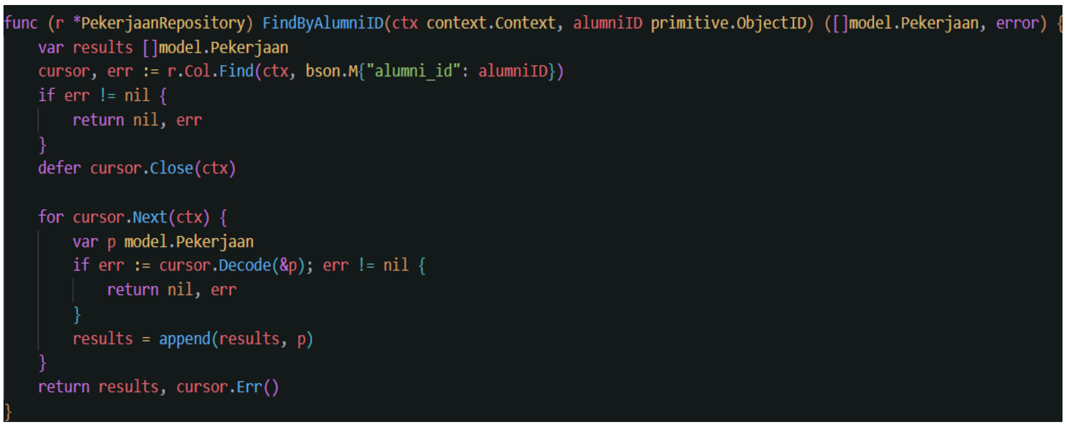
JSON Content

```json
{
  "nama_perusahaan": "PT Shopee Indonesia",
  "posisi_jabatan": "HRD"
}
```

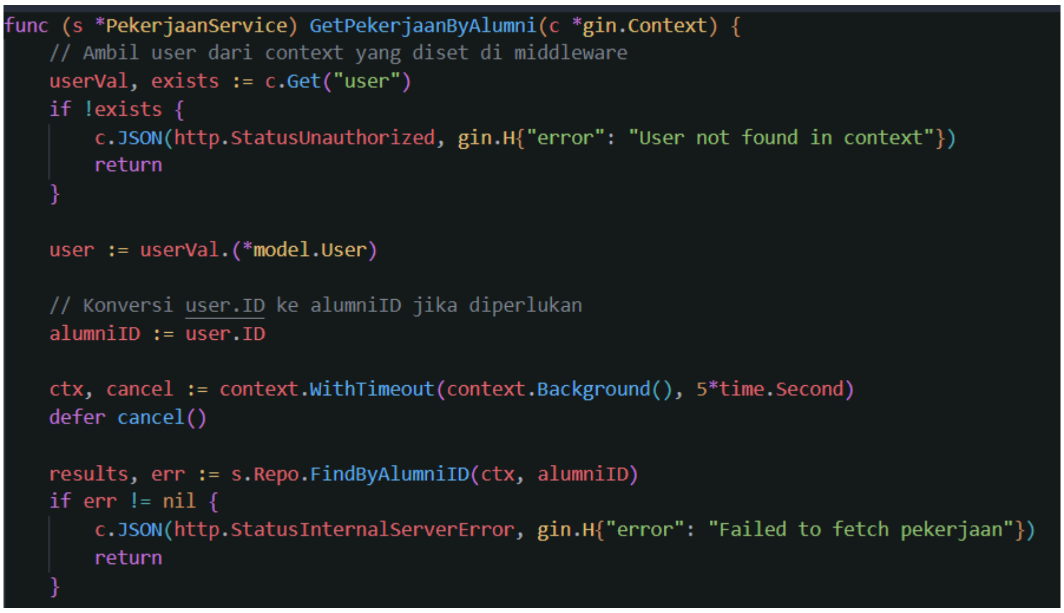
Status: 200 OK Size: 321 Bytes Time: 18 ms

```json
{
  "id": "68f5a793c1f302870bc65789",
  "alumni_id": "68f5a588c1f302870bc6577f",
  "nama_perusahaan": "PT Gojek indonesia",
  "posisi_jabatan": "Full stack developer",
  "bidang_industri": "Teknologi informasi",
  "Lokasi_kerja": "Surabaya",
  "Gaji_range": "",
  "Tanggal_Kerja": 2023,
  "Tanggal_selesai": 0,
  "Status": "aktif",
  "created_at": 0,
  "Updated_at": 0
}
```

❖ Get Pekerjaan By Alumni ID (admin)

