



Ondokuz Mayıs Üniversitesi

Bilgisayar Mühendisliği

BİL490 - Yazılım Mühendisliği Laboratuvarı

Ders Görevlisi: Dr. Öğr. Üyesi Durmuş Özkan ŞAHİN

Kütüphane Yönetim Sistemi

Uygulama Raporu

İçerik

1	Tasarım Deseni	1
2	Veri Tabanı Seçimi ve Yönetimi	1
3	Veri Tabanı Tabloları	1
4	Tablolar Arası İlişkiler	2
4.1	Kitap Ödünç Alma (borrows Tablosu)	2
4.2	Akademisyen Dallar (branches ve academicianBranch Tabloları)	2
4.3	Kitaplar (books Tablosu)	2
4.4	Kategoriler (categories Tablosu)	2
4.5	Öğrenciler (students Tablosu)	2
4.6	Bölümler (departmans Tablosu)	2
4.7	Akademisyenler (academicians Tablosu)	3
4.8	Memurlar (employee Tablosu)	3
4.9	Makaleler (articles Tablosu)	3
4.10	Makale Yayınlama (academicianArticle Tablosu)	3
5	Giriş Ekranı (WelcomeForm)	3
5.1	Kitap Adına Göre Arama İşlevi	3
5.2	Kitap Kategorisine Göre Arama	3
5.3	Kitap Yılına Göre Arama	6
5.4	İçerik Diline Göre Arama	6
5.5	Yazar İsmine Göre Arama	7
5.6	Giriş Yap ve Kayıt Ol	7
6	Kayıt Ekranı (RegisterForm)	8
7	Giriş Formu (LoginForm)	10
8	Öğrenci Kişisel Sayfası	12

Şekiller Listesi

1	Veri Tabanı Tabloları	1
2	Giriş Ekranı	4
3	Kitap Adına Göre Arama	4
4	Kitap Kategorisine Göre Arama	5
5	Kitap Yılına Göre Arama	6

6	İçerik Diline Göre Arama	6
7	Yazar İsmine Göre Arama	7
8	Giriş Yap ve Kayıt Ol Butonları	7
9	Kayıt Ekranı	8
10	Öğrenci Kayıt Olma	8
11	Akademisyen Kayıt Olma	9
12	Veri Tabanı Tabloları	10
13	Öğrenci Girişi	10
14	Akademisyen Girişi	11
15	Öğrenci Kişisel Sayfası	12
16	Kitap Ödünç Al Butonu	13
17	Ödünç Kitapları Görme	14

Çizelgeler Listesi

1 Tasarım Deseni

Projede her bir bileşeni ayrı ve diğerinden bağımsız geliştirmek için Model View Controller (MVC) mimarisi tercih edilmiştir. Böylelikle test ve oluşabilecek değişim aşamalarında Windows Formlar birbirlerinden bağımsız halde geliştirilmiştir.

2 Veri Tabanı Seçimi ve Yönetimi

Veri tabanı seçiminde sadece veri tabanının ilişkisel bir veri tabanı olması durumu göz önüne alınmıştır. MSSQL tercih edilmiştir. Veri tabanı işlemleri için masaüstü uygulaması geliştirmek üzere C++/CLI .NET aracılığı ile Windows Form'ları geliştirilmiştir.

