

THE UNIVERSITY OF WAIKATO Te Whare Wānanga o Waikato

DEPARTMENT OF COMPUTER SCIENCE Te Tari Rorohiko



2025

Contributors

J. Turner

R. Mercado

V. Moxham-Bettridge

J. Kasmara

© 2025 University of Waikato. All rights reserved. No part of this book may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without prior consent of the Department of Computer Science, University of Waikato.

The course material may be used only for the University's educational purposes. It includes extracts of copyright works copied under copyright licences. You may not copy or distribute any part of this material to any other person, and may print from it only for your own use. You may not make a further copy for any other purpose. Failure to comply with the terms of this warning may expose you to legal action for copyright infringement and/or disciplinary action by the University.

THE SOFTWARE DEVELOPMENT LIFECYCLE: IMPLEMENTATION

Last week you began building the user interface for the Parkway Commute project program in Visual Studio using the C# programming language. Today's exercises will familiarise you with the backend of implementation with a focus on databases.

Backend Development

In our previous session we discussed backend and frontend development. Recall that backend programming is developing the part of the software that the user can't see. In most modern applications today this refers to the functionality of the program. Note that when most people talk about backend development they usually do this in reference to web development. However, this concept can be more generally applied.

For example, consider a simple calculator application. The front end of the application contains buttons for the user to input an equation, the backend will perform the calculation and return the answer for display.



Figure 17: Calculator Application

Most modern applications use what we call the Model-View-Controller software design pattern, or MVC pattern. This separates the software into three clear components: the view which is essentially the user interface of the system; the controller, which sends information between the frontend and backend; and the model, which represents the backend functionality.



Figure 18: MVC Software Design Pattern

The primary benefit of using the MVC pattern is that it separates the parts of the system into clear reusable components. The backend for a web application or mobile application may require the same functionality but different user interfaces. Separating out the functionality in this way allows us to simply swap the view and controller for different platforms. That is, the application is not tied to any one platform, a concept which has become increasingly important as the types of devices we use to run software becomes more diverse. For example, desktop, web, smart phone, smart watch, smart speaker and so on.

Most modern programming languages will implement the MVC pattern when developing software that has a user interface. Consider the programs that you have created in Visual Studio, a form window can be considered a view and the controller is the code that is attached to that form. We can create other C# classes which implement the functionality of a program and allow us to interact with data.

Building a Database

Today's exercises are adapted from the Microsoft C# Visual Studio tutorials. We are going to create a simple application which will "mock" a database setup. Typically, in order to use a

database an application would need to connect to a server. That server would then receive requests to modify the database. Consider the following image.



Figure 19: Database Architecture

A piece of software runs on several different computers. That software connects to the database server where the database is stored. The server maintains the connection between the software and the data, including ensuring that safe and secure connections are established. The database can only be modified via the server, that is, the client application cannot modify the database directly.

In Visual Studio, the Integrated Development Environment (IDE) allows us to use the computer we are running the software on to also be a "local server". This essentially allows us to mock the typical database architecture by using the machine we are running our software on as a "fake" external server for our application. Therefore, all the software can be developed in a local environment without the cost of using an external server.

Building a Database Application

Before you continue working on your Parkway Commute Application, let's demonstrate how to work with a local database in Visual Studio. Open Visual Studio and create a new Windows Form App (.NET Framework) for the C# programming language. Name the project "DatabaseExample". Setup the Local Data Server

1. Open the SQL Server Object Explorer window.



Figure 20: SQL Server Object Explorer Menu



Figure 21: SQL Server Object Explorer

2. Create a New Query by right clicking on the local database instance (localdb).



Figure 22: Create a New Query

SQLQuery1.sql 🛛 🗢 🗙 Form1.cs [Design]		÷ \$
▷ - □ ✓ 周 📑 🕂 펜 master	 ■ a · 5 ■ 	
1		÷

Figure 23: Query Window

3. Copy the <u>Northwind Transact-SQL script¹</u> and paste it into the query window.



Figure 24: Pasted Northwind Script

4. Select the execute button.



If the query executed successfully you should see the following output.

¹ <u>https://raw.githubusercontent.com/MicrosoftDocs/visualstudio-docs/main/docs/data-tools/samples/northwind.sql</u>





SQL Server Object Explorer 🔹 🤻 🛪								
ひ 11 簡								
 SQL Server 								
Icoaldb)\MSSQLLocalDB (SQL Server 15.0.4153 - WAIKATO)								
 Databases 								
System Databases								
 Northwind 								
🔺 🚄 Tables								
System Tables								
External Tables								
dbo.Categories								
dbo.CustomerCustomerDemo								
dbo.CustomerDemographics								
dbo.Customers								
dbo.Employees								
dbo.EmployeeTerritories								
dbo.Order Details								
dbo.Orders								
dbo.Products								
dbo.Region								
dbo.Shippers								
dbo.Suppliers								
dbo.Territories								

