**Adım 1: Proje Klasör Yapısı Oluşturma**

Lexify/

├── Models/ # Veri modelleri

├── ViewModels/ # ViewModel sınıfları

├── Views/ # XAML görünümleri

├── Services/ # Veritabanı ve diğer hizmetler

├── Helpers/ # Yardımcı sınıflar

└── Resources/ # Stil, tema ve diğer kaynaklar

**Adım 2: Her Klasöre Dosyaları Yerleştirme**

# **1.Models Klasörü**

Bu klasörde veritabanı modellerinizi oluşturdu:

## 1.1.Word.cs dosyası güncel içeriği

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lexify.Models

{

public class Word

{

[Key]

public int WordID { get; set; }

[Required]

[StringLength(100)]

public string WordText { get; set; }

[StringLength(50)]

public string WordType { get; set; } // İsim, fiil, sıfat, vb.

[StringLength(50)]

public string Language { get; set; }

[StringLength(20)]

public string ColorCode { get; set; } // Renk kodu (#RRGGBB)

[StringLength(20)]

public string LearningStatus { get; set; } // Yeni, Öğreniliyor, Öğrenildi

public DateTime DateAdded { get; set; }

// Navigation properties

public virtual ICollection<Definition> Definitions { get; set; } = new List<Definition>();

public virtual ICollection<Example> Examples { get; set; } = new List<Example>();

// İlişkili kelimeler (eş/zıt anlamlılar)

public virtual ICollection<WordRelation> RelatedWordsFrom { get; set; } = new List<WordRelation>();

public virtual ICollection<WordRelation> RelatedWordsTo { get; set; } = new List<WordRelation>();

}

}

## 1.2.Definition.cs dosyası güncel içeriği

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lexify.Models

{

public class Definition

{

[Key]

public int DefinitionID { get; set; }

public int WordID { get; set; }

[Required]

public string DefinitionText { get; set; }

// Foreign key relationship

[ForeignKey("WordID")]

public virtual Word Word { get; set; }

}

}

## 1.3.Example.cs dosyası güncel içeriği

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace Lexify.Models

{

public class Example

{

[Key]

public int ExampleID { get; set; }

public int WordID { get; set; }

[Required]

public string ExampleText { get; set; }

// Foreign key relationship

[ForeignKey("WordID")]

public virtual Word Word { get; set; }

}

}

## 1.4.WordRelation.cs dosyası güncel içeriği

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using Lexify.Services;

using System.Windows;

namespace Lexify.Models

{

public class WordRelation

{

[Key]

public int RelationID { get; set; }

public int WordID { get; set; }

public int RelatedWordID { get; set; }

[StringLength(20)]

public string RelationType { get; set; } // "Synonym" veya "Antonym"

// Foreign key relationships

[ForeignKey("WordID")]

public virtual Word Word { get; set; }

[ForeignKey("RelatedWordID")]

public virtual Word RelatedWord { get; set; }

}

}

## 1.5.TestQuestion dosyası güncel içeriği

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

// Models/TestQuestion.cs

namespace Lexify.Models

{

public class TestQuestion

{

public string QuestionText { get; set; }

public List<string> Options { get; set; } = new();

public string CorrectAnswer { get; set; }

public string SelectedAnswer { get; set; }

public bool IsCorrect => SelectedAnswer == CorrectAnswer;

}

}

# **2.Services Klasörü**

Bu klasörde veritabanı işlemlerini yapılıyor servis sınıfları oluşturdu:

## 2.1ApplicationDbContext.cs dosyası güncel içeriği

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Microsoft.EntityFrameworkCore;

using Lexify.Models;

using System.IO;

namespace Lexify.Services

{

public class ApplicationDbContext : DbContext

{

public DbSet<Word> Words { get; set; }

public DbSet<Definition> Definitions { get; set; }

public DbSet<Example> Examples { get; set; }

public DbSet<WordRelation> WordRelations { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

// Projenin yürütme dizinini al

string baseDirectory = AppDomain.CurrentDomain.BaseDirectory;

// Veriler klasörünü oluştur (yoksa)

string dataDirectory = Path.Combine(baseDirectory, "Veriler");

if (!Directory.Exists(dataDirectory))

{

Directory.CreateDirectory(dataDirectory);

}

// Veritabanı dosya yolunu belirle

string dbPath = Path.Combine(dataDirectory, "LexifyDatabase.db");

optionsBuilder.UseSqlite($"Data Source={dbPath}");

}

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

// Kelime ilişkilerini yapılandırma

modelBuilder.Entity<WordRelation>()

.HasOne(wr => wr.Word)

.WithMany(w => w.RelatedWordsFrom)

.HasForeignKey(wr => wr.WordID)

.OnDelete(DeleteBehavior.Restrict);

modelBuilder.Entity<WordRelation>()

.HasOne(wr => wr.RelatedWord)

.WithMany(w => w.RelatedWordsTo)

.HasForeignKey(wr => wr.RelatedWordID)

.OnDelete(DeleteBehavior.Restrict);

}

}

}

## 2.2.DatabaseService.cs dosyası güncel içeriği

using Lexify.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace Lexify.Services

{

public class DatabaseInitializer

{

private readonly ApplicationDbContext \_context;

public DatabaseInitializer(ApplicationDbContext context)

{

\_context = context;

}

public async Task InitializeAsync()

{

// Veritabanını oluştur (yoksa)

\_context.Database.EnsureCreated();

// Eğer veritabanında hiç kelime yoksa örnek veriler ekle

if (!\_context.Words.Any())

{

await SeedSampleDataAsync();

}

}

private async Task SeedSampleDataAsync()

{

// Örnek kelimeler ekle

var words = new List<Word>

{

new Word

{

WordText = "Ephemeral",

WordType = "Adjective",

Language = "English",

ColorCode = "#5E81AC",

LearningStatus = "Yeni",

DateAdded = DateTime.Now.AddDays(-5),

Definitions = new List<Definition>

{

new Definition { DefinitionText = "Lasting for a very short time." },

new Definition { DefinitionText = "Transitory; fleeting." }

},

Examples = new List<Example>

{

new Example { ExampleText = "The ephemeral nature of cherry blossoms makes them more precious." }

}

},

new Word

{

WordText = "Serendipity",

WordType = "Noun",

Language = "English",

ColorCode = "#A3BE8C",

LearningStatus = "Öğreniliyor",

DateAdded = DateTime.Now.AddDays(-3),

Definitions = new List<Definition>

{

new Definition { DefinitionText = "The occurrence and development of events by chance in a happy or beneficial way." }

},

Examples = new List<Example>

{

new Example { ExampleText = "Finding that rare book was pure serendipity." }

}

},

new Word

{

WordText = "Ubiquitous",

WordType = "Adjective",

Language = "English",

ColorCode = "#EBCB8B",

LearningStatus = "Öğrenildi",

DateAdded = DateTime.Now.AddDays(-10),

Definitions = new List<Definition>

{

new Definition { DefinitionText = "Present, appearing, or found everywhere." }

},

Examples = new List<Example>

{

new Example { ExampleText = "Mobile phones have become ubiquitous in modern life." }

}

},

new Word

{

WordText = "Çekirdek",

WordType = "İsim",

Language = "Türkçe",

ColorCode = "#B48EAD",

LearningStatus = "Öğrenildi",

DateAdded = DateTime.Now.AddDays(-7),

Definitions = new List<Definition>

{

new Definition { DefinitionText = "Meyvelerin içinde bulunan, çoğu sert bir kabukla kaplı tohum." },

new Definition { DefinitionText = "Bir şeyin özü, esası." }

},

Examples = new List<Example>

{

new Example { ExampleText = "Elmanın çekirdeğini çıkardı." }

}

},

new Word

{

WordText = "Gelincik",

WordType = "İsim",

Language = "Türkçe",

ColorCode = "#BF616A",

LearningStatus = "Yeni",

DateAdded = DateTime.Now.AddDays(-1),

Definitions = new List<Definition>

{

new Definition { DefinitionText = "Kırmızı renkli, uzun saplı, otsu bir bitki." },

new Definition { DefinitionText = "Bir tür yırtıcı, etçil hayvan." }

},

Examples = new List<Example>

{

new Example { ExampleText = "Tarlalar gelinciklerle dolmuştu." }

}

}

};

// Kelimeleri veritabanına ekle

\_context.Words.AddRange(words);

await \_context.SaveChangesAsync();

// Eş/zıt anlamlı kelime ilişkileri ekle

var wordRelations = new List<WordRelation>

{

new WordRelation

{

WordID = words[0].WordID, // Ephemeral

RelatedWordID = words[2].WordID, // Ubiquitous

RelationType = "Antonym"

}

};

\_context.WordRelations.AddRange(wordRelations);

await \_context.SaveChangesAsync();

}

}

}

## 2.3.DatabaseInitializer.cs dosyası güncel içeriği

using Lexify.Models;

using Microsoft.EntityFrameworkCore;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace Lexify.Services

{

public class DatabaseService

{

private readonly ApplicationDbContext \_context;

private readonly DatabaseInitializer \_initializer;

public DatabaseService()

{

\_context = new ApplicationDbContext();

\_initializer = new DatabaseInitializer(\_context);

// Veritabanını başlat

InitializeDatabaseAsync();

}

private async void InitializeDatabaseAsync()

{

await \_initializer.InitializeAsync();

}

// Kelime işlemleri

public async Task<List<Word>> GetAllWordsAsync()

{

return await \_context.Words

.Include(w => w.Definitions)

.Include(w => w.Examples)

.ToListAsync();

}

public async Task<Word> GetWordByIdAsync(int id)

{

return await \_context.Words

.Include(w => w.Definitions)

.Include(w => w.Examples)

.Include(w => w.RelatedWordsFrom)

.ThenInclude(wr => wr.RelatedWord)

.Include(w => w.RelatedWordsTo)

.ThenInclude(wr => wr.Word)

.FirstOrDefaultAsync(w => w.WordID == id);

}

public async Task<Word> AddWordAsync(Word word)

{

\_context.Words.Add(word);

await \_context.SaveChangesAsync();

return word;

}

public async Task UpdateWordAsync(Word word)

{

\_context.Entry(word).State = EntityState.Modified;

await \_context.SaveChangesAsync();

}

public async Task DeleteWordAsync(int id)

{

var word = await \_context.Words.FindAsync(id);

if (word != null)

{

\_context.Words.Remove(word);

await \_context.SaveChangesAsync();

}

}

// Tanım işlemleri

public async Task<Definition> AddDefinitionAsync(Definition definition)

{

\_context.Definitions.Add(definition);

await \_context.SaveChangesAsync();

return definition;

}

// Örnek cümle işlemleri

public async Task<Example> AddExampleAsync(Example example)

{

\_context.Examples.Add(example);

await \_context.SaveChangesAsync();

return example;

}

// Kelime ilişkisi (eş/zıt anlamlı) işlemleri

public async Task<WordRelation> AddWordRelationAsync(WordRelation relation)

{

\_context.WordRelations.Add(relation);

await \_context.SaveChangesAsync();

return relation;

}

}

}

## 2.4.TestGeneratorService dosyası güncel içeriği

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

// Services/TestGeneratorService.cs

using Lexify.Models;

namespace Lexify.Services

{

public class TestGeneratorService

{

private readonly DatabaseService \_databaseService;

private readonly Random \_random = new();

public TestGeneratorService(DatabaseService databaseService)

{

\_databaseService = databaseService;

}

public async Task<List<TestQuestion>> GenerateDefinitionToWordQuestionsAsync(int count = 5)