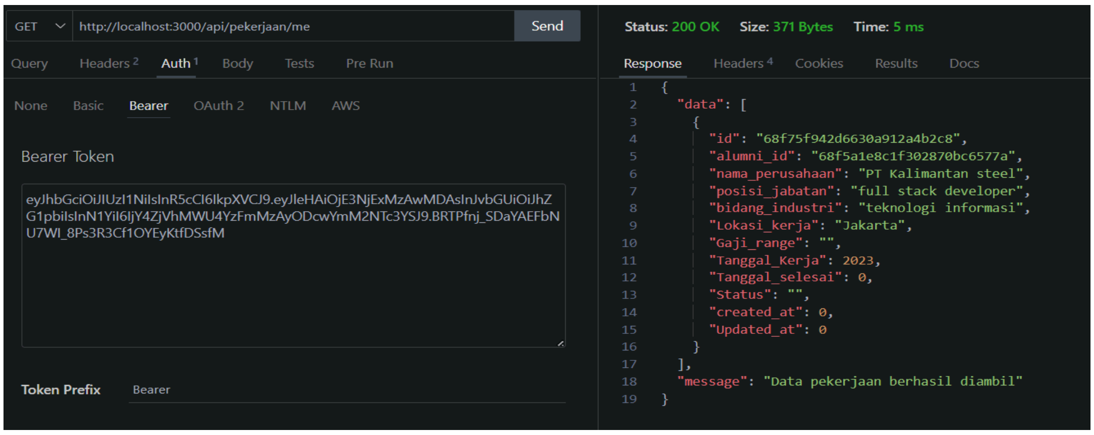


```go
func (r *PekerjaanRepository) FindByAlumniID(ctx context.Context, alumniID primitive.ObjectID) ([]model.Pekerjaan, error) {
    var results []model.Pekerjaan
    cursor, err := r.Col.Find(ctx, bson.M{"alumni_id": alumniID})
    if err != nil {
        return nil, err
    }
    defer cursor.Close(ctx)

    for cursor.Next(ctx) {
        var p model.Pekerjaan
        if err := cursor.Decode(&p); err != nil {
            return nil, err
        }
        results = append(results, p)
    }
    return results, cursor.Err()
}
```



```go
func (s *PekerjaanService) GetPekerjaanByAlumni(c *gin.Context) {
    // Ambil user dari context yang diset di middleware
    userVal, exists := c.Get("user")
    if !exists {
        c.JSON(http.StatusUnauthorized, gin.H{"error": "User not found in context"})
        return
    }

    user := userVal.(*model.User)

    // Konversi user.ID ke alumniID jika diperlukan
    alumniID := user.ID

    ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
    defer cancel()

    results, err := s.Repo.FindByAlumniID(ctx, alumniID)
    if err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Failed to fetch pekerjaan"})
        return
    }
```

Result:



```
http://localhost:3000/api/pekerjaan/me
```

Bearer Token

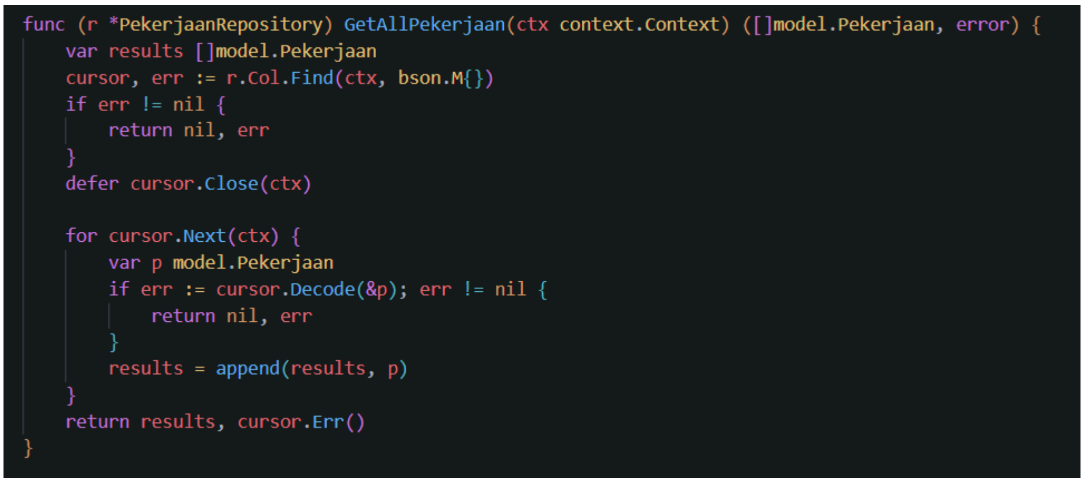
```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3NjExMzAwMDAsInJvbGUiOiJhZ
G1pbiIsInN1YiI6IjY4ZjVhMWU4YzFmMzAyODcwYmM2NTc3YSJ9.BRTPfnj_SDaYAEFbN
U7WI_8Ps3R3Cf1OYEyKtfDSsfM
```