3 Veri Tabanı Tabloları

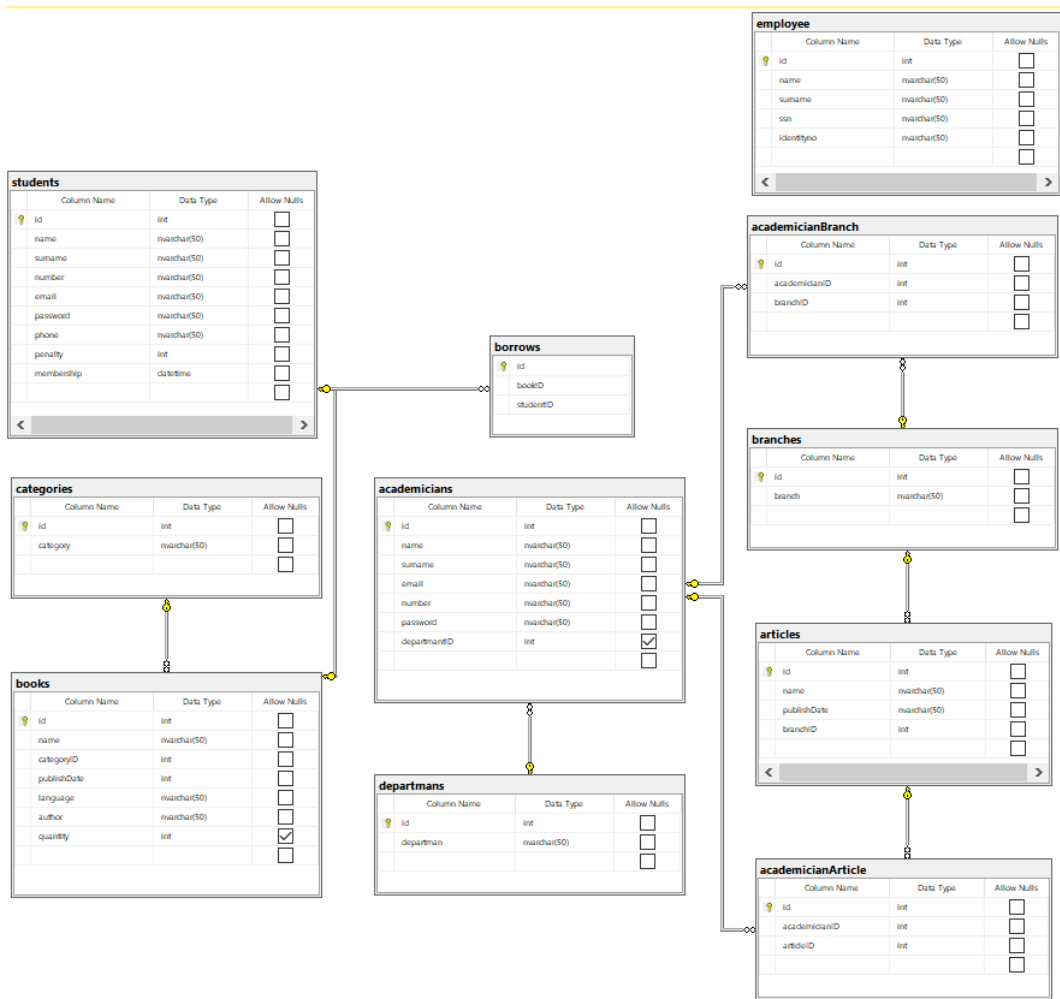


Figure 1: Veri Tabanı Tabloları

4 Tablolar Arası İlişkiler

4.1 Kitap Ödünç Alma (borrows Tablosu)

Bir öğrenci birden fazla kitabı ödünç alabileceği ve bir kitap birden fazla öğrenci tarafından ödünç alınabileceği için öğrenciler (*students*) ve kitaplar (*books*) arasında n-n düzeyinde bir ilişki oluşmaktadır. Bu nedenden dolayı her iki tablonun birincil anahtarı kullanarak yeni bir tablo oluşturulmuştur. Bu tablo *borrows* tablosudur. Bu tablo aracılığı ile sisteme giriş yapan öğrencinin ödünç aldığı kitaplar *students.id* kolonu ile tespit edilerek kullanıcıya gösterilecektir.

4.2 Akademisyen Dalları (branches ve academicianBranch Tabloları)

Bir akademisyen birden fazla dala sahip olabileceği ve birden fazla akademisyen aynı dala sahip olabileceği için akademisyenler (*academicians*) ve dallar (*branches*) arasında n-n düzeyinde bir ilişki oluşmaktadır. Bu nedenden dolayı her iki tablonun birincil anahtarı kullanarak yeni bir tablo oluşturulmuştur. Bu tablo *academicianBranch* tablosudur. Bu tablo aracılığı ile sisteme giriş yapan akademisyenin veya sisteme kayıt olacak akademisyenin dal bilgisi depolanır.

4.3 Kitaplar (books Tablosu)

Kitap nesneleri depolanırken isim, kategori, yayın yılı, içerik dili, yazar ve stok sayısı değerleri her bir kolonu temsil edecek şekilde tablanmıştır. Her bir kitap tek bir kategoriye sahip olacağı ve bir kategori birden fazla kitaba ait olabileceği için 1-n düzeyinde ilişki oluşmuştur. Bu sebepten dolayı kategori tablosu (*categories*) kitap tablosuna (*books*) yabancı anahtar vermiştir.

4.4 Kategoriler (categories Tablosu)

Kategori bilgilerini depolamak için kullanılan tablodur.