{

var words = await \_databaseService.GetAllWordsAsync();

var questions = new List<TestQuestion>();

var definitionsPool = words.SelectMany(w => w.Definitions.Select(d => (word: w.WordText, def: d.DefinitionText))).ToList();

for (int i = 0; i < count; i++)

{

var selected = definitionsPool[\_random.Next(definitionsPool.Count)];

var correct = selected.word;

var definition = selected.def;

var incorrectOptions = definitionsPool

.Where(p => p.word != correct)

.OrderBy(\_ => \_random.Next())

.Select(p => p.word)

.Distinct()

.Take(3)

.ToList();

var options = incorrectOptions.Append(correct).OrderBy(\_ => \_random.Next()).ToList();

questions.Add(new TestQuestion

{

QuestionText = $"Tanımı: \"{definition}\" olan kelime nedir?",

Options = options,

CorrectAnswer = correct

});

}

return questions;

}

}

}

# **3.ViewModels Klasörü**

Bu klasörde MVVM yapısını ViewModel'lerini oluşturdu:

## 3.1.ObservableObject.cs dosyası güncel içeriği

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.ComponentModel;

using System.Runtime.CompilerServices;

namespace Lexify.ViewModels

{

public class ObservableObject : INotifyPropertyChanged

{

public event PropertyChangedEventHandler PropertyChanged;

protected virtual void OnPropertyChanged([CallerMemberName] string propertyName = null)

{

PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));

}

protected bool SetProperty<T>(ref T field, T value, [CallerMemberName] string propertyName = null)

{

if (Equals(field, value)) return false;

field = value;

OnPropertyChanged(propertyName);

return true;

}

}

}

## 3.2.RelayCommand.cs dosyası güncel içeriği

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Input;

namespace Lexify.ViewModels

{

public class RelayCommand : ICommand

{

private readonly Action<object> \_execute;

private readonly Func<object, bool> \_canExecute;

public RelayCommand(Action<object> execute, Func<object, bool> canExecute = null)

{

\_execute = execute ?? throw new ArgumentNullException(nameof(execute));

\_canExecute = canExecute;

}

public event EventHandler CanExecuteChanged

{

add { CommandManager.RequerySuggested += value; }

remove { CommandManager.RequerySuggested -= value; }

}

public bool CanExecute(object parameter) => \_canExecute == null || \_canExecute(parameter);

public void Execute(object parameter) => \_execute(parameter);

}

}

## 3.3.MainViewModel.cs dosyası güncel içeriği

using Lexify.Models;

using Lexify.Services;

using System.Windows.Input;

using System.Threading.Tasks;

namespace Lexify.ViewModels

{

public class MainViewModel : ObservableObject

{

private readonly DatabaseService \_databaseService;

private object \_currentView;

public MainViewModel()

{

\_databaseService = new DatabaseService();

// İlk görünüm: Dashboard

CurrentView = new DashboardViewModel(\_databaseService);

// Komutları oluştur

NavigateToDashboardCommand = new RelayCommand(param => NavigateToDashboard());

NavigateToWordsCommand = new RelayCommand(param => NavigateToWords());

NavigateToAddWordCommand = new RelayCommand(param => NavigateToAddWord());

NavigateToTestsCommand = new RelayCommand(param => NavigateToTests());

NavigateToStatisticsCommand = new RelayCommand(param => NavigateToStatistics());

NavigateToSettingsCommand = new RelayCommand(param => NavigateToSettings());

}

public object CurrentView

{

get => \_currentView;

set => SetProperty(ref \_currentView, value);

}

// Navigasyon komutları

public ICommand NavigateToDashboardCommand { get; }

public ICommand NavigateToWordsCommand { get; }

public ICommand NavigateToAddWordCommand { get; }

public ICommand NavigateToTestsCommand { get; }

public ICommand NavigateToStatisticsCommand { get; }

public ICommand NavigateToSettingsCommand { get; }

// Dashboard'a yönlendir

public void NavigateToDashboard()

{

CurrentView = new DashboardViewModel(\_databaseService);

}

// Tüm kelimeleri listeleyen sayfaya yönlendir

public void NavigateToWords()

{

CurrentView = new WordsViewModel(\_databaseService, this);

}

// Yeni kelime ekleme sayfasına yönlendir

public void NavigateToAddWord()

{

var addEditViewModel = new AddEditWordViewModel(\_databaseService);

// Kelime eklenince ya da iptal edilince ana sayfaya dön

addEditViewModel.NavigationCompleted += (sender, e) =>

{

NavigateToDashboard();

};

CurrentView = addEditViewModel;

}

// Kelime düzenleme sayfasına yönlendir

public async void NavigateToEditWord(Word word)

{

if (word == null) return;

var addEditViewModel = new AddEditWordViewModel(\_databaseService);

await addEditViewModel.LoadWordAsync(word.WordID);

addEditViewModel.NavigationCompleted += (sender, e) =>

{

NavigateToDashboard();

};

CurrentView = addEditViewModel;

}

// Kelime detay sayfasına yönlendir

public async void NavigateToWordDetail(Word word)

{

if (word == null) return;

var wordDetailViewModel = new WordDetailViewModel(\_databaseService);

await wordDetailViewModel.LoadWordAsync(word.WordID);

CurrentView = wordDetailViewModel;

}

// Test sayfasına yönlendir

public void NavigateToTests()

{

CurrentView = new TestsViewModel(\_databaseService);

}

// İstatistik sayfasına yönlendir

public void NavigateToStatistics()

{

CurrentView = new StatisticsViewModel(\_databaseService);

}

// Ayarlar sayfasına yönlendir

public void NavigateToSettings()

{

CurrentView = new SettingsViewModel();

}

}

}

## 3.4.DashboardViewModel.cs dosyası güncel içeriği

using Lexify.Models;

using Lexify.Services;

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Input;

namespace Lexify.ViewModels

{

public class DashboardViewModel : ObservableObject

{

private readonly DatabaseService \_databaseService;

private int \_totalWordsCount;

private int \_learnedWordsCount;

private int \_learningWordsCount;

private int \_newWordsCount;

private ObservableCollection<Word> \_recentWords;

private Word \_wordOfTheDay;

public DashboardViewModel(DatabaseService databaseService)

{

\_databaseService = databaseService;

\_recentWords = new ObservableCollection<Word>();

// Verileri yükle

LoadDashboardDataAsync();

}

public int TotalWordsCount

{

get => \_totalWordsCount;

set => SetProperty(ref \_totalWordsCount, value);

}

public int LearnedWordsCount

{

get => \_learnedWordsCount;

set => SetProperty(ref \_learnedWordsCount, value);

}

public int LearningWordsCount

{

get => \_learningWordsCount;

set => SetProperty(ref \_learningWordsCount, value);

}

public int NewWordsCount

{

get => \_newWordsCount;

set => SetProperty(ref \_newWordsCount, value);

}

public ObservableCollection<Word> RecentWords

{

get => \_recentWords;

set => SetProperty(ref \_recentWords, value);

}

public Word WordOfTheDay

{

get => \_wordOfTheDay;

set => SetProperty(ref \_wordOfTheDay, value);

}

private async Task LoadDashboardDataAsync()

{

try

{

var allWords = await \_databaseService.GetAllWordsAsync();

// İstatistikleri güncelle

TotalWordsCount = allWords.Count;

LearnedWordsCount = allWords.Count(w => w.LearningStatus == "Öğrenildi");

LearningWordsCount = allWords.Count(w => w.LearningStatus == "Öğreniliyor");

NewWordsCount = allWords.Count(w => w.LearningStatus == "Yeni");

// Son eklenen kelimeleri yükle (en fazla 5 tane)

var recentWords = allWords

.OrderByDescending(w => w.DateAdded)

.Take(5)

.ToList();

RecentWords.Clear();

foreach (var word in recentWords)

{

RecentWords.Add(word);

}

// Günün kelimesini belirle (rastgele veya algoritma ile)

if (allWords.Any())

{

// Basit bir örnek: Rastgele bir kelime seç

Random random = new Random();

int randomIndex = random.Next(allWords.Count);

WordOfTheDay = allWords[randomIndex];

}

}

catch (Exception ex)

{

// Hata işleme

System.Diagnostics.Debug.WriteLine($"Dashboard veri yükleme hatası: {ex.Message}");

}

}

}

}

## 3.5.WordsViewModel.cs dosyası güncel içeriği

using Lexify.Models;

using Lexify.Services;

using System;

using System.Collections.ObjectModel;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Input;

namespace Lexify.ViewModels

{

public class WordsViewModel : ObservableObject

{

private readonly DatabaseService \_databaseService;

private readonly MainViewModel \_mainViewModel;

private ObservableCollection<Word> \_words;

private string \_searchText;

public WordsViewModel(DatabaseService databaseService, MainViewModel mainViewModel = null)

{

\_databaseService = databaseService;

\_mainViewModel = mainViewModel;

\_words = new ObservableCollection<Word>();

// Komutları başlat

AddNewWordCommand = new RelayCommand(param => AddNewWord());

ViewWordDetailCommand = new RelayCommand(param => ViewWordDetail(param as Word));

TestWordCommand = new RelayCommand(param => TestWord(param as Word));

ShowFiltersCommand = new RelayCommand(param => ShowFilters());

ShowSortOptionsCommand = new RelayCommand(param => ShowSortOptions());

// Verileri yükle

LoadWordsAsync();

}

public ObservableCollection<Word> Words

{

get => \_words;

set => SetProperty(ref \_words, value);

}

public string SearchText

{

get => \_searchText;

set

{

if (SetProperty(ref \_searchText, value))

{

// Arama metni değiştiğinde filtreleme yap

FilterWords();

}

}

}

// Komutlar

public ICommand AddNewWordCommand { get; }

public ICommand ViewWordDetailCommand { get; }

public ICommand TestWordCommand { get; }

public ICommand ShowFiltersCommand { get; }

public ICommand ShowSortOptionsCommand { get; }

private async Task LoadWordsAsync()

{

var allWords = await \_databaseService.GetAllWordsAsync();

Words.Clear();

foreach (var word in allWords)

{

Words.Add(word);

}

}

private void FilterWords()

{

// Basit arama işlevi

if (string.IsNullOrWhiteSpace(SearchText))

{

LoadWordsAsync(); // Arama yoksa tüm kelimeleri yükle

}

else

{

// Filtreleme işlemi - tüm kelimeleri yükle ve sonra filtrele

Task.Run(async () => {

var allWords = await \_databaseService.GetAllWordsAsync();

var filteredWords = allWords.Where(w =>

w.WordText.Contains(SearchText, StringComparison.OrdinalIgnoreCase) ||

(w.Definitions != null && w.Definitions.Any(d => d.DefinitionText.Contains(SearchText, StringComparison.OrdinalIgnoreCase)))

).ToList();

// UI thread'de ObservableCollection'ı güncelle

App.Current.Dispatcher.Invoke(() => {

Words.Clear();

foreach (var word in filteredWords)

{

Words.Add(word);

}

});

});

}

}

private void AddNewWord()

{

if (\_mainViewModel != null)

{

\_mainViewModel.NavigateToAddWord();

}

}

private void ViewWordDetail(Word word)

{

if (word == null || \_mainViewModel == null) return;

\_mainViewModel.NavigateToWordDetail(word);

}

private void TestWord(Word word)

{

if (word == null) return;

// Bu kısmı ileride test sayfası işlevselliği eklendiğinde tamamlayacağız

}

private void ShowFilters()

{

// Filtreleme panelini göster (ileride eklenecek)

}

private void ShowSortOptions()

{

// Sıralama seçeneklerini göster (ileride eklenecek)

}

}

}

## 3.6.AddEditWordViewModel.cs dosyası güncel içeriği

using Lexify.Models;

using Lexify.Services;