Figures 27: Database Added Successfully

5. Be sure to save or copy your server name (e.g. (localdb) \MSSQLLocalDB) as you will need this in the next exercise.

Create the Data Source

Q

1. Open the data sources window.

00	File	e Edit	Vie	w	Git	Project	Build	Debug	Format	Tes	>	Command Window	Ctrl+Alt+A	le
	Wi	ndow I	0	Cod	de			F7			5	Data Sources	Shift+Alt+D	
	•		5	Des	igner			Shi	ift+F7		д *	Load Test Runs		
Sen	Form	n1.cs [De	C	Ope	en m We						G	Data Tools Operations		
ver E			_	Ope	en wi	tn						Microsoft Azure Activity Log		7
xplc		🖳 Form1	M	Solu	ution	Explorer		Cti	rl+Alt+L		-	Diagnostic Events		-
orer				Git	Chan	ges		Cti	rl+0, Ctrl+G		Ê	HDInsight Task List Window		1.0
ਰੂ				Git	Repos	sitory		Cti	rl+0, Ctrl+R		0	HiveServer2 Output Window		
olloo			R	Tea	m Exp	lorer		Cti	rl+ Ctrl+M		Ŷ	Application Insights Search		
ê				Sen	ver Ex	plorer		Cti	rl+Alt+S		B	Live Share		
ata				Dat	a Lake	e Analytic	s Explorer				€	Web Publish Activity		
Sour			12	SQL	Serv	er Object	Explorer	Cti	rl+ Ctrl+S		>	Task Runner Explorer	Ctrl+Alt+Bkspce	
Ces			₫ž	Test	t Explo	orer		Cti	rl+E, T			JSON Outline		
			(C)	Coo	okiecu	itter Explo	orer				đ	Containers		
			司	Boo	okmar	k Windov	v	Ctr	rl+K, Ctrl+W			Device Log		
			2	Call	l Hiera	archy		Cti	rl+Alt+K		>_	Package Manager Console		
			Ô:	Clas	ss Vie	w		Cti	rl+Shift+C		R	IntelliCode		
			0	Cod	de Def	finition W	indow	Cti	rl+∖, D		A	Browser Link Dashboard		
			t	Obj	ject Bi	rowser		Cti	rl+Alt+J			Document Outline	Ctrl+Alt+T	
			Ĝ	Erro	or List			Ct	rl+∖, E		×	Property Manager		
			₿	Out	tput			Cti	rl+Alt+O		¢	Resource View	Ctrl+Shift+E	
			Ê	Tasl	k List			Ct	rl+∖, T			Python Environments	Ctrl+K, Ctrl+`	
			₽	Тоо	lbox			Cti	rl+Alt+X		F 2	Python Interactive Window	Alt+I	
			Q	Not	tificati	ions		Ct	rl+ Ctrl+N		71	Node.js Interactive Window	Ctrl+K, N	
			>_	Terr	minal			Ctr	rl+`		æ	F# Interactive	Ctrl+Alt+F	
				Oth	ner Wi	ndows				•	E4	C# Interactive		prm
	L			Тоо	lbars							Code Metrics Results		
			57	Full	Scree	en		Shi	ift+Alt+Ente	r	~	Python Performance Explorer		on
												Backgro	undimagel avou Tile	

Figure 20: Open Data Sources Window

When you open the window be sure to have your project selected as shown below, otherwise you will have a disabled data sources window.



Figure 21: Data Sources Window

2. Select the "Add New Data Source" button.

역 Add New Data Source	

Figure 22: Add New Data Source

3. When the prompt opens select "Database" option and click the next button.

Data Sourc	e Confi	guration Wi	zard				?	×
i.	Cho	ose a Dat	a Source T	ype				
Where	will the	application	get data fro	m?				
		::⊕						
Datab	ase	Service	Object					
Lets you	l conne	ct to a datal	base and choo	ose the database ob	jects for your applic	ation.		
				< Previous	Next >	Finish	Cancel	

Figure 23: Choose a Data Source Type

4. Select the "Dataset" option and click the next button.

9



Figure 24: Choose a Database Model

5. Select the "New Connection" button. On the window select the "Microsoft SQL Server" option and click continue.

ata Source Configuration Wizar	d			?	×
Choose Your Da	ta Connection				
Which data connection shoul	d your application use to conn	ect to the database?			
			~	New Connection.	
This connection string appears the database. However, storing this sensitive data in the conne	to contain sensitive data (for e sensitive data in the connection ction string?	xample, a password), w on string can be a securi	hich is reo ty risk. Do	quired to connect o you want to inclu	to Ide
 No, exclude sensitive d 	ata from the connection string	. I will set this information	on in my	application code.	
 Yes, include sensitive d 	ata in the connection string.				
Show the connection string	that you will save in the applic	ation			
	< Previous	Next >	Finish	Cancel	