4.5 Öğrenciler (students Tablosu)

Öğrenci nesneleri depolanırken isim, soyisim, öğrenci numarası, e-posta, şifre, telefon numarası, ceza ve üye olma tarihi değerleri her bir kolonu temsil edecek şekilde tablanmıştır.

4.6 Bölümler (departmans Tablosu)

Bölüm bilgilerini depolamak için kullanılan tablodur.

4.7 Akademisyenler (academicians Tablosu)

Akademisyen nesneleri depolanırken isim, soyisim, akademisyen numarası, e-posta, şifre, ve bölüm değerleri herbir kolonu temsil edecek şekilde tablolandırılmıştır. Akademisyen (*academicians*) ve bölüm (*departmans*) tablosu arasında her bir akademisyen tek bir bölüme sahip olacağı ve birden fazla akademisyen aynı bölüme ait olabileceği için 1-n ilişkisi oluşmaktadır. Bu nedenle *departmans* tablosu *academicians* tablosuna yabancı anahtar vermektedir.

4.8 Memurlar (employee Tablosu)

Memur nesneleri depolanırken isim, soyisim, sosyal güvenlik numarası ve kimlik numarası değerleri herbir kolonu temsil edecek şekilde tablolandırılmıştır. Memur tablosunun herhangi bir tablo ile ilişkisi bulunmamaktadır.

4.9 Makaleler (articles Tablosu)

Memur nesneleri depolanırken isim, yayın yılı ve dal değerleri herbir kolonu temsil edecek şekilde tablolandırılmıştır.

4.10 Makale Yayınlama (academicianArticle Tablosu)

Her bir makale birden fazla akademisyenin katkıları ile yazılabileceği gibi bir akademisyen tarafında birden fazla makale yayınlabileceği için akademisyen *academicians* ve makale (*articles*) arasında n-n ilişkisi oluşmaktadır. Bu nedenden dolayı her iki tablonun birincil anahtarları kullanılarak *academicianArticle* tablosu oluşturulmuştur. Bu tablo aracılığı ile akademisyenin katkı gösterdiği makaleler bulunabilecektir.

5 Giriş Ekranı (WelcomeForm)

Uygulamanın başlangıç noktasıdır. Kişinin sisteme kayıt olmasına, giriş yapmasına ve kitap arayabilmesine imkan vermektedir.

5.1 Kitap Adına Göre Arama İşlevi

Kitap Adı bölgesine girilen her yeni karakterde sorgu yenilenmekte ve sorgunun geri döngüsü sağ taraftaki *dataGridView*'de sergilenmektedir.

5.2 Kitap Kategorisine Göre Arama

Kullanıcı *comboBox*'dan istediği kategoriye seçer ve bu kategoriye sahip olan kitaplar veri tabanında sorgulanarak *dataGridView*'e aktarılır.

WelcomeForm

Kütüphane Yönetim Sistemi

Giriş Yap
Kayıt Ol

Kitap Adı

Kategori

Yayın Yılı

İçerik Dili

Yazar

Figure 2: Giriş Ekranı

```
private: System::Void textBox1_TextChanged(System::Object^ sender, System
    ↳ ::EventArgs^ e) {
    String^ bookName = textBox1->Text;
    SearchBooks(bookName);
}
void SearchBooks(String^ bookName) {
    String^ query = "SELECT name, categoryID, publishDate, language, author
        ↳ FROM books WHERE name LIKE '%" + bookName + "%'";

    String^ connString = "Data Source=localhost;Initial Catalog=library;
        ↳ Integrated Security=True";
    SqlConnection connection(connString);
    connection.Open();

    SqlCommand^ command = gcnew SqlCommand(query, %connection);

    SqlDataReader^ reader = command->ExecuteReader();

    DataTable^ dataTable = gcnew DataTable();

    dataTable->Load(reader);

    connection.Close();
    reader->Close();

    dataGridView1->DataSource = dataTable;
}
```

Figure 3: Kitap Adına Göre Arama

```

private: System::Void comboBox1_SelectedIndexChanged(System::Object^
    ⇨ sender, System::EventArgs^ e) {
    String^ selectedCategory = comboBox1->SelectedItem->ToString();

    String^ query = "SELECT books.name, categories.category, books.
        ⇨ publishDate, books.language, books.author "
        "FROM books "
        "JOIN categories ON books.categoryID = categories.id "
        "WHERE categories.category = '" + selectedCategory + "'";

    if (selectedCategory == "Tüm Kategoriler") {
        query = "SELECT books.name, categories.category, books.publishDate,
            ⇨ books.language, books.author "
            "FROM books "
            "JOIN categories ON books.categoryID = categories.id";
    }

    SqlConnection^ connection = gcnew SqlConnection("Data Source=localhost;
        ⇨ Initial Catalog=library;Integrated Security=True");
    connection->Open();

    SqlCommand^ command = gcnew SqlCommand(query, connection);

    SqlDataReader^ reader = command->ExecuteReader();

    DataTable^ dataTable = gcnew DataTable();

    dataTable->Load(reader);

    connection->Close();

    dataGridView1->DataSource = dataTable;
}