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Threading.Tasks;

using System.Windows.Input;

namespace Lexify.ViewModels

{

public class AddEditWordViewModel : ObservableObject

{

private readonly DatabaseService \_databaseService;

private Word \_word;

public AddEditWordViewModel(DatabaseService databaseService)

{

\_databaseService = databaseService;

}

public Word Word

{

get => \_word;

set => SetProperty(ref \_word, value);

}

public async Task LoadWordAsync(int wordId)

{

Word = await \_databaseService.GetWordByIdAsync(wordId);

}

public event EventHandler NavigationCompleted;

protected void OnNavigationCompleted()

{

NavigationCompleted?.Invoke(this, EventArgs.Empty);

}

}

}

## 3.7.WordDetailViewModel.cs dosyası güncel içeriği

using Lexify.Models;

using Lexify.Services;

using System.Threading.Tasks;

namespace Lexify.ViewModels

{

public class WordDetailViewModel : ObservableObject

{

private readonly DatabaseService \_databaseService;

private Word \_word;

public WordDetailViewModel(DatabaseService databaseService)

{

\_databaseService = databaseService;

}

public Word Word

{

get => \_word;

set => SetProperty(ref \_word, value);

}

public async Task LoadWordAsync(int wordId)

{

Word = await \_databaseService.GetWordByIdAsync(wordId);

}

}

}

## 3.8.TestsViewModel.cs dosyası güncel içeriği

using Lexify.Services;

namespace Lexify.ViewModels

{

public class TestsViewModel : ObservableObject

{

private readonly DatabaseService \_databaseService;

public TestsViewModel(DatabaseService databaseService)

{

\_databaseService = databaseService;

}

}

}

## 3.9.StatisticsViewModel.cs dosyası güncel içeriği

using Lexify.Services;

namespace Lexify.ViewModels

{

public class StatisticsViewModel : ObservableObject

{

private readonly DatabaseService \_databaseService;

public StatisticsViewModel(DatabaseService databaseService)

{

\_databaseService = databaseService;

}

}

}

## 3.10.SettingsViewModel.cs dosyası güncel içeriği

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lexify.ViewModels

{

public class SettingsViewModel : ObservableObject

{

public SettingsViewModel()

{

}

}

}

## 3.11.TestSessionViewModel dosyası güncel içeriği

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Lexify.Models;

using Lexify.Services;

using System.Collections.ObjectModel;

using System.Windows.Input;

// ViewModels/TestSessionViewModel.cs

namespace Lexify.ViewModels

{

public class TestSessionViewModel : ObservableObject

{

private readonly TestGeneratorService \_testGenerator;

private int \_currentIndex;

private int \_score;

public ObservableCollection<TestQuestion> Questions { get; set; } = new();

public TestQuestion CurrentQuestion => Questions.Count > 0 ? Questions[\_currentIndex] : null;

public int Score

{

get => \_score;

set => SetProperty(ref \_score, value);

}

public ICommand SelectAnswerCommand { get; }

public TestSessionViewModel(DatabaseService dbService)

{

\_testGenerator = new TestGeneratorService(dbService);

SelectAnswerCommand = new RelayCommand(param => SelectAnswer(param?.ToString()));

LoadQuestionsAsync();

}

private async void LoadQuestionsAsync()

{

var generated = await \_testGenerator.GenerateDefinitionToWordQuestionsAsync();

Questions.Clear();

foreach (var q in generated)

Questions.Add(q);

\_currentIndex = 0;

OnPropertyChanged(nameof(CurrentQuestion));

}

private void SelectAnswer(string selected)

{

if (CurrentQuestion == null || selected == null)

return;

CurrentQuestion.SelectedAnswer = selected;

if (CurrentQuestion.IsCorrect)

Score++;

\_currentIndex++;

if (\_currentIndex >= Questions.Count)

{

// Test tamamlandı

// Burada sonucu gösterecek bir mantık eklenebilir

}

OnPropertyChanged(nameof(CurrentQuestion));

}

}

}

## 3.12.PronunciationTestViewModel.cs dosyası güncel içeriği

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Lexify.Helpers;

using Lexify.Services;

namespace Lexify.ViewModels

{

public class PronunciationTestViewModel : ObservableObject

{

private readonly DatabaseService \_databaseService;

public PronunciationTestViewModel(DatabaseService databaseService)

{

\_databaseService = databaseService;

// Ses dosyasını çal, doğru yazılışı seç testinin mantığı buraya yazılır

}

}

}

## 3.13.WordToDefinitionTestViewModel.cs dosyası güncel içeriği

using Lexify.Models;

using Lexify.Services;

using Lexify.Helpers;

using System.Collections.ObjectModel;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Input;

namespace Lexify.ViewModels

{

public class WordToDefinitionTestViewModel : ObservableObject

{

private readonly DatabaseService \_databaseService;

private ObservableCollection<TestQuestion> \_questions = new();

private int \_currentIndex;

public WordToDefinitionTestViewModel(DatabaseService databaseService)

{

\_databaseService = databaseService;

SelectAnswerCommand = new RelayCommand(SelectAnswer);

\_ = LoadQuestionsAsync();

}

public ICommand SelectAnswerCommand { get; }

public TestQuestion? CurrentQuestion =>

\_questions.Count > 0 && \_currentIndex < \_questions.Count ? \_questions[\_currentIndex] : null;

private async Task LoadQuestionsAsync()

{

var words = await \_databaseService.GetAllWordsAsync();

\_questions = new ObservableCollection<TestQuestion>(

words.Where(w => w.Definitions.Any()).Select(w => new TestQuestion

{

WordID = w.WordID,

QuestionText = w.WordText, // Kelime gösterilir

CorrectAnswer = w.Definitions.First().DefinitionText,

Options = words.Where(x => x.Definitions.Any())

.OrderBy(\_ => Guid.NewGuid()).Take(3)

.Select(x => x.Definitions.First().DefinitionText)

.Append(w.Definitions.First().DefinitionText)

.OrderBy(x => Guid.NewGuid())

.ToList()

})

);

\_currentIndex = 0;

OnPropertyChanged(nameof(CurrentQuestion));

}

private void SelectAnswer(object? selected)

{

if (CurrentQuestion == null || selected is not string answer) return;

CurrentQuestion.SelectedAnswer = answer;

OnPropertyChanged(nameof(CurrentQuestion));

Task.Delay(1000).ContinueWith(\_ =>

{

\_currentIndex++;

OnPropertyChanged(nameof(CurrentQuestion));

});

}

}

}

## 3.14.MixedTestViewModel.cs dosyası güncel içeriği

using Lexify.Models;

using Lexify.Services;

using Lexify.Helpers;

using System.Collections.ObjectModel;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Input;

namespace Lexify.ViewModels

{

public class MixedTestViewModel : ObservableObject

{

private readonly DatabaseService \_databaseService;

private ObservableCollection<TestQuestion> \_questions = new();

private int \_currentIndex;

public MixedTestViewModel(DatabaseService databaseService)

{

\_databaseService = databaseService;

SelectAnswerCommand = new RelayCommand(SelectAnswer);

\_ = LoadQuestionsAsync();

}

public ICommand SelectAnswerCommand { get; }

public TestQuestion? CurrentQuestion => \_questions.Count > 0 && \_currentIndex < \_questions.Count ? \_questions[\_currentIndex] : null;

private async Task LoadQuestionsAsync()

{

var words = await \_databaseService.GetAllWordsAsync();

\_questions = new ObservableCollection<TestQuestion>(

words.Select(w => new TestQuestion

{

WordID = w.WordID,

QuestionText = w.Definitions.FirstOrDefault()?.DefinitionText ?? "(Tanım Yok)",

CorrectAnswer = w.WordText,

Options = words.OrderBy(\_ => Guid.NewGuid()).Take(3).Select(x => x.WordText)

.Append(w.WordText).OrderBy(x => Guid.NewGuid()).ToList()

})

);

\_currentIndex = 0;

OnPropertyChanged(nameof(CurrentQuestion));

}

private void SelectAnswer(object? selected)

{

if (CurrentQuestion == null || selected is not string answer) return;

CurrentQuestion.SelectedAnswer = answer;

OnPropertyChanged(nameof(CurrentQuestion));

Task.Delay(1000).ContinueWith(\_ =>

{

\_currentIndex++;

OnPropertyChanged(nameof(CurrentQuestion));

});

}

}

}

# **4.Views Klasörü**

Bu klasörde XAML görünümleri ve xaml.cs dosyaları oluşturdu:

## 4.1.DashboardView.xaml

<UserControl x:Class="Lexify.Views.DashboardView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Lexify.Views"

mc:Ignorable="d"

d:DesignHeight="600" d:DesignWidth="800">

<Grid Background="#ECEFF4" Margin="20">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<!-- Başlık -->

<TextBlock Grid.Row="0"

Text="Ana Sayfa"

FontSize="24"

FontWeight="Bold"

Margin="0,0,0,20"/>

<!-- İstatistik Kartları -->

<Grid Grid.Row="1" Margin="0,0,0,20">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<!-- Toplam Kelimeler -->

<Border Grid.Column="0"

Background="#5E81AC"

CornerRadius="8"

Margin="4"

Padding="15"

MinHeight="100">

<StackPanel VerticalAlignment="Center">

<TextBlock Text="Toplam Kelimeler"

Foreground="White"

FontSize="16"

TextWrapping="Wrap"/>

<TextBlock Text="{Binding TotalWordsCount}"

Foreground="White"

FontSize="28"

FontWeight="Bold"

Margin="0,10,0,0"/>

</StackPanel>

</Border>

<!-- Öğrenilen Kelimeler -->

<Border Grid.Column="1"

Background="#A3BE8C"

CornerRadius="8"

Margin="4"

Padding="15"

MinHeight="100">

<StackPanel VerticalAlignment="Center">

<TextBlock Text="Öğrenilen"

Foreground="White"

FontSize="16"

TextWrapping="Wrap"/>

<TextBlock Text="{Binding LearnedWordsCount}"

Foreground="White"

FontSize="28"

FontWeight="Bold"

Margin="0,10,0,0"/>

</StackPanel>

</Border>

<!-- Öğreniliyor Kelimeler -->

<Border Grid.Column="2"

Background="#EBCB8B"

CornerRadius="8"

Margin="4"

Padding="15"

MinHeight="100">

<StackPanel VerticalAlignment="Center">

<TextBlock Text="Öğreniliyor"

Foreground="White"

FontSize="16"

TextWrapping="Wrap"/>

<TextBlock Text="{Binding LearningWordsCount}"

Foreground="White"

FontSize="28"

FontWeight="Bold"

Margin="0,10,0,0"/>

</StackPanel>

</Border>

<!-- Yeni Kelimeler -->

<Border Grid.Column="3"

Background="#B48EAD"

CornerRadius="8"

Margin="4"

Padding="15"

MinHeight="100">

<StackPanel VerticalAlignment="Center">

<TextBlock Text="Yeni"

Foreground="White"

FontSize="16"

TextWrapping="Wrap"/>

<TextBlock Text="{Binding NewWordsCount}"

Foreground="White"

FontSize="28"

FontWeight="Bold"

Margin="0,10,0,0"/>

</StackPanel>

</Border>

</Grid>

<!-- Orta Kısım -->

<Grid Grid.Row="2">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="2\*"/>

<ColumnDefinition Width="1\*"/>

</Grid.ColumnDefinitions>

<!-- Son Eklenen Kelimeler -->

<Border Grid.Column="0"

Background="White"

CornerRadius="8"

Margin="4,0,8,0"

Padding="20"

BorderBrush="#E5E9F0"