Token Prefix Bearer

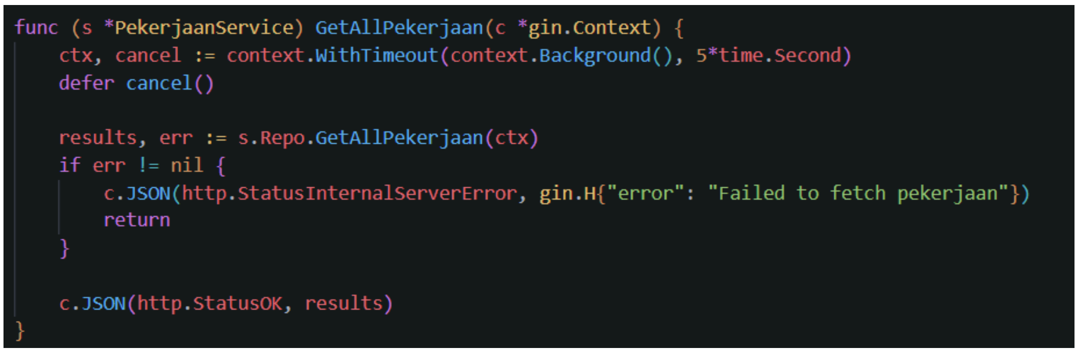
Status: 200 OK Size: 371 Bytes Time: 5 ms

```json
{
  "data": [
    {
      "id": "68f75f942d6630a912a4b2c8",
      "alumni_id": "68f5a1e8c1f302870bc6577a",
      "nama_perusahaan": "PT Kalimantan steel",
      "posisi_jabatan": "full stack developer",
      "bidang_industri": "teknologi informasi",
      "Lokasi_kerja": "Jakarta",
      "Gaji_range": "",
      "Tanggal_Kerja": 2023,
      "Tanggal_selesai": 0,
      "Status": "",
      "created_at": 0,
      "Updated_at": 0
    }
  ],
  "message": "Data pekerjaan berhasil diambil"
}
```

❖ Get all Pekerjaan (User)

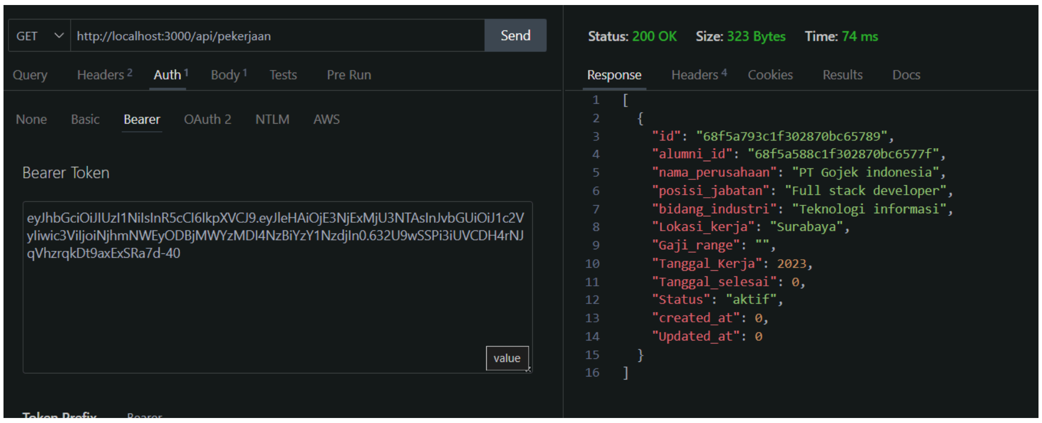


```go
func (r *PekerjaanRepository) GetAllPekerjaan(ctx context.Context) ([]model.Pekerjaan, error) {
    var results []model.Pekerjaan
    cursor, err := r.Col.Find(ctx, bson.M{})
    if err != nil {
        return nil, err
    }
    defer cursor.Close(ctx)

    for cursor.Next(ctx) {
        var p model.Pekerjaan
        if err := cursor.Decode(&p); err != nil {
            return nil, err
        }
        results = append(results, p)
    }
    return results, cursor.Err()
}
```



```go
func (s *PekerjaanService) GetAllPekerjaan(c *gin.Context) {
    ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
    defer cancel()

    results, err := s.Repo.GetAllPekerjaan(ctx)
    if err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Failed to fetch pekerjaan"})
        return
    }

    c.JSON(http.StatusOK, results)
}
```

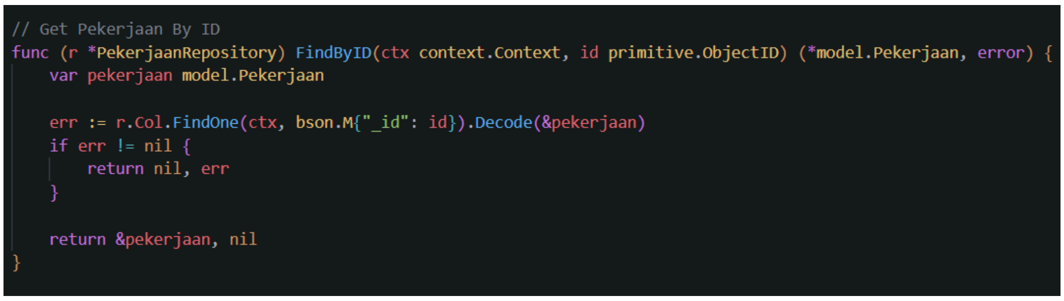
Result:



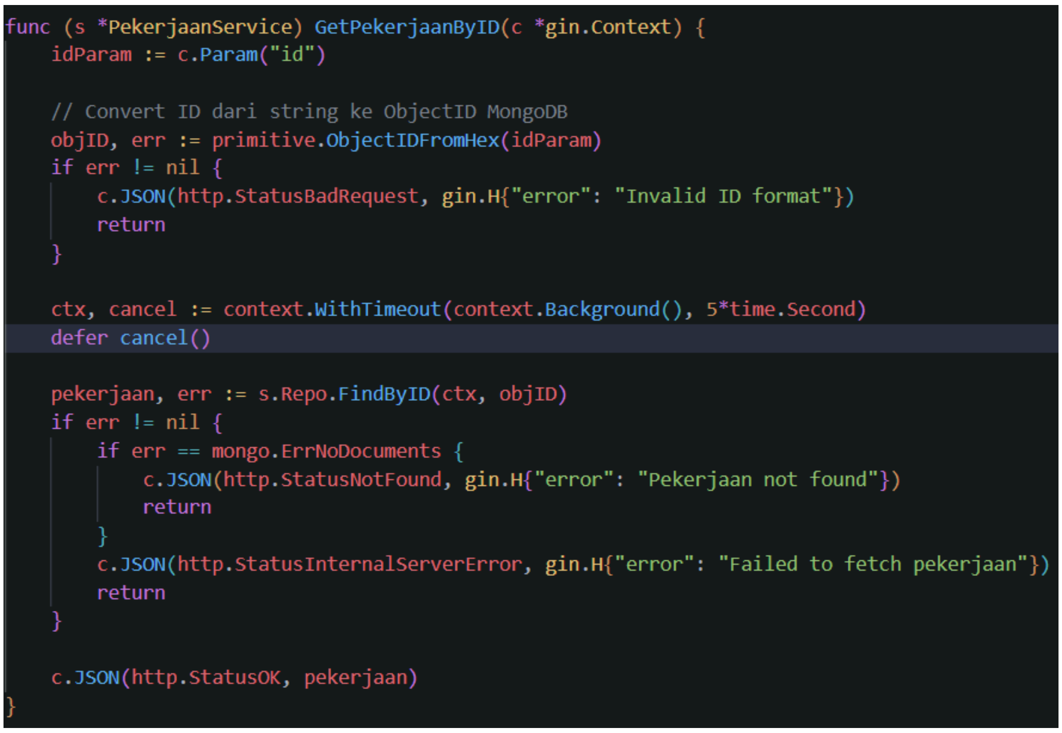
```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3NjExMjU3NTAsInJvbGUiOiJ1c2V
yIiwic3ViIjoiNjhmNWEyODBjMWYzMDI4NzBiYzY1NzdjIn0.632U9wSSPi3iUVCDH4rNJ
qVhzrqkDt9axExSRa7d-40
```

```
[
  {
    "id": "68f5a793c1f302870bc65789",
    "alumni_id": "68f5a588c1f302870bc6577f",
    "nama_perusahaan": "PT Gojek indonesia",
    "posisi_jabatan": "Full stack developer",
    "bidang_industri": "Teknologi informasi",
    "Lokasi_kerja": "Surabaya",
    "Gaji_range": "",
    "Tanggal_Kerja": 2023,
    "Tanggal_selesai": 0,
    "Status": "aktif",
    "created_at": 0,
    "Updated_at": 0
  }
]
```

❖ Get Pekerjaan By ID (User)