```

Figure 4: Kitap Kategorisine Göre Arama

5.3 Kitap Yılına Göre Arama

```
private: System::Void textBox2_TextChanged(System::Object^ sender, System
    ↳ ::EventArgs^ e) {
    String^ selectedYear = textBox2->Text;

    String^ query = "SELECT books.name, categories.category, books.
        ↳ publishDate, books.language, books.author "
        "FROM books "
        "JOIN categories ON books.categoryID = categories.id "
        "WHERE books.publishDate = '" + selectedYear + "'";
    SqlConnection^ connection = gcnew SqlConnection("Data Source=localhost;
        ↳ Initial Catalog=library;Integrated Security=True");
    connection->Open();

    SqlCommand^ command = gcnew SqlCommand(query, connection);
    SqlDataReader^ reader = command->ExecuteReader();
    DataTable^ dataTable = gcnew DataTable();
    dataTable->Load(reader);
    connection->Close();
    dataGridView1->DataSource = dataTable;
}
```

Figure 5: Kitap Yılına Göre Arama

5.4 İçerik Diline Göre Arama

```
private: System::Void textBox3_TextChanged(System::Object^ sender, System
    ↳ ::EventArgs^ e) {
    String^ selectedLang = textBox3->Text;

    String^ query = "SELECT books.name, categories.category, books.
        ↳ publishDate, books.language, books.author "
        "FROM books "
        "JOIN categories ON books.categoryID = categories.id "
        "WHERE books.language LIKE '%" + selectedLang + "%'";
    SqlConnection^ connection = gcnew SqlConnection("Data Source=localhost;
        ↳ Initial Catalog=library;Integrated Security=True");
    connection->Open();

    SqlCommand^ command = gcnew SqlCommand(query, connection);
    SqlDataReader^ reader = command->ExecuteReader();
    DataTable^ dataTable = gcnew DataTable();
    dataTable->Load(reader);
    connection->Close();
    dataGridView1->DataSource = dataTable;
}
```

Figure 6: İçerik Diline Göre Arama

5.5 Yazar İsmine Göre Arama

```
private: System::Void textBox4_TextChanged(System::Object^ sender, System
    ↳ EventArgs^ e) {
    String^ selectedAuthor = textBox4->Text;

    String^ query = "SELECT books.name, categories.category, books.
        ↳ publishDate, books.language, books.author "
        "FROM books "
        "JOIN categories ON books.categoryID = categories.id "
        "WHERE books.author LIKE '%" + selectedAuthor + "%'";
    SqlConnection^ connection = gcnew SqlConnection("Data Source=localhost;
        ↳ Initial Catalog=library;Integrated Security=True");
    connection->Open();

    SqlCommand^ command = gcnew SqlCommand(query, connection);
    SqlDataReader^ reader = command->ExecuteReader();
    DataTable^ dataTable = gcnew DataTable();
    dataTable->Load(reader);
    connection->Close();
    dataGridView1->DataSource = dataTable;
}
```

Figure 7: Yazar İsmine Göre Arama

5.6 Giriş Yap ve Kayıt Ol

Kullanıcı Giriş Yap veya Kayıt Ol yazılarına tıkladığında kayıtlanma (*RegisterForm*) veya giriş yapma (*LoginForm*) sayfalarına yönlendirilir.

```
private: System::Void label2_Click(System::Object^ sender, System::
    ↳ EventArgs^ e) {
    LoginForm^ loginPage = gcnew LoginForm();
    //this->Hide();
    loginPage->ShowDialog();
}
private: System::Void label3_Click(System::Object^ sender, System::
    ↳ EventArgs^ e) {
    RegisterForm^ registerPage = gcnew RegisterForm();
    //this->Hide();
    registerPage->ShowDialog();
}
```

Figure 8: Giriş Yap ve Kayıt Ol Butonları

6 Kayıt Ekranı (RegisterForm)

Kullanıcı kayıt ekranına yönlendirildiğinde öncelikle kullanıcı tipini seçmelidir. Seçilen kullanıcı tipine göre birtakım kutucuklar görünmez hale gelecektir ve kullanıcı görünür halde olan kutucukları dolduracaktır.