BorderThickness="1">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<TextBlock Grid.Row="0"

Text="Son Eklenen Kelimeler"

FontSize="18"

FontWeight="Bold"

Margin="0,0,0,15"/>

<ListView Grid.Row="1"

ItemsSource="{Binding RecentWords}"

BorderThickness="0"

Background="Transparent"

ScrollViewer.VerticalScrollBarVisibility="Auto"

ScrollViewer.HorizontalScrollBarVisibility="Disabled">

<ListView.ItemContainerStyle>

<Style TargetType="ListViewItem">

<Setter Property="HorizontalContentAlignment" Value="Stretch"/>

<Setter Property="Padding" Value="0"/>

<Setter Property="Margin" Value="0,4"/>

</Style>

</ListView.ItemContainerStyle>

<ListView.ItemTemplate>

<DataTemplate>

<Border Background="#F8F9FC"

CornerRadius="6"

Padding="15"

BorderBrush="#E5E9F0"

BorderThickness="1">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<StackPanel Grid.Column="0">

<TextBlock Text="{Binding WordText}"

FontWeight="Bold"

FontSize="16"

TextWrapping="Wrap"/>

<Grid Margin="0,6,0,0">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Column="0"

Text="{Binding WordType}"

Foreground="#4C566A"

FontSize="12"

Margin="0,0,12,0"/>

<TextBlock Grid.Column="1"

Text="{Binding Language}"

Foreground="#4C566A"

FontStyle="Italic"

FontSize="12"/>

</Grid>

<!-- İlk tanımı göster -->

<TextBlock Text="{Binding Definitions[0].DefinitionText}"

Foreground="#2E3440"

Margin="0,8,0,0"

TextWrapping="Wrap"

MaxHeight="40"

TextTrimming="CharacterEllipsis"/>

</StackPanel>

<Border Grid.Column="1"

Background="{Binding ColorCode}"

Width="10"

Height="Auto"

CornerRadius="5"

Margin="15,0,0,0"/>

</Grid>

</Border>

</DataTemplate>

</ListView.ItemTemplate>

</ListView>

</Grid>

</Border>

<!-- Günün Kelimesi -->

<Border Grid.Column="1"

Background="White"

CornerRadius="8"

Margin="8,0,4,0"

Padding="20"

BorderBrush="#E5E9F0"

BorderThickness="1">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<TextBlock Grid.Row="0"

Text="Günün Kelimesi"

FontSize="18"

FontWeight="Bold"

Margin="0,0,0,15"/>

<ScrollViewer Grid.Row="1"

VerticalScrollBarVisibility="Auto"

HorizontalScrollBarVisibility="Disabled">

<StackPanel Visibility="{Binding WordOfTheDay, Converter={StaticResource NullToVisibilityConverter}}">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Column="0"

Text="{Binding WordOfTheDay.WordText}"

FontSize="22"

FontWeight="Bold"

Margin="0,0,0,12"

TextWrapping="Wrap"/>

<Border Grid.Column="1"

Background="{Binding WordOfTheDay.ColorCode}"

Width="10"

Height="30"

CornerRadius="5"

Margin="10,0,0,0"

VerticalAlignment="Top"/>

</Grid>

<Grid Margin="0,0,0,15">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Column="0"

Text="{Binding WordOfTheDay.WordType}"

FontStyle="Italic"

Foreground="#4C566A"

Margin="0,0,12,0"/>

<TextBlock Grid.Column="1"

Text="{Binding WordOfTheDay.Language}"

FontStyle="Italic"

Foreground="#4C566A"/>

</Grid>

<TextBlock Text="Tanımlar"

FontWeight="SemiBold"

Margin="0,0,0,8"/>

<ItemsControl ItemsSource="{Binding WordOfTheDay.Definitions}"

Margin="0,0,0,15">

<ItemsControl.ItemTemplate>

<DataTemplate>

<Border Background="#F8F9FC"

CornerRadius="4"

Margin="0,0,0,5"

Padding="10">

<TextBlock Text="{Binding DefinitionText}"

TextWrapping="Wrap"/>

</Border>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

<TextBlock Text="Örnekler"

FontWeight="SemiBold"

Margin="0,0,0,8"

Visibility="{Binding WordOfTheDay.Examples.Count, Converter={StaticResource NullToVisibilityConverter}}"/>

<ItemsControl ItemsSource="{Binding WordOfTheDay.Examples}">

<ItemsControl.ItemTemplate>

<DataTemplate>

<Border Background="#F8F9FC"

CornerRadius="4"

Margin="0,0,0,5"

Padding="10">

<TextBlock Text="{Binding ExampleText}"

TextWrapping="Wrap"

FontStyle="Italic"/>

</Border>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

</StackPanel>

</ScrollViewer>

</Grid>

</Border>

</Grid>

</Grid>

</UserControl>

### 4.1.1.DashboardView.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Lexify.Views

{

public partial class DashboardView : UserControl

{

public DashboardView()

{

InitializeComponent();

}

}

}

## 4.2.WordsView.xaml

<UserControl x:Class="Lexify.Views.WordsView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Lexify.Views"

mc:Ignorable="d"

d:DesignHeight="600" d:DesignWidth="800">

<Grid Background="#ECEFF4" Margin="10">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<!-- Başlık -->

<TextBlock Grid.Row="0"

Text="Kelimelerim"

FontSize="24"

FontWeight="Bold"

Margin="0,0,0,20"/>

<!-- Arama ve Filtreleme -->

<Grid Grid.Row="1" Margin="0,0,0,15">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<TextBox Grid.Column="0"

Text="{Binding SearchText, UpdateSourceTrigger=PropertyChanged}"

Padding="10"

Margin="0,0,10,0"

FontSize="14"

BorderThickness="1"

BorderBrush="#D8DEE9"/>

<Button Grid.Column="1"

Content="Filtrele"

Padding="15,8"

Background="#5E81AC"

Foreground="White"

BorderThickness="0"/>

</Grid>

<!-- Kelime Listesi -->

<ListView Grid.Row="2"

ItemsSource="{Binding Words}"

BorderThickness="0"

Background="White"

Margin="0,5"

BorderBrush="#E5E9F0">

<ListView.ItemTemplate>

<DataTemplate>

<Border Background="#F8F9FC"

CornerRadius="5"

Margin="0,4"

Padding="15">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<StackPanel Grid.Column="0">

<TextBlock Text="{Binding WordText}"

FontWeight="Bold"

FontSize="16"/>

<TextBlock Text="{Binding WordType}"

Foreground="#4C566A"

FontSize="12"

Margin="0,2,0,0"/>

<TextBlock Text="{Binding Language}"

Foreground="#4C566A"

FontStyle="Italic"

FontSize="12"

Margin="0,2,0,0"/>

<TextBlock Text="{Binding Definitions[0].DefinitionText}"

TextWrapping="Wrap"

Margin="0,5,0,0"/>

</StackPanel>

<Border Grid.Column="1"

Background="{Binding ColorCode}"

Width="8"

CornerRadius="4"

Margin="10,0,0,0"/>

</Grid>

</Border>

</DataTemplate>

</ListView.ItemTemplate>

</ListView>

</Grid>

</UserControl>

### 4.2.1WordsView.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Lexify.Views

{

public partial class WordsView : UserControl

{

public WordsView()

{

InitializeComponent();

}

}

}

## 4.3.AddEditWordView.xaml

<UserControl x:Class="Lexify.Views.AddEditWordView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Lexify.Views"

mc:Ignorable="d"

d:DesignHeight="600" d:DesignWidth="800">

<Grid Background="#ECEFF4">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<!-- Başlık -->

<TextBlock Grid.Row="0"

Text="Yeni Kelime Ekle"

FontSize="24"

FontWeight="Bold"

Margin="20,20,20,20"/>

<!-- Form Alanları -->

<ScrollViewer Grid.Row="1"

VerticalScrollBarVisibility="Auto"

HorizontalScrollBarVisibility="Disabled"

Margin="20,0,20,0">

<StackPanel>

<!-- Kelime Metni -->

<TextBlock Text="Kelime"

FontWeight="Bold"

Margin="0,0,0,8"/>

<TextBox Text="{Binding CurrentWord.WordText, UpdateSourceTrigger=PropertyChanged}"

Padding="12"

FontSize="14"

Margin="0,0,0,20"/>

<Grid Margin="0,0,0,20">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="20"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<!-- Dil Seçimi -->

<StackPanel Grid.Column="0">

<TextBlock Text="Dil"

FontWeight="Bold"

Margin="0,0,0,8"/>

<ComboBox Text="{Binding CurrentWord.Language}"

Padding="12"

FontSize="14">

<ComboBoxItem Content="Türkçe"/>

<ComboBoxItem Content="English"/>

<ComboBoxItem Content="Deutsch"/>

<ComboBoxItem Content="Français"/>

<ComboBoxItem Content="Español"/>

</ComboBox>

</StackPanel>

<!-- Kelime Türü -->

<StackPanel Grid.Column="2">

<TextBlock Text="Tür"

FontWeight="Bold"

Margin="0,0,0,8"/>

<ComboBox Text="{Binding CurrentWord.WordType}"

Padding="12"

FontSize="14">

<ComboBoxItem Content="İsim"/>

<ComboBoxItem Content="Fiil"/>

<ComboBoxItem Content="Sıfat"/>

<ComboBoxItem Content="Zarf"/>

<ComboBoxItem Content="Edat"/>

<ComboBoxItem Content="Bağlaç"/>

<ComboBoxItem Content="Adjective"/>

<ComboBoxItem Content="Noun"/>

<ComboBoxItem Content="Verb"/>

<ComboBoxItem Content="Adverb"/>

</ComboBox>

</StackPanel>

</Grid>

<Grid Margin="0,0,0,20">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="20"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<!-- Renk Seçimi -->

<StackPanel Grid.Column="0">

<TextBlock Text="Renk"

FontWeight="Bold"

Margin="0,0,0,8"/>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Rectangle Grid.Column="0"

Width="40"

Height="40"

Fill="{Binding CurrentWord.ColorCode}"

Stroke="#D8DEE9"

StrokeThickness="1"

Margin="0,0,12,0"

RadiusX="4"

RadiusY="4"/>

<ComboBox Grid.Column="1"

SelectedValuePath="Tag"

SelectedValue="{Binding CurrentWord.ColorCode}"

Padding="12"

FontSize="14">

<ComboBoxItem Content="Mavi" Tag="#5E81AC"/>

<ComboBoxItem Content="Yeşil" Tag="#A3BE8C"/>

<ComboBoxItem Content="Sarı" Tag="#EBCB8B"/>

<ComboBoxItem Content="Mor" Tag="#B48EAD"/>

<ComboBoxItem Content="Kırmızı" Tag="#BF616A"/>

</ComboBox>

</Grid>

</StackPanel>

<!-- Öğrenme Durumu -->

<StackPanel Grid.Column="2">

<TextBlock Text="Öğrenme Durumu"

FontWeight="Bold"

Margin="0,0,0,8"/>

<ComboBox Text="{Binding CurrentWord.LearningStatus}"

Padding="12"

FontSize="14">

<ComboBoxItem Content="Yeni"/>

<ComboBoxItem Content="Öğreniliyor"/>

<ComboBoxItem Content="Öğrenildi"/>

</ComboBox>

</StackPanel>

</Grid>

<!-- Tanımlar -->

<TextBlock Text="Tanımlar"

FontWeight="Bold"

Margin="0,0,0,8"/>

<Border Background="White"

BorderBrush="#D8DEE9"

BorderThickness="1"

Padding="12"

Margin="0,0,0,8"

CornerRadius="4">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<TextBox Grid.Column="0"