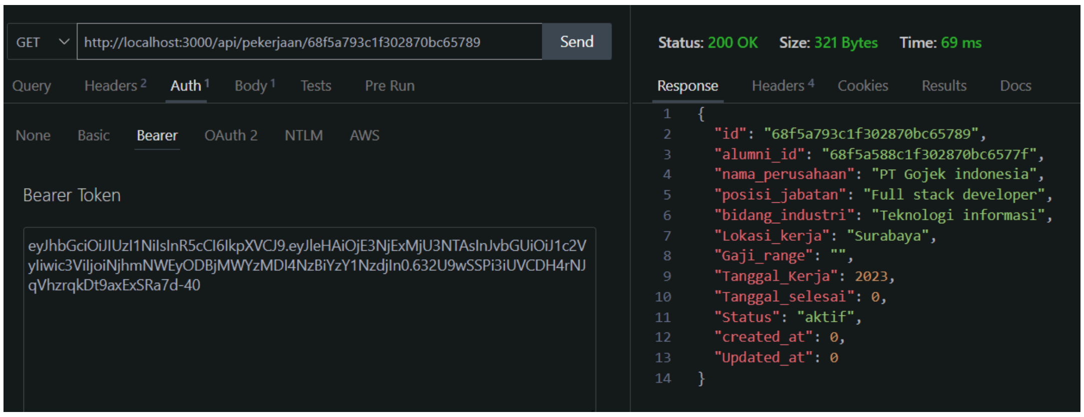


```go
// Get Pekerjaan By ID
func (r *PekerjaanRepository) FindByID(ctx context.Context, id primitive.ObjectID) (*model.Pekerjaan, error) {
    var pekerjaan model.Pekerjaan

    err := r.Col.FindOne(ctx, bson.M{"_id": id}).Decode(&pekerjaan)
    if err != nil {
        return nil, err
    }

    return &pekerjaan, nil
}
```



```go
func (s *PekerjaanService) GetPekerjaanByID(c *gin.Context) {
    idParam := c.Param("id")

    // Convert ID dari string ke ObjectID MongoDB
    objID, err := primitive.ObjectIDFromHex(idParam)
    if err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "Invalid ID format"})
        return
    }

    ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
    defer cancel()

    pekerjaan, err := s.Repo.FindByID(ctx, objID)
    if err != nil {
        if err == mongo.ErrNoDocuments {
            c.JSON(http.StatusNotFound, gin.H{"error": "Pekerjaan not found"})
            return
        }
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Failed to fetch pekerjaan"})
        return
    }

    c.JSON(http.StatusOK, pekerjaan)
}
```

Result:



http://localhost:3000/api/pekerjaan/68f5a793c1f302870bc65789

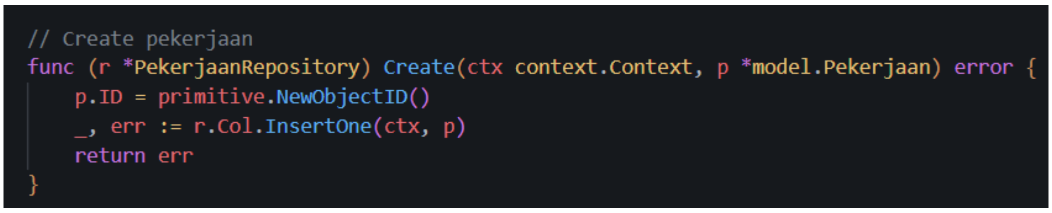
Status: 200 OK Size: 321 Bytes Time: 69 ms

Bearer Token

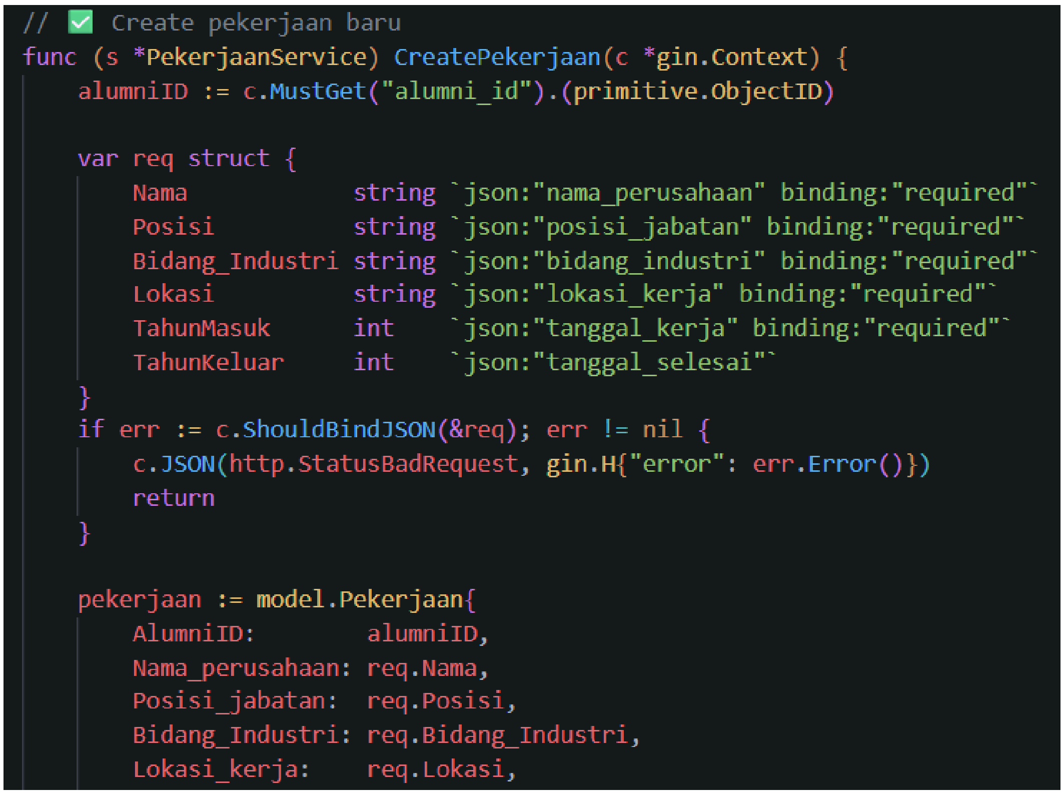
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3NjExMjU3NTAsInJvbGUiOiJ1c2Vyliwic3ViljoiNjhmNWEyODBjMWYzMDI4NzBiYzY1NzdjIn0.632U9wSSPi3iUVCDH4rNJqVhzrqkDt9axExSRa7d-40

```json
{
  "id": "68f5a793c1f302870bc65789",
  "alumni_id": "68f5a588c1f302870bc6577f",
  "nama_perusahaan": "PT Gojek indonesia",
  "posisi_jabatan": "Full stack developer",
  "bidang_industri": "Teknologi informasi",
  "Lokasi_kerja": "Surabaya",
  "Gaji_range": "",
  "Tanggal_Kerja": 2023,
  "Tanggal_selesai": 0,
  "Status": "aktif",
  "created_at": 0,
  "Updated_at": 0
}
```