RegisterForm

Kullanıcı Tipi

İsim

Soyisim

Email

Akademisyen Numarası

Departman

Dal-1

Dal-2

Şifre

Kayıt Ol

İsim

Soyisim

Öğrenci Numarası

Email

Şifre

Telefon

Kayıt Ol

Figure 9: Kayıt Ekranı

```

connection.Open();
String^ query = "INSERT INTO students (name,surname,number,email,password,
    ↪ phone)"
    "VALUES ('" + name + "','" + surname + "','" + number + "','" + email
    ↪ + "','" + password + "','" + phone + "')";
SqlCommand^ command = gcnew SqlCommand(query, % connection);
command->ExecuteNonQuery();
MessageBox::Show("ğÖrenci şıBaaryla Veri ıTabanna Eklendi", "İşlem şııBaarl
    ↪ ", MessageBoxButtons::OK);

```

Figure 10: Öğrenci Kayıt Olma

```

String^ query = "INSERT INTO academicians (name,surname,email,number,
    ⇨ password,departmantID)"
    "VALUES ('"+name+"','"+surname+"','"+email + "','"+ number + "','"+
    ⇨ + password + "','"+ departmentID + "')";
SqlCommand^ command = gcnew SqlCommand(query, %connection);
command->ExecuteNonQuery();

String^ queryGetAcademicianID = "SELECT id FROM academicians WHERE number
    ⇨ = '" + number + "'";
SqlCommand^ commandGetAcademicianID = gcnew SqlCommand(
    ⇨ queryGetAcademicianID, % connection);
reader = commandGetAcademicianID->ExecuteReader();

if (reader->Read()) {
    academicianID = reader->GetInt32(0);
}
reader->Close();

String^ queryInsertBranch = "INSERT INTO academicianBranch (academicianID,
    ⇨ branchID)"
    "VALUES ('"+academicianID+"','"+branchID + "')";
String^ queryInsertBranch2 = "INSERT INTO academicianBranch (academicianID
    ⇨ ,branchID)"
    "VALUES ('" + academicianID + "','"+branchID2 + "')";
SqlCommand^ commandInsertBranch = gcnew SqlCommand(queryInsertBranch, %
    ⇨ connection);
SqlCommand^ commandInsertBranch2 = gcnew SqlCommand(queryInsertBranch2, %
    ⇨ connection);
commandInsertBranch->ExecuteNonQuery();
commandInsertBranch2->ExecuteNonQuery();

```

Figure 11: Akademisyen Kayıt Olma

7 Giriş Formu (LoginForm)

Kullanıcı sisteme giriş yaparken numarasını ve şifresini kullanacaktır. Sorgunun hangi veri tabanı tablosuna yapılacağını ise kullanıcı tipinin seçildiği *comboBox* belirleyecektir.




Figure 12: Veri Tabanı Tabloları

```
String^ queryGetUserProperties = "SELECT * FROM students WHERE number =  
    ↳ @userID AND password = @password";  
SqlCommand commandGetUserProperties(queryGetUserProperties, % connection)  
    ↳ ;  
  
commandGetUserProperties.Parameters->AddWithValue("@userID", userID);  
commandGetUserProperties.Parameters->AddWithValue("@password", password);  
  
reader = commandGetUserProperties.ExecuteReader();  
if (reader->Read()) {  
    Student^ student = gcnew Student;  
    student->ID = reader->GetInt32(0);  
    student->name = reader->GetString(1);  
    student->surname = reader->GetString(2);  
    student->number = reader->GetString(3);  
    student->email = reader->GetString(4);  
    student->phone = reader->GetString(6);  
    student->penalty = reader->GetInt32(7);  
  
    StudentPersonelPage^ studentPage = gcnew StudentPersonelPage(student);  
    this->Hide();  
    studentPage->ShowDialog();  
}
```