Text="{Binding NewDefinitionText, UpdateSourceTrigger=PropertyChanged}"

BorderThickness="0"

Padding="0"

FontSize="14"

VerticalAlignment="Center"

Background="Transparent"/>

<Button Grid.Column="1"

Content="Ekle"

Padding="15,8"

Background="#5E81AC"

Foreground="White"

BorderThickness="0"

Command="{Binding AddDefinitionCommand}"/>

</Grid>

</Border>

<ItemsControl ItemsSource="{Binding Definitions}"

Margin="0,0,0,20"

MaxHeight="200">

<ItemsControl.Template>

<ControlTemplate TargetType="ItemsControl">

<ScrollViewer VerticalScrollBarVisibility="Auto">

<ItemsPresenter />

</ScrollViewer>

</ControlTemplate>

</ItemsControl.Template>

<ItemsControl.ItemTemplate>

<DataTemplate>

<Border Background="#F8F9FC"

CornerRadius="4"

Margin="0,4"

Padding="12">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Column="0"

Text="{Binding DefinitionText}"

TextWrapping="Wrap"/>

<Button Grid.Column="1"

Content="Sil"

Foreground="#BF616A"

Background="Transparent"

BorderThickness="0"

Command="{Binding DataContext.RemoveDefinitionCommand,

RelativeSource={RelativeSource AncestorType=ItemsControl}}"

CommandParameter="{Binding}"/>

</Grid>

</Border>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

<!-- Örnek Cümleler -->

<TextBlock Text="Örnek Cümleler"

FontWeight="Bold"

Margin="0,0,0,8"/>

<Border Background="White"

BorderBrush="#D8DEE9"

BorderThickness="1"

Padding="12"

Margin="0,0,0,8"

CornerRadius="4">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<TextBox Grid.Column="0"

Text="{Binding NewExampleText, UpdateSourceTrigger=PropertyChanged}"

BorderThickness="0"

Padding="0"

FontSize="14"

VerticalAlignment="Center"

Background="Transparent"/>

<Button Grid.Column="1"

Content="Ekle"

Padding="15,8"

Background="#5E81AC"

Foreground="White"

BorderThickness="0"

Command="{Binding AddExampleCommand}"/>

</Grid>

</Border>

<ItemsControl ItemsSource="{Binding Examples}"

Margin="0,0,0,20"

MaxHeight="200">

<ItemsControl.Template>

<ControlTemplate TargetType="ItemsControl">

<ScrollViewer VerticalScrollBarVisibility="Auto">

<ItemsPresenter />

</ScrollViewer>

</ControlTemplate>

</ItemsControl.Template>

<ItemsControl.ItemTemplate>

<DataTemplate>

<Border Background="#F8F9FC"

CornerRadius="4"

Margin="0,4"

Padding="12">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Column="0"

Text="{Binding ExampleText}"

TextWrapping="Wrap"

FontStyle="Italic"/>

<Button Grid.Column="1"

Content="Sil"

Foreground="#BF616A"

Background="Transparent"

BorderThickness="0"

Command="{Binding DataContext.RemoveExampleCommand,

RelativeSource={RelativeSource AncestorType=ItemsControl}}"

CommandParameter="{Binding}"/>

</Grid>

</Border>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

<!-- İlişkili Kelimeler (Gelecek sürümde eklenecek) -->

<TextBlock Text="İlişkili Kelimeler (Eş/Zıt Anlamlılar)"

FontWeight="Bold"

Margin="0,0,0,8"/>

<Border Background="#F8F9FC"

BorderBrush="#D8DEE9"

BorderThickness="1"

Padding="12"

CornerRadius="4"

Margin="0,0,0,20">

<TextBlock Text="Bu özellik yakında eklenecektir."

Foreground="#4C566A"/>

</Border>

</StackPanel>

</ScrollViewer>

<!-- Kaydet Butonu -->

<Grid Grid.Row="2"

Margin="20">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<Button Grid.Column="1"

Content="İptal"

Padding="20,12"

Margin="0,0,10,0"

Background="#E5E9F0"

BorderThickness="0"

Command="{Binding CancelCommand}"/>

<Button Grid.Column="2"

Content="Kaydet"

Padding="20,12"

Background="#5E81AC"

Foreground="White"

BorderThickness="0"

Command="{Binding SaveCommand}"/>

</Grid>

</Grid>

</UserControl>

### 4.3.1.AddEditWordView.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Lexify.Views

{

public partial class AddEditWordView : UserControl

{

public AddEditWordView()

{

InitializeComponent();

}

}

}

## 4.4.WordDetailView.xaml

<UserControl x:Class="Lexify.Views.WordDetailView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Lexify.Views"

mc:Ignorable="d">

<Grid Background="#ECEFF4" Margin="20">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<!-- Başlık ve Butonlar -->

<Grid Grid.Row="0" Margin="0,0,0,20">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<Button Grid.Column="0"

Content="Geri"

Padding="15,10"

Margin="0,0,15,0"

Background="#E5E9F0"

BorderThickness="0"

FontSize="14"/>

<TextBlock Grid.Column="1"

Text="Kelime Detayları"

FontSize="24"

FontWeight="Bold"

VerticalAlignment="Center"/>

<StackPanel Grid.Column="2" Orientation="Horizontal">

<Button Content="Düzenle"

Padding="15,10"

Margin="0,0,10,0"

Background="#EBCB8B"

Foreground="White"

BorderThickness="0"

FontSize="14"/>

<Button Content="Sil"

Padding="15,10"

Background="#BF616A"

Foreground="White"

BorderThickness="0"

FontSize="14"/>

</StackPanel>

</Grid>

<!-- Kelime Detayları -->

<ScrollViewer Grid.Row="1"

VerticalScrollBarVisibility="Auto"

HorizontalScrollBarVisibility="Disabled">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="2\*" MinWidth="300"/>

<ColumnDefinition Width="1\*" MinWidth="200"/>

</Grid.ColumnDefinitions>

<!-- Sol Panel - Ana Bilgiler -->

<Border Grid.Column="0"

Background="White"

CornerRadius="8"

Margin="0,0,10,0"

Padding="25"

BorderBrush="#E5E9F0"

BorderThickness="1">

<StackPanel>

<!-- Kelime Başlığı -->

<Grid Margin="0,0,0,15">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Column="0"

Text="{Binding Word.WordText}"

FontSize="32"

FontWeight="Bold"

TextWrapping="Wrap"/>

<Border Grid.Column="1"

Background="{Binding Word.ColorCode}"

Width="12"

Height="40"

CornerRadius="6"

Margin="15,0,0,0"

VerticalAlignment="Center"/>

</Grid>

<!-- Kelime Bilgileri -->

<Grid Margin="0,0,0,25">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<Border Grid.Column="0"

Background="#F0F1F4"

CornerRadius="4"

Padding="10,5"

Margin="0,0,10,0">

<TextBlock Text="{Binding Word.WordType}"

FontSize="14"

FontWeight="SemiBold"

Foreground="#4C566A"/>

</Border>

<Border Grid.Column="1"

Background="#F0F1F4"

CornerRadius="4"

Padding="10,5">

<TextBlock Text="{Binding Word.Language}"

FontSize="14"

FontWeight="SemiBold"

Foreground="#4C566A"/>

</Border>

</Grid>

<!-- Tanımlar -->

<TextBlock Text="Tanımlar"

FontSize="20"

FontWeight="Bold"

Margin="0,0,0,15"/>

<ItemsControl ItemsSource="{Binding Word.Definitions}" Margin="0,0,0,25">

<ItemsControl.ItemTemplate>

<DataTemplate>

<Border Background="#F8F9FC"

CornerRadius="6"

Margin="0,0,0,8"

Padding="15"

BorderBrush="#E5E9F0"

BorderThickness="1">

<TextBlock Text="{Binding DefinitionText}"

TextWrapping="Wrap"

FontSize="15"

LineHeight="24"/>

</Border>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

<!-- Örnekler -->

<TextBlock Text="Örnek Cümleler"

FontSize="20"

FontWeight="Bold"

Margin="0,0,0,15"

Visibility="{Binding Word.Examples.Count, Converter={StaticResource NullToVisibilityConverter}}"/>

<ItemsControl ItemsSource="{Binding Word.Examples}">

<ItemsControl.ItemTemplate>

<DataTemplate>

<Border Background="#F8F9FC"

CornerRadius="6"

Margin="0,0,0,8"

Padding="15"

BorderBrush="#E5E9F0"

BorderThickness="1">

<TextBlock Text="{Binding ExampleText}"

TextWrapping="Wrap"

FontStyle="Italic"

FontSize="15"

LineHeight="24"/>

</Border>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

</StackPanel>

</Border>

<!-- Sağ Panel - Ek Bilgiler -->

<Border Grid.Column="1"

Background="White"

CornerRadius="8"

Margin="10,0,0,0"

Padding="25"

BorderBrush="#E5E9F0"

BorderThickness="1">

<StackPanel>

<!-- Öğrenme Durumu -->

<TextBlock Text="Öğrenme Durumu"

FontSize="18"

FontWeight="Bold"

Margin="0,0,0,15"/>

<Border Background="#F8F9FC"

CornerRadius="6"

Padding="15"

Margin="0,0,0,25"

BorderBrush="#E5E9F0"

BorderThickness="1">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Column="0"

Text="{Binding Word.LearningStatus}"

FontWeight="SemiBold"

FontSize="15"/>

<!-- Status indicator -->

<Border Grid.Column="1"

Width="10"

Height="10"

CornerRadius="5"

Margin="10,0,0,0"

VerticalAlignment="Center">

<Border.Style>

<Style TargetType="Border">

<Setter Property="Background" Value="#EBCB8B"/>

<Style.Triggers>

<DataTrigger Binding="{Binding Word.LearningStatus}" Value="Öğrenildi">

<Setter Property="Background" Value="#A3BE8C"/>

</DataTrigger>

<DataTrigger Binding="{Binding Word.LearningStatus}" Value="Yeni">

<Setter Property="Background" Value="#B48EAD"/>

</DataTrigger>

</Style.Triggers>

</Style>

</Border.Style>

</Border>

</Grid>

</Border>

<!-- Eş/Zıt Anlamlılar -->

<TextBlock Text="İlişkili Kelimeler"

FontSize="18"

FontWeight="Bold"

Margin="0,0,0,15"/>

<ItemsControl ItemsSource="{Binding Word.RelatedWordsFrom}"

Margin="0,0,0,25">

<ItemsControl.ItemTemplate>

<DataTemplate>

<Border Background="#F8F9FC"

CornerRadius="6"

Margin="0,0,0,8"

Padding="15"

BorderBrush="#E5E9F0"

BorderThickness="1">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<TextBlock Grid.Row="0"

Text="{Binding RelationType}"

FontSize="13"

Foreground="#4C566A"

Margin="0,0,0,5"/>

<TextBlock Grid.Row="1"

Text="{Binding RelatedWord.WordText}"

FontWeight="SemiBold"

FontSize="15"/>

</Grid>

</Border>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

<!-- Ekleme Tarihi -->

<TextBlock Text="Ekleme Tarihi"

FontSize="18"

FontWeight="Bold"

Margin="0,0,0,15"/>

<Border Background="#F8F9FC"

CornerRadius="6"

Padding="15"

BorderBrush="#E5E9F0"

BorderThickness="1">

<TextBlock Text="{Binding Word.DateAdded, StringFormat=dd.MM.yyyy}"

FontSize="15"/>

</Border>

<!-- Test butonu -->

<Button Content="Bu Kelimeyi Test Et"

Margin="0,25,0,0"

Padding="15,10"

Background="#5E81AC"

Foreground="White"

BorderThickness="0"

FontSize="14"

HorizontalAlignment="Stretch"/>

</StackPanel>

</Border>

</Grid>

</ScrollViewer>

</Grid>

</UserControl>

### 4.4.1WordDetailView.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Lexify.Views

{

/// <summary>

/// WordDetailView.xaml etkileşim mantığı

/// </summary>

public partial class WordDetailView : UserControl

{

public WordDetailView()

{

InitializeComponent();

}

}

}

## 4.5.TestsView.xaml

<UserControl x:Class="Lexify.Views.TestsView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Lexify.Views"

mc:Ignorable="d">

<Grid Background="#ECEFF4" Margin="20">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<!-- Başlık -->

<TextBlock Grid.Row="0"

Text="Testler ve Alıştırmalar"

FontSize="24"

FontWeight="Bold"

Margin="0,0,0,20"/>

<!-- Test Kartları -->

<ScrollViewer Grid.Row="1"

HorizontalScrollBarVisibility="Disabled"

VerticalScrollBarVisibility="Auto">

<WrapPanel Orientation="Horizontal"

HorizontalAlignment="Center">

<!-- Karışık Test -->

<Border Background="White"

BorderBrush="#E5E9F0"

BorderThickness="1"

MinWidth="250"

MaxWidth="320"

Height="220"

CornerRadius="10"

Margin="8,8,8,8">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Border Grid.Row="0"

Background="#5E81AC"

Height="10"/>

<StackPanel Grid.Row="1" Margin="20,15">

<TextBlock Text="Karışık Test"

FontSize="20"

FontWeight="Bold"

Margin="0,0,0,12"/>

<TextBlock Text="Kelimelerinizden rastgele sorular içeren bir test çözün."