❖ Tambah Pekerjaan (User)

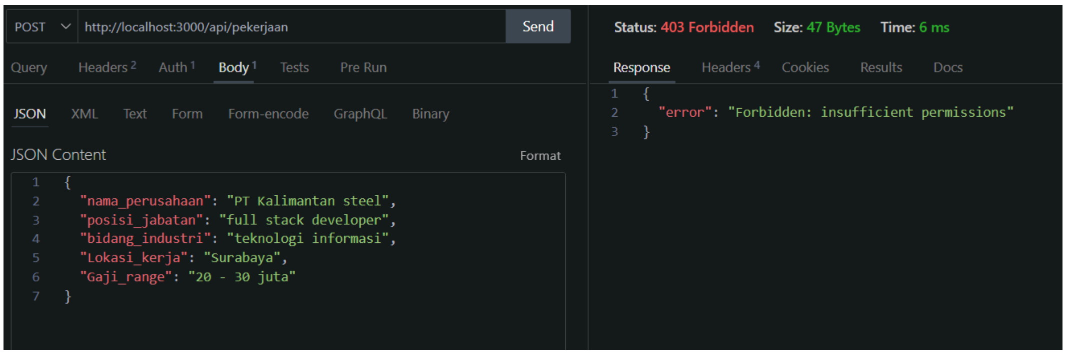


```go
// Create pekerjaan
func (r *PekerjaanRepository) Create(ctx context.Context, p *model.Pekerjaan) error {
    p.ID = primitive.NewObjectID()
    _, err := r.Col.InsertOne(ctx, p)
    return err
}
```



```go
// ✅ Create pekerjaan baru
func (s *PekerjaanService) CreatePekerjaan(c *gin.Context) {
    alumniID := c.MustGet("alumni_id").(primitive.ObjectID)

    var req struct {
        Nama            string `json:"nama_perusahaan" binding:"required"`
        Posisi          string `json:"posisi_jabatan" binding:"required"`
        Bidang_Industri string `json:"bidang_industri" binding:"required"`
        Lokasi          string `json:"lokasi_kerja" binding:"required"`
        TahunMasuk      int    `json:"tanggal_kerja" binding:"required"`
        TahunKeluar     int    `json:"tanggal_selesai"`
    }
    if err := c.ShouldBindJSON(&req); err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": err.Error()})
        return
    }

    pekerjaan := model.Pekerjaan{
        AlumniID:         alumniID,
        Nama_perusahaan: req.Nama,
        Posisi_jabatan:  req.Posisi,
        Bidang_Industri: req.Bidang_Industri,
        Lokasi_kerja:    req.Lokasi,
```

Result:



POST http://localhost:3000/api/pekerjaan Send

Status: 403 Forbidden Size: 47 Bytes Time: 6 ms

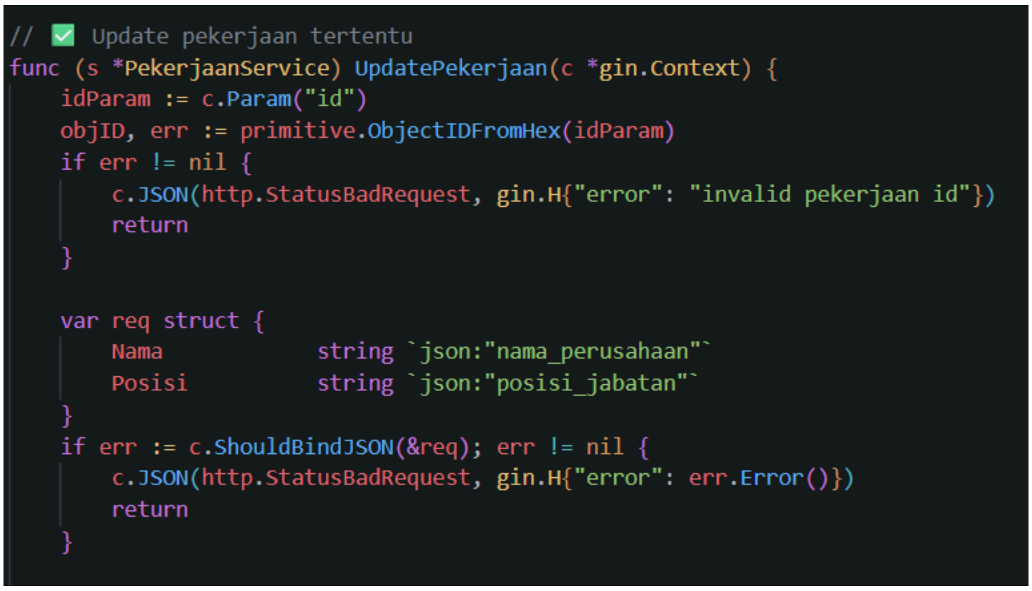
JSON Content

```json
{
  "nama_perusahaan": "PT Kalimantan steel",
  "posisi_jabatan": "full stack developer",
  "bidang_industri": "teknologi informasi",
  "Lokasi_kerja": "Surabaya",
  "Gaji_range": "20 - 30 juta"
}
```