Figure 13: Öğrenci Girişi

```

String^ queryryGetUserProperties = "SELECT * FROM academicians WHERE number
    ⇨ = @userID AND password = @password";
SqlCommand commandGetUserProperties(queryryGetUserProperties, % connection)
    ⇨ ;

commandGetUserProperties.Parameters->AddWithValue("@userID", userID);
commandGetUserProperties.Parameters->AddWithValue("@password", password);

reader = commandGetUserProperties.ExecuteReader();
if (reader->Read()) {
    Academician^ academician = gcnew Academician;
    academician->ID = reader->GetInt32(0);
    academician->name = reader->GetString(1);
    academician->surname = reader->GetString(2);
    academician->email = reader->GetString(3);
    academician->number = reader->GetString(4);
    academician->departmentID = reader->GetInt32(6);
    AcademicianPersonelPage^ academicianPage = gcnew
        ⇨ AcademicianPersonelPage(academician);
    this->Hide();
    academicianPage->ShowDialog();
}
else {
    MessageBox::Show("The person you have called can not answer the phone"
        ⇨ , "İİKullanc ıBulunamad", MessageBoxButtons::OK);
}

```

Figure 14: Akademisyen Girişi

8 Öğrenci Kişisel Sayfası

Giriş yapan kişi bir öğrenci ise bu sayfaya yönlendirilecektir. Kullanıcı bu sayfada kitap arayabilir ödünç kitaplarını görebilir ve kitap ödünç alabilir. Öğrenci sol taraftaki arama kutucuklarını kullanarak istediği kitabı bulduktan sonra üstüne tıklar ve *Ödünç Al* butonuna tıklayarak kitabı ödünç alabilir.

The screenshot shows a window titled "StudentPersonalPage". On the left side, there is a search form with the following fields: "İsim", "Soyisim", "Numara", "Ceza", "Kitap Adı", "Kategori" (a dropdown menu), "Yayın Yılı", "İçerik Dili", "Yazar", and "Seçilen". Below these fields is a button labeled "Ödünç Al". To the right of the search form, there are two large, empty rectangular boxes, one above the other, which likely serve as placeholders for search results or book details.

Figure 15: Öğrenci Kişisel Sayfası

```

private: System::Void buttonBorrow_Click(System::Object^ sender, System::
    ↳ EventArgs^ e) {

    String^ connString = "Data Source=localhost;Initial Catalog=library;
        ↳ Integrated Security=True";
    SqlConnection^ connection = gcnew SqlConnection(connString);
    SqlDataReader^ reader;
    int^ booksTID;
    int^ studentTID;
    try
    {
        connection->Open();
        String^ query = "SELECT id FROM books where name = '"+textBox5->Text+"
            ↳ ' ";
        SqlCommand^ command = gcnew SqlCommand(query,connection);
        reader = command->ExecuteReader();
        if (reader->Read()) {
            booksTID = reader->GetInt32(0);
        }
        reader->Close();

        String^ queryGetStudentID = "SELECT id FROM students where number = '"
            ↳ + textBoxNumber->Text + "' ";
        SqlCommand^ commandGetStudentID = gcnew SqlCommand(queryGetStudentID,
            ↳ connection);
        reader = commandGetStudentID->ExecuteReader();
        if (reader->Read()) {
            studentTID = reader->GetInt32(0);
        }
        reader->Close();

        String^ queryInsertBorrow = "INSERT INTO borrows (bookID,studentID)"
            ↳ "VALUES ('" + booksTID + "','"+ studentTID + "')";
        "VALUES ('" + booksTID + "','"+ studentTID + "')";
        "VALUES ('" + booksTID + "','"+ studentTID + "')";
        "VALUES ('" + booksTID + "','"+ studentTID + "')";
        "VALUES ('"+booksTID+"','"+ studentTID + "')";
        SqlCommand^ commandInsertBorrow = gcnew SqlCommand(queryInsertBorrow,
            ↳ connection);
        commandInsertBorrow->ExecuteNonQuery();

        MessageBox::Show("Oldu ıısanrm", "Bilgi ıMesaj", MessageBoxButtons::OK
            ↳ );
    }
    catch (Exception^ &e)
    {
        MessageBox::Show(e->Message, "Hata var ııııgdsn ğyediim",
            ↳ MessageBoxButtons::OK);
    }
    finally {

    }
}

```

Figure 16: Kitap Ödünç Al Butonu


```
DataTable^ borrows = gcnew DataTable();
String^ queryGetBorrows = "SELECT b.name, s.name FROM borrows AS br "
    "JOIN books AS b ON br.bookID = b.id "
    "JOIN students AS s ON br.studentID = s.id "
    "WHERE br.studentID = '" + student->ID + "'";
SqlCommand commandGetBorrows(queryGetBorrows, % connection);
reader = commandGetBorrows.ExecuteReader();
borrows->Load(reader);
dataGridView2->DataSource = borrows;
reader->Close();
```

Figure 17: Ödünç Kitapları Görme