TextWrapping="Wrap"

Foreground="#4C566A"

FontSize="14"

LineHeight="22"/>

</StackPanel>

<Button Grid.Row="2"

Content="Teste Başla"

Margin="20,0,20,20"

Padding="15,10"

Background="#5E81AC"

Foreground="White"

BorderThickness="0"

FontSize="14"/>

</Grid>

</Border>

<!-- Kelimeden Anlama Testi -->

<Border Background="White"

BorderBrush="#E5E9F0"

BorderThickness="1"

MinWidth="250"

MaxWidth="320"

Height="220"

CornerRadius="10"

Margin="8,8,8,8">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Border Grid.Row="0"

Background="#A3BE8C"

Height="10"/>

<StackPanel Grid.Row="1" Margin="20,15">

<TextBlock Text="Kelimeden Anlama"

FontSize="20"

FontWeight="Bold"

Margin="0,0,0,12"/>

<TextBlock Text="Verilen kelimelerden doğru tanımı seçin."

TextWrapping="Wrap"

Foreground="#4C566A"

FontSize="14"

LineHeight="22"/>

</StackPanel>

<Button Grid.Row="2"

Content="Teste Başla"

Margin="20,0,20,20"

Padding="15,10"

Background="#A3BE8C"

Foreground="White"

BorderThickness="0"

FontSize="14"/>

</Grid>

</Border>

<!-- Tanımdan Kelime Testi -->

<Border Background="White"

BorderBrush="#E5E9F0"

BorderThickness="1"

MinWidth="250"

MaxWidth="320"

Height="220"

CornerRadius="10"

Margin="8,8,8,8">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Border Grid.Row="0"

Background="#EBCB8B"

Height="10"/>

<StackPanel Grid.Row="1" Margin="20,15">

<TextBlock Text="Tanımdan Kelime"

FontSize="20"

FontWeight="Bold"

Margin="0,0,0,12"/>

<TextBlock Text="Verilen tanımın hangi kelimeye ait olduğunu bulun."

TextWrapping="Wrap"

Foreground="#4C566A"

FontSize="14"

LineHeight="22"/>

</StackPanel>

<Button Grid.Row="2"

Content="Teste Başla"

Margin="20,0,20,20"

Padding="15,10"

Background="#EBCB8B"

Foreground="White"

BorderThickness="0"

FontSize="14"/>

</Grid>

</Border>

<!-- Sesletimden Kelime Testi -->

<Border Background="White"

BorderBrush="#E5E9F0"

BorderThickness="1"

MinWidth="250"

MaxWidth="320"

Height="220"

CornerRadius="10"

Margin="8,8,8,8">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Border Grid.Row="0"

Background="#B48EAD"

Height="10"/>

<StackPanel Grid.Row="1" Margin="20,15">

<TextBlock Text="Sesletimden Kelime"

FontSize="20"

FontWeight="Bold"

Margin="0,0,0,12"/>

<TextBlock Text="Duyduğunuz kelimenin doğru yazılışını bulun."

TextWrapping="Wrap"

Foreground="#4C566A"

FontSize="14"

LineHeight="22"/>

</StackPanel>

<Button Grid.Row="2"

Content="Teste Başla"

Margin="20,0,20,20"

Padding="15,10"

Background="#B48EAD"

Foreground="White"

BorderThickness="0"

FontSize="14"/>

</Grid>

</Border>

<!-- Resimden Kelime Testi -->

<Border Background="White"

BorderBrush="#E5E9F0"

BorderThickness="1"

MinWidth="250"

MaxWidth="320"

Height="220"

CornerRadius="10"

Margin="8,8,8,8">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Border Grid.Row="0"

Background="#81A1C1"

Height="10"/>

<StackPanel Grid.Row="1" Margin="20,15">

<TextBlock Text="Resimden Kelime"

FontSize="20"

FontWeight="Bold"

Margin="0,0,0,12"/>

<TextBlock Text="Gösterilen resmin hangi kelimeye ait olduğunu bulun."

TextWrapping="Wrap"

Foreground="#4C566A"

FontSize="14"

LineHeight="22"/>

</StackPanel>

<Button Grid.Row="2"

Content="Yakında Gelecek"

Margin="20,0,20,20"

Padding="15,10"

Background="#D8DEE9"

Foreground="#4C566A"

BorderThickness="0"

FontSize="14"

IsEnabled="False"/>

</Grid>

</Border>

<!-- Cümle Tamamlama Testi -->

<Border Background="White"

BorderBrush="#E5E9F0"

BorderThickness="1"

MinWidth="250"

MaxWidth="320"

Height="220"

CornerRadius="10"

Margin="8,8,8,8">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Border Grid.Row="0"

Background="#88C0D0"

Height="10"/>

<StackPanel Grid.Row="1" Margin="20,15">

<TextBlock Text="Cümle Tamamlama"

FontSize="20"

FontWeight="Bold"

Margin="0,0,0,12"/>

<TextBlock Text="Cümledeki boşluğa en uygun kelimeyi seçin."

TextWrapping="Wrap"

Foreground="#4C566A"

FontSize="14"

LineHeight="22"/>

</StackPanel>

<Button Grid.Row="2"

Content="Yakında Gelecek"

Margin="20,0,20,20"

Padding="15,10"

Background="#D8DEE9"

Foreground="#4C566A"

BorderThickness="0"

FontSize="14"

IsEnabled="False"/>

</Grid>

</Border>

</WrapPanel>

</ScrollViewer>

</Grid>

</UserControl>

### 4.5.1.TestsView.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Lexify.Views

{

/// <summary>

/// TestsView.xaml etkileşim mantığı

/// </summary>

public partial class TestsView : UserControl

{

public TestsView()

{

InitializeComponent();

}

}

}

## 4.6.StatisticsView.xaml

<UserControl x:Class="Lexify.Views.StatisticsView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Lexify.Views"

mc:Ignorable="d">

<Grid Background="#ECEFF4" Margin="20">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<!-- Başlık -->

<TextBlock Grid.Row="0"

Text="İstatistikler"

FontSize="24"

FontWeight="Bold"

Margin="0,0,0,20"/>

<!-- İstatistik Kartları -->

<Grid Grid.Row="1" Margin="0,0,0,20">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<!-- Toplam Kelimeler -->

<Border Grid.Column="0"

Background="#5E81AC"

CornerRadius="8"

Margin="4"

Padding="15"

MinHeight="100">

<StackPanel VerticalAlignment="Center">

<TextBlock Text="Toplam Kelimeler"

Foreground="White"

FontSize="16"

TextWrapping="Wrap"/>

<TextBlock Text="42"

Foreground="White"

FontSize="28"

FontWeight="Bold"

Margin="0,10,0,0"/>

</StackPanel>

</Border>

<!-- Öğrenilen Kelimeler -->

<Border Grid.Column="1"

Background="#A3BE8C"

CornerRadius="8"

Margin="4"

Padding="15"

MinHeight="100">

<StackPanel VerticalAlignment="Center">

<TextBlock Text="Öğrenilen"

Foreground="White"

FontSize="16"

TextWrapping="Wrap"/>

<TextBlock Text="18"

Foreground="White"

FontSize="28"

FontWeight="Bold"

Margin="0,10,0,0"/>

</StackPanel>

</Border>

<!-- Öğreniliyor Kelimeler -->

<Border Grid.Column="2"

Background="#EBCB8B"

CornerRadius="8"

Margin="4"

Padding="15"

MinHeight="100">

<StackPanel VerticalAlignment="Center">

<TextBlock Text="Öğreniliyor"

Foreground="White"

FontSize="16"

TextWrapping="Wrap"/>

<TextBlock Text="15"

Foreground="White"

FontSize="28"

FontWeight="Bold"

Margin="0,10,0,0"/>

</StackPanel>

</Border>

<!-- Yeni Kelimeler -->

<Border Grid.Column="3"

Background="#B48EAD"

CornerRadius="8"

Margin="4"

Padding="15"

MinHeight="100">

<StackPanel VerticalAlignment="Center">

<TextBlock Text="Yeni"

Foreground="White"

FontSize="16"

TextWrapping="Wrap"/>

<TextBlock Text="9"

Foreground="White"

FontSize="28"

FontWeight="Bold"

Margin="0,10,0,0"/>

</StackPanel>

</Border>

</Grid>

<!-- Grafikler -->

<Grid Grid.Row="2">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<!-- Öğrenme İlerlemesi Grafiği -->

<Border Grid.Row="0"

Grid.Column="0"

Background="White"

CornerRadius="8"

Margin="4,4,8,8"

Padding="20"

BorderBrush="#E5E9F0"

BorderThickness="1">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<TextBlock Grid.Row="0"

Text="Öğrenme İlerlemesi"

FontSize="18"

FontWeight="Bold"

Margin="0,0,0,15"/>

<!-- Burada gerçek grafik bileşeni olacak -->

<Border Grid.Row="1"

Background="#F8F9FC"

CornerRadius="6">

<Grid>

<TextBlock Text="Grafik Verisi Yükleniyor..."

VerticalAlignment="Center"

HorizontalAlignment="Center"

Foreground="#4C566A"/>

<!-- Sahte grafik gösterimi -->

<Canvas Opacity="0.5" Margin="20">

<Polyline Points="0,100 20,80 40,85 60,60 80,55 100,40 120,45 140,30 160,20 180,25 200,10"

Stroke="#5E81AC"

StrokeThickness="3"

StrokeDashArray="1,0" />

</Canvas>

</Grid>

</Border>

</Grid>

</Border>

<!-- Dillere Göre Dağılım -->

<Border Grid.Row="0"

Grid.Column="1"

Background="White"

CornerRadius="8"

Margin="8,4,4,8"

Padding="20"

BorderBrush="#E5E9F0"

BorderThickness="1">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<TextBlock Grid.Row="0"

Text="Dillere Göre Dağılım"

FontSize="18"

FontWeight="Bold"

Margin="0,0,0,15"/>

<!-- Burada gerçek grafik bileşeni olacak -->

<Border Grid.Row="1"

Background="#F8F9FC"

CornerRadius="6">

<Grid>

<TextBlock Text="Grafik Verisi Yükleniyor..."