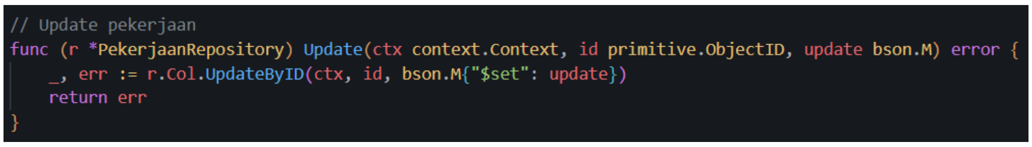
Response

```json
{
  "error": "Forbidden: insufficient permissions"
}
```

❖ Update Pekerjaan (User)

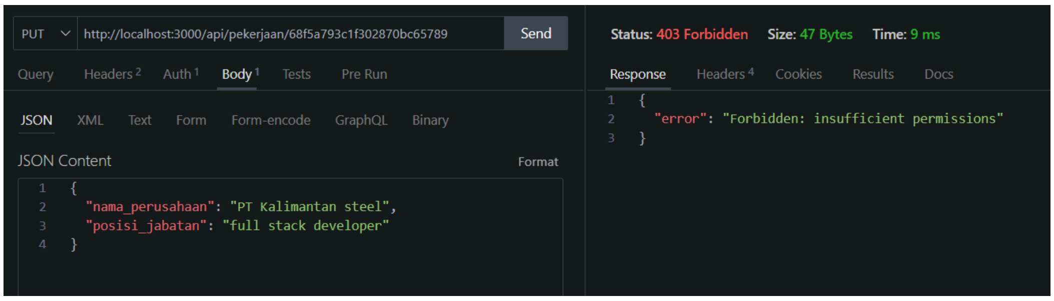


```go
// ✅ Update pekerjaan tertentu
func (s *PekerjaanService) UpdatePekerjaan(c *gin.Context) {
    idParam := c.Param("id")
    objID, err := primitive.ObjectIDFromHex(idParam)
    if err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "invalid pekerjaan id"})
        return
    }

    var req struct {
        Nama            string `json:"nama_perusahaan"`
        Posisi          string `json:"posisi_jabatan"`
    }
    if err := c.ShouldBindJSON(&req); err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": err.Error()})
        return
    }
```



```go
// Update pekerjaan
func (r *PekerjaanRepository) Update(ctx context.Context, id primitive.ObjectID, update bson.M) error {
    _, err := r.Col.UpdateByID(ctx, id, bson.M{"$set": update})
    return err
}
```

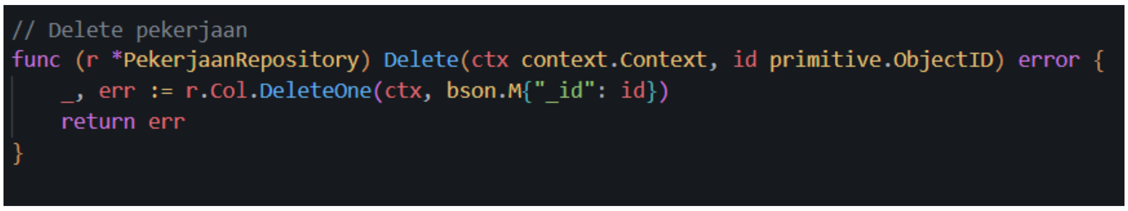
Result:



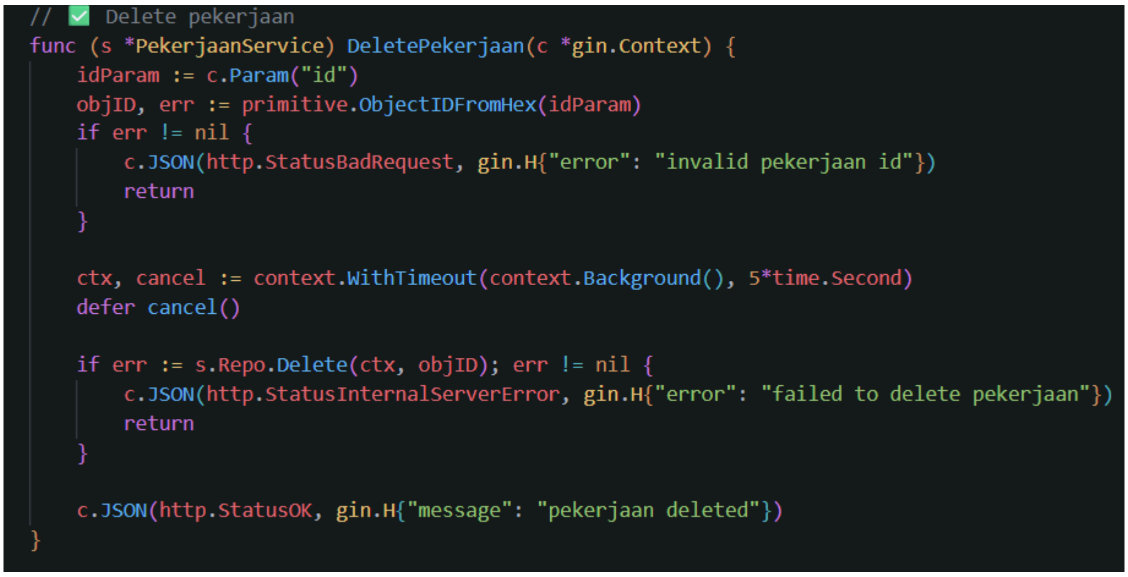
```
PUT  http://localhost:3000/api/pekerjaan/68f5a793c1f302870bc65789   Send

Query   Headers 2   Auth 1   Body 1   Tests   Pre Run

JSON   XML   Text   Form   Form-encode   GraphQL   Binary

JSON Content                                             Format
{
  "nama_perusahaan": "PT Kalimantan steel",
  "posisi_jabatan": "full stack developer"
}
```

```
Status: 403 Forbidden    Size: 47 Bytes    Time: 9 ms

Response   Headers 4   Cookies   Results   Docs
{
  "error": "Forbidden: insufficient permissions"
}
```

❖ Delete Pekerjaan (User)

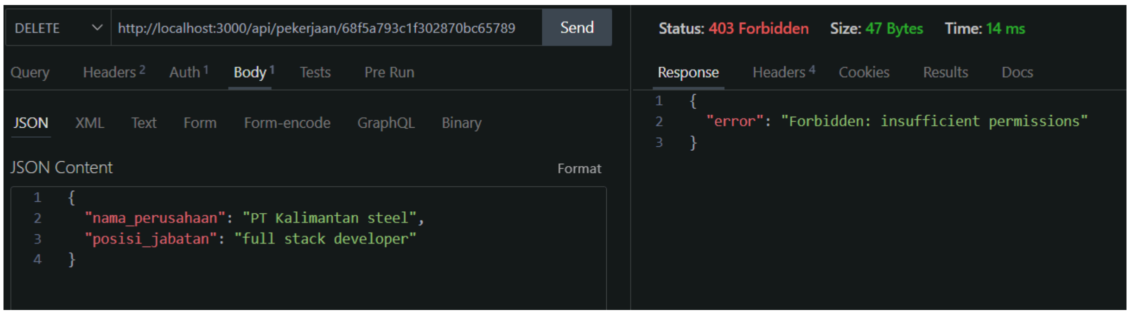


```go
// Delete pekerjaan
func (r *PekerjaanRepository) Delete(ctx context.Context, id primitive.ObjectID) error {
    _, err := r.Col.DeleteOne(ctx, bson.M{"_id": id})
    return err
}
```



```go
// ✅ Delete pekerjaan
func (s *PekerjaanService) DeletePekerjaan(c *gin.Context) {
    idParam := c.Param("id")
    objID, err := primitive.ObjectIDFromHex(idParam)
    if err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "invalid pekerjaan id"})
        return
    }

    ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
    defer cancel()

    if err := s.Repo.Delete(ctx, objID); err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "failed to delete pekerjaan"})
        return
    }

    c.JSON(http.StatusOK, gin.H{"message": "pekerjaan deleted"})
}
```

Result:



DELETE http://localhost:3000/api/pekerjaan/68f5a793c1f302870bc65789 Send

Status: 403 Forbidden Size: 47 Bytes Time: 14 ms

```json
{
  "nama_perusahaan": "PT Kalimantan steel",
  "posisi_jabatan": "full stack developer"
}
```

```json
{
  "error": "Forbidden: insufficient permissions"
}
```

Kesimpulan:

Admin dapat mengakses endpoint POST , PUT dan DELETE , GET sedangkan **User** dapat mengakses endpoint GET.