VerticalAlignment="Center"

HorizontalAlignment="Center"

Foreground="#4C566A"/>

<!-- Sahte pasta grafiği gösterimi -->

<Canvas Opacity="0.5" Margin="20">

<Path Fill="#5E81AC">

<Path.Data>

<PathGeometry>

<PathFigure StartPoint="100,100">

<LineSegment Point="100,10"/>

<ArcSegment Point="175,100" Size="90,90" SweepDirection="Clockwise"/>

<LineSegment Point="100,100"/>

</PathFigure>

</PathGeometry>

</Path.Data>

</Path>

<Path Fill="#A3BE8C">

<Path.Data>

<PathGeometry>

<PathFigure StartPoint="100,100">

<LineSegment Point="175,100"/>

<ArcSegment Point="100,175" Size="90,90" SweepDirection="Clockwise"/>

<LineSegment Point="100,100"/>

</PathFigure>

</PathGeometry>

</Path.Data>

</Path>

<Path Fill="#EBCB8B">

<Path.Data>

<PathGeometry>

<PathFigure StartPoint="100,100">

<LineSegment Point="100,175"/>

<ArcSegment Point="25,100" Size="90,90" SweepDirection="Clockwise"/>

<LineSegment Point="100,100"/>

</PathFigure>

</PathGeometry>

</Path.Data>

</Path>

<Path Fill="#B48EAD">

<Path.Data>

<PathGeometry>

<PathFigure StartPoint="100,100">

<LineSegment Point="25,100"/>

<ArcSegment Point="100,10" Size="90,90" SweepDirection="Clockwise"/>

<LineSegment Point="100,100"/>

</PathFigure>

</PathGeometry>

</Path.Data>

</Path>

</Canvas>

</Grid>

</Border>

</Grid>

</Border>

<!-- Kelime Türlerine Göre Dağılım -->

<Border Grid.Row="1"

Grid.Column="0"

Background="White"

CornerRadius="8"

Margin="4,8,8,4"

Padding="20"

BorderBrush="#E5E9F0"

BorderThickness="1">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<TextBlock Grid.Row="0"

Text="Kelime Türlerine Göre Dağılım"

FontSize="18"

FontWeight="Bold"

Margin="0,0,0,15"/>

<!-- Burada gerçek grafik bileşeni olacak -->

<Border Grid.Row="1"

Background="#F8F9FC"

CornerRadius="6">

<Grid>

<TextBlock Text="Grafik Verisi Yükleniyor..."

VerticalAlignment="Center"

HorizontalAlignment="Center"

Foreground="#4C566A"/>

<!-- Sahte bar grafik gösterimi -->

<Canvas Opacity="0.5" Margin="20">

<Rectangle Canvas.Left="20" Canvas.Bottom="10" Width="30" Height="100" Fill="#5E81AC" />

<Rectangle Canvas.Left="70" Canvas.Bottom="10" Width="30" Height="70" Fill="#A3BE8C" />

<Rectangle Canvas.Left="120" Canvas.Bottom="10" Width="30" Height="50" Fill="#EBCB8B" />

<Rectangle Canvas.Left="170" Canvas.Bottom="10" Width="30" Height="30" Fill="#B48EAD" />

</Canvas>

</Grid>

</Border>

</Grid>

</Border>

<!-- Aylık Eklenen Kelimeler -->

<Border Grid.Row="1"

Grid.Column="1"

Background="White"

CornerRadius="8"

Margin="8,8,4,4"

Padding="20"

BorderBrush="#E5E9F0"

BorderThickness="1">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<TextBlock Grid.Row="0"

Text="Aylık Eklenen Kelimeler"

FontSize="18"

FontWeight="Bold"

Margin="0,0,0,15"/>

<!-- Burada gerçek grafik bileşeni olacak -->

<Border Grid.Row="1"

Background="#F8F9FC"

CornerRadius="6">

<Grid>

<TextBlock Text="Grafik Verisi Yükleniyor..."

VerticalAlignment="Center"

HorizontalAlignment="Center"

Foreground="#4C566A"/>

<!-- Sahte çizgi grafik gösterimi -->

<Canvas Opacity="0.5" Margin="20">

<Polyline Points="0,80 30,60 60,70 90,50 120,45 150,55 180,30"

Stroke="#5E81AC"

StrokeThickness="3"

StrokeDashArray="1,0" />

<Ellipse Canvas.Left="0" Canvas.Top="77" Width="6" Height="6" Fill="#5E81AC"/>

<Ellipse Canvas.Left="30" Canvas.Top="57" Width="6" Height="6" Fill="#5E81AC"/>

<Ellipse Canvas.Left="60" Canvas.Top="67" Width="6" Height="6" Fill="#5E81AC"/>

<Ellipse Canvas.Left="90" Canvas.Top="47" Width="6" Height="6" Fill="#5E81AC"/>

<Ellipse Canvas.Left="120" Canvas.Top="42" Width="6" Height="6" Fill="#5E81AC"/>

<Ellipse Canvas.Left="150" Canvas.Top="52" Width="6" Height="6" Fill="#5E81AC"/>

<Ellipse Canvas.Left="180" Canvas.Top="27" Width="6" Height="6" Fill="#5E81AC"/>

</Canvas>

</Grid>

</Border>

</Grid>

</Border>

</Grid>

</Grid>

</UserControl>

### 4.6.1.StatisticsView.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Lexify.Views

{

/// <summary>

/// StatisticsView.xaml etkileşim mantığı

/// </summary>

public partial class StatisticsView : UserControl

{

public StatisticsView()

{

InitializeComponent();

}

}

}

## 4.7.SettingsView.xaml

<UserControl x:Class="Lexify.Views.SettingsView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Lexify.Views"

mc:Ignorable="d">

<Grid Background="#ECEFF4" Margin="20">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<!-- Başlık -->

<TextBlock Grid.Row="0"

Text="Ayarlar"

FontSize="24"

FontWeight="Bold"

Margin="0,0,0,20"/>

<!-- Ayarlar Listesi -->

<ScrollViewer Grid.Row="1"

VerticalScrollBarVisibility="Auto"

HorizontalScrollBarVisibility="Disabled">

<StackPanel>

<!-- Uygulama Ayarları -->

<Border Background="White"

CornerRadius="8"

Margin="0,0,0,20"

Padding="25"

BorderBrush="#E5E9F0"

BorderThickness="1">

<StackPanel>

<TextBlock Text="Uygulama Ayarları"

FontSize="20"

FontWeight="Bold"

Margin="0,0,0,20"/>

<!-- Tema Seçimi -->

<Grid Margin="0,0,0,15">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="180"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Column="0"

Text="Tema"

VerticalAlignment="Center"

FontSize="15"/>

<ComboBox Grid.Column="1"

MinWidth="220"

MaxWidth="350"

HorizontalAlignment="Left"

Padding="12,8"

FontSize="14">

<ComboBoxItem Content="Açık Tema" IsSelected="True"/>

<ComboBoxItem Content="Koyu Tema"/>

<ComboBoxItem Content="Sistem Teması"/>

</ComboBox>

</Grid>

<!-- Dil Seçimi -->

<Grid Margin="0,0,0,15">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="180"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Column="0"

Text="Arayüz Dili"

VerticalAlignment="Center"

FontSize="15"/>

<ComboBox Grid.Column="1"

MinWidth="220"

MaxWidth="350"

HorizontalAlignment="Left"

Padding="12,8"

FontSize="14">

<ComboBoxItem Content="Türkçe" IsSelected="True"/>

<ComboBoxItem Content="English"/>

</ComboBox>

</Grid>

<!-- Yazı Tipi Boyutu -->

<Grid Margin="0,0,0,15">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="180"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Column="0"

Text="Yazı Tipi Boyutu"

VerticalAlignment="Center"

FontSize="15"/>

<ComboBox Grid.Column="1"

MinWidth="220"

MaxWidth="350"

HorizontalAlignment="Left"

Padding="12,8"

FontSize="14">

<ComboBoxItem Content="Küçük"/>

<ComboBoxItem Content="Orta" IsSelected="True"/>

<ComboBoxItem Content="Büyük"/>

</ComboBox>

</Grid>

</StackPanel>

</Border>

<!-- Veri Yönetimi -->

<Border Background="White"

CornerRadius="8"

Margin="0,0,0,20"

Padding="25"

BorderBrush="#E5E9F0"

BorderThickness="1">

<StackPanel>

<TextBlock Text="Veri Yönetimi"

FontSize="20"

FontWeight="Bold"

Margin="0,0,0,20"/>

<!-- Veri Yedekleme -->

<StackPanel Margin="0,0,0,25">

<TextBlock Text="Veri Yedekleme ve Geri Yükleme"

FontSize="16"

FontWeight="SemiBold"

Margin="0,0,0,10"/>

<TextBlock Text="Kelimelerinizi, tanımlarınızı ve diğer verilerinizi yedekleyebilir ve başka bir bilgisayara aktarabilirsiniz."

TextWrapping="Wrap"

Foreground="#4C566A"

Margin="0,0,0,15"/>

<WrapPanel Orientation="Horizontal">

<Button Content="Verileri Yedekle"

Padding="15,10"

Margin="0,0,15,10"

Background="#5E81AC"

Foreground="White"

BorderThickness="0"

FontSize="14"/>

<Button Content="Yedekten Geri Yükle"

Padding="15,10"

Margin="0,0,0,10"

Background="#EBCB8B"

Foreground="White"

BorderThickness="0"

FontSize="14"/>

</WrapPanel>

</StackPanel>

<!-- Verileri Sıfırlama -->

<StackPanel>

<TextBlock Text="Verileri Sıfırlama"

FontSize="16"

FontWeight="SemiBold"

Margin="0,0,0,10"/>

<TextBlock Text="Tüm kelime verilerinizi silip, uygulamayı ilk kurulum durumuna geri döndürebilirsiniz. Bu işlem geri alınamaz!"

TextWrapping="Wrap"

Foreground="#4C566A"

Margin="0,0,0,15"/>

<Button Content="Verileri Sıfırla"

Padding="15,10"

Background="#BF616A"

Foreground="White"

BorderThickness="0"

HorizontalAlignment="Left"

FontSize="14"/>

</StackPanel>

</StackPanel>

</Border>

<!-- Hakkında -->

<Border Background="White"

CornerRadius="8"

Margin="0,0,0,0"

Padding="25"

BorderBrush="#E5E9F0"

BorderThickness="1">

<StackPanel>

<TextBlock Text="Hakkında"

FontSize="20"

FontWeight="Bold"

Margin="0,0,0,20"/>

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<TextBlock Text="Lexify"

FontWeight="Bold"

FontSize="16"/>

<TextBlock Text=" - Kelime Öğrenme Uygulaması"

FontSize="16"/>

</StackPanel>

<TextBlock Text="Sürüm 1.0.0"

Foreground="#4C566A"

Margin="0,0,0,15"

FontSize="14"/>

<TextBlock Text="Lexify, dil öğrenimini kolaylaştırmak için geliştirilmiş bir kelime öğrenme uygulamasıdır. Yeni kelimeler ekleyebilir, öğrenme durumunuzu takip edebilir ve testlerle kelime bilginizi geliştirebilirsiniz."

TextWrapping="Wrap"

Margin="0,0,0,15"

FontSize="15"

LineHeight="24"/>

<TextBlock Text="© 2025 Lexify Team"

Foreground="#4C566A"

HorizontalAlignment="Right"

Margin="0,10,0,0"

FontSize="12"/>

</StackPanel>

</Border>

</StackPanel>

</ScrollViewer>

</Grid>

</UserControl>

### 4.7.1.SettingsView.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Lexify.Views

{

/// <summary>

/// SettingsView.xaml etkileşim mantığı

/// </summary>

public partial class SettingsView : UserControl

{

public SettingsView()

{

InitializeComponent();

}

}

## 4.8.DefinitionToWordTestView.xaml

<UserControl x:Class="Lexify.Views.DefinitionToWordTestView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="600" d:DesignWidth="800">

<Grid Background="#ECEFF4" Margin="20">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<!-- Başlık -->

<TextBlock Grid.Row="0"

Text="Tanımdan Kelime Testi"

FontSize="24"

FontWeight="Bold"

Margin="0,0,0,20"/>

<!-- Soru ve Şıklar -->

<StackPanel Grid.Row="1" VerticalAlignment="Top">

<!-- Soru -->

<TextBlock Text="{Binding CurrentQuestion.QuestionText}"

FontSize="18"

TextWrapping="Wrap"

Margin="0,0,0,10"/>

<!-- Şıklar -->

<ItemsControl ItemsSource="{Binding CurrentQuestion.Options}">

<ItemsControl.ItemContainerStyle>

<Style TargetType="ContentPresenter">

<Setter Property="Margin" Value="0,5"/>

</Style>

</ItemsControl.ItemContainerStyle>

<ItemsControl.ItemTemplate>

<DataTemplate>

<Button Content="{Binding}"

Command="{Binding DataContext.SelectAnswerCommand, RelativeSource={RelativeSource AncestorType=UserControl}}"

CommandParameter="{Binding}"

FontSize="16"

Padding="15"

Background="#5E81AC"

Foreground="White"

BorderBrush="Transparent"

BorderThickness="0"

Cursor="Hand"/>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

</StackPanel>

</Grid>

</UserControl>

### 4.8.1.DefinitionToWordTestView.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Lexify.Views

{

/// <summary>

/// DefinitionToWordTestView.xaml etkileşim mantığı

/// </summary>

public partial class DefinitionToWordTestView : Page

{

public DefinitionToWordTestView()

{

InitializeComponent();

}

}

}

## 4.9.MixedTestView.xaml

<UserControl x:Class="Lexify.Views.MixedTestView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:local="clr-namespace:Lexify.Views"

xmlns:helpers="clr-namespace:Lexify.Helpers"

Background="#ECEFF4" Height="146" Width="193">

<Grid Margin="20">

<StackPanel>

<TextBlock Text="Karışık Test" FontSize="24" FontWeight="Bold" Margin="0,0,0,20"/>

<TextBlock Text="{Binding CurrentQuestion.QuestionText}" FontSize="18" TextWrapping="Wrap" Margin="0,0,0,20"/>

<ItemsControl ItemsSource="{Binding CurrentQuestion.Options}">

<ItemsControl.ItemTemplate>

<DataTemplate>

<Button Content="{Binding}"

Command="{Binding DataContext.SelectAnswerCommand, RelativeSource={RelativeSource AncestorType=UserControl}}"

CommandParameter="{Binding}"

Margin="0,5" Padding="10"

Background="{Binding ., Converter={StaticResource AnswerToBrushConverter}, ConverterParameter={Binding DataContext.CurrentQuestion, RelativeSource={RelativeSource AncestorType=UserControl}}}"

Foreground="White"

BorderThickness="0"

FontSize="16"/>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

</StackPanel>

</Grid>

</UserControl>

### 4.9.1.MixedTestView.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Lexify.Views

{

/// <summary>

/// MixedTestView.xaml etkileşim mantığı

/// </summary>

public partial class MixedTestView : UserControl

{

public MixedTestView()

{

InitializeComponent();

}

}

}

## 4.10.WordToDefinitionTestView.xaml

<UserControl x:Class="Lexify.Views.WordToDefinitionTestView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:local="clr-namespace:Lexify.Views"

Background="#ECEFF4">

<Grid Margin="20">

<StackPanel>

<TextBlock Text="Kelimeden Anlama" FontSize="24" FontWeight="Bold" Margin="0,0,0,20"/>

<TextBlock Text="{Binding CurrentQuestion.QuestionText}" FontSize="18" TextWrapping="Wrap" Margin="0,0,0,20"/>

<ItemsControl ItemsSource="{Binding CurrentQuestion.Options}">

<ItemsControl.ItemTemplate>

<DataTemplate>

<Button Content="{Binding}"

Command="{Binding DataContext.SelectAnswerCommand, RelativeSource={RelativeSource AncestorType=UserControl}}"

CommandParameter="{Binding}"

Margin="0,5" Padding="10"

Background="{Binding ., Converter={StaticResource AnswerToBrushConverter}, ConverterParameter={Binding DataContext.CurrentQuestion, RelativeSource={RelativeSource AncestorType=UserControl}}}"

Foreground="White"

BorderThickness="0"

FontSize="16"/>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

</StackPanel>

</Grid>

</UserControl>

### 4.10.1.WordToDefinitionTestView.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Lexify.Views

{

/// <summary>

/// WordToDefinitionTestView.xaml etkileşim mantığı

/// </summary>

public partial class WordToDefinitionTestView : UserControl

{

public WordToDefinitionTestView()

{

InitializeComponent();

}

}

}

## 4.11.PronunciationTestView.xaml

<UserControl x:Class="Lexify.Views.PronunciationTestView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

<Grid Background="#ECEFF4" Margin="20">

<TextBlock Text="🔊 Sesletimden Kelime Testi"

FontSize="24"

FontWeight="Bold"

Foreground="#2E3440"/>

</Grid>

</UserControl>

### 4.11.1.PronunciationTestView.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Lexify.Views

{

/// <summary>

/// PronunciationTestView.xaml etkileşim mantığı

/// </summary>

public partial class PronunciationTestView : UserControl

{

public PronunciationTestView()

{

InitializeComponent();

}

}

}

## 4.12.TestCard.xaml

<UserControl x:Class="Lexify.Views.TestCard"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

<Border Background="White" BorderBrush="#E5E9F0" BorderThickness="1"

MinWidth="250" MaxWidth="320" Height="220" CornerRadius="10" Margin="8">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Border Grid.Row="0" Background="{Binding BackgroundColor, RelativeSource={RelativeSource AncestorType=UserControl}}" Height="10"/>

<StackPanel Grid.Row="1" Margin="20,15">

<TextBlock Text="{Binding Title, RelativeSource={RelativeSource AncestorType=UserControl}}" FontSize="20" FontWeight="Bold" Margin="0,0,0,12"/>

<TextBlock Text="{Binding Description, RelativeSource={RelativeSource AncestorType=UserControl}}"

TextWrapping="Wrap" Foreground="#4C566A" FontSize="14" LineHeight="22"/>

</StackPanel>

<Button Grid.Row="2" Content="Teste Başla" Margin="20,0,20,20" Padding="15,10"

Background="{Binding BackgroundColor, RelativeSource={RelativeSource AncestorType=UserControl}}"

Foreground="White" BorderThickness="0" FontSize="14"

IsEnabled="{Binding IsEnabled, RelativeSource={RelativeSource AncestorType=UserControl}}"

Command="{Binding Command, RelativeSource={RelativeSource AncestorType=UserControl}}"/>

</Grid>

</Border>

</UserControl>

### 4.12.1.TestCard.xaml.cs

using System.Windows;

using System.Windows.Controls;

using System.Windows.Input;

using System.Windows.Media;

namespace Lexify.Views

{

public partial class TestCard : UserControl

{

public static readonly DependencyProperty TitleProperty = DependencyProperty.Register(nameof(Title), typeof(string), typeof(TestCard));

public static readonly DependencyProperty DescriptionProperty = DependencyProperty.Register(nameof(Description), typeof(string), typeof(TestCard));

public static readonly DependencyProperty BackgroundColorProperty = DependencyProperty.Register(nameof(BackgroundColor), typeof(Brush), typeof(TestCard));

public static readonly DependencyProperty CommandProperty = DependencyProperty.Register(nameof(Command), typeof(ICommand), typeof(TestCard));

public static readonly DependencyProperty IsEnabledProperty = DependencyProperty.Register(nameof(IsEnabled), typeof(bool), typeof(TestCard), new PropertyMetadata(true));

public string Title { get => (string)GetValue(TitleProperty); set => SetValue(TitleProperty, value); }

public string Description { get => (string)GetValue(DescriptionProperty); set => SetValue(DescriptionProperty, value); }

public Brush BackgroundColor { get => (Brush)GetValue(BackgroundColorProperty); set => SetValue(BackgroundColorProperty, value); }

public ICommand Command { get => (ICommand)GetValue(CommandProperty); set => SetValue(CommandProperty, value); }

public new bool IsEnabled { get => (bool)GetValue(IsEnabledProperty); set => SetValue(IsEnabledProperty, value); }

public TestCard()

{

InitializeComponent();

}

}

}

# **5.Helpers Klasörü**

## **5.1.AnswerToBrushConverter.cs**

using System;

using System.Globalization;

using System.Windows.Data;

using System.Windows.Media;

using Lexify.Models;

namespace Lexify.Helpers

{

public class AnswerToBrushConverter : IMultiValueConverter

{

public object Convert(object[] values, Type targetType, object parameter, CultureInfo culture)

{

if (values.Length < 2) return new SolidColorBrush(Color.FromRgb(94, 129, 172));

var selected = values[0] as string;

var question = values[1] as TestQuestion;

if (question == null || string.IsNullOrEmpty(question.SelectedAnswer))

return new SolidColorBrush(Color.FromRgb(94, 129, 172)); // mavi

if (selected == question.CorrectAnswer)

return new SolidColorBrush(Color.FromRgb(163, 190, 140)); // yeşil

if (selected == question.SelectedAnswer)

return new SolidColorBrush(Color.FromRgb(191, 97, 106)); // kırmızı

return new SolidColorBrush(Color.FromRgb(94, 129, 172));

}

public object[] ConvertBack(object value, Type[] targetTypes, object parameter, CultureInfo culture)

{

throw new NotImplementedException();

}

}

}

## **5.2.ObservableObject.cs**

using System.Collections.Generic;

using System.ComponentModel;

using System.Runtime.CompilerServices;

namespace Lexify.Helpers

{

public class ObservableObject : INotifyPropertyChanged

{

public event PropertyChangedEventHandler? PropertyChanged;

protected virtual void OnPropertyChanged([CallerMemberName] string? propertyName = null)

{

if (!string.IsNullOrEmpty(propertyName))

{

PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));

}

}

protected bool SetProperty<T>(ref T field, T value, [CallerMemberName] string? propertyName = null)

{

if (EqualityComparer<T>.Default.Equals(field, value))

return false;

field = value;

OnPropertyChanged(propertyName);

return true;

}

}

}

## **5.3.RelayCommand.cs**

using System;

using System.Windows.Input;

namespace Lexify.Helpers

{

public class RelayCommand : ICommand

{

private readonly Action<object?> \_execute;

private readonly Predicate<object?>? \_canExecute;

public RelayCommand(Action<object?> execute, Predicate<object?>? canExecute = null)

{

\_execute = execute ?? throw new ArgumentNullException(nameof(execute));

\_canExecute = canExecute;

}

public bool CanExecute(object? parameter)

{

return \_canExecute == null || \_canExecute(parameter);

}

public void Execute(object? parameter)

{

\_execute(parameter);

}

public event EventHandler? CanExecuteChanged

{

add => CommandManager.RequerySuggested += value;

remove => CommandManager.RequerySuggested -= value;

}

}

}

# **6.Resources Klasörü**