Figure 25: Choose Your Data Connection

Choose Data Source	? ×
Data source: Microsoft Access Database File Microsoft ODBC Data Source Microsoft SQL Server Microsoft SQL Server Database File Oracle Database <other></other>	Description Use this selection to connect to Microsoft SQL Server 2005 or above, or to Microsoft SQL Azure using the .NET Framework Data Provider for SQL Server.
Data provider:	
.NET Framework Data Provider for SQL S $ \sim $	
Always use this selection	Continue Cancel

Figure 26: Choose Data Source

Q

6. Type the name of your local database instance into the Server name field and select the Northwind database. Then click the OK button.

dd Connection		?	×
du connection			~
Enter information t data source and/or	o connect to the selected data source or click "Change" to choo provider.	se a differen	t
Data source:			
Microsoft SQL Sen	/er (SqlClient)	Change	
Server name:			
(localdb)\MSSQLL	ocalDB ~	Refresh	
Log on to the ser	ver		
Authentication:	Windows Authentication		~
(Jean manage			
User name;			
Password:			
	Save my password		
Connect to a data	abase		
Select or entered	er a database name:		
Northwind			~
O Attach a data	base file:		
		Browse	
Logical nam	e;		
		Advanced	
Test Connection	OK	Canad	

Figure 27: Add Connection

7. The connection will now be loaded, click the next button on the remaining window.

Q

lab-tcbd211-06\localdb#4d7b180f.Northwind.dbo	V New Connection
This connection string appears to contain sensitive data (for example, a the database. However, storing sensitive data in the connection string contributes this sensitive data in the connection string?	password), which is required to connect to an be a security risk. Do you want to includ
\bigcirc No, exclude sensitive data from the connection string. I will set t	his information in my application code.
Ves, include sensitive data in the connection string.	
 Yes, include sensitive data in the connection string. Show the connection string that you will save in the application — 	
 Yes, include sensitive data in the connection string. Show the connection string that you will save in the application — 	
 Yes, include sensitive data in the connection string. Show the connection string that you will save in the application — 	
 Yes, include sensitive data in the connection string. Show the connection string that you will save in the application — 	
 Yes, include sensitive data in the connection string. Show the connection string that you will save in the application	

Figure 28: Choose Your Data Connection with String

Ö.

Data Source	e Configuration Wizard	?	\times						
i.	Save the Connection String to the Application Configuration Fil	e							
Storing co connectio	onnection strings in your application configuration file eases maintenance and de on string in the application configuration file, enter a name in the box and then cli want to save the connection string to the application configuration file?	ployment. To save the ck Next.							
✓ Yes, save the connection as:									
NorthwindConnectionString									
NorthwindConnectionString									
	< Previous Next > Fini	sh Cancel							

Figure 29: Save the Connection String

8. On the next window, select the tables option and click the finish button.

Ö.

Data Sourc	e Configuration Wizard	?	\times
ı.	Choose Your Database Objects		
Which d	atabase objects do you want in your dataset?		
	Tables Views Stored Procedures Functions		
DataSet	name:		
Northwi	ndDataSet		
	< Previous Next > Finish	Cancel	

Figure 30: Choose Your Database Objects

Q

If you have completed this successfully you should see the following in your solution explorer.



Figure 31: Database Added to Solution Explorer

View the Data on the Form

1. Make sure your windows form is open and the data sources window is visible. Expand the NorthwindDataset to locate the customers table. Drag and drop the customers table onto your form.



	🕂 Form1											
ł	•	0 of {0}	N 4 ×	•								
		CustomerID	CompanyName	ContactName	ContactTitle	Address	City	Region	Posta			
	•	_	8									
	<								>			
-												

Figure 33: Form1 After Adding Customers Data

You should notice that the form populates automatically with different widgets. This will allow the user to explore the dataset. Try running the application and seeing what happens when you explore the data. Can you add a new customer?

2. Below the form design select the customersTableAdapter. Then select the "Add Query" option in the properties window.

-	Properties	→ ₽ ×
InorthwindDataSet	Customers lableAdapter	Databasetxampie.NorthwindData34 •
ि tableAdapterManager ि customersBindingNavigator Output - म् ×	Committee Manager (Name)	customersTableAdapter
Show output from: Debug ↓ <th>Edit Queries in DataSet De</th> <th>Private signer; Add Query; Preview</th>	Edit Queries in DataSet De	Private signer; Add Query; Preview
'DatabaseExample.exe' (CLR v4.0.30319: DatabaseExample.exe): Loaded 'C:\WIND 'DatabaseExample.exe' (CLR v4.0.30319: DatabaseExample.exe): Loaded 'C:\WIND 'DatabaseExample.exe' (CLR v4.0.30319: DatabaseExample.exe): Loaded 'C:\WIND 'DatabaseExample.exe' (CLR v4.0.30319: DatabaseExample.exe): Loaded 'C:\WIND The program '[23400] DatabaseExample.exe' has exited with code 0 (0x0).	(Name)	
<	indicates the name used i	in code to identify the object.

Figure 34: Adapter and Add Query

Ó

3. In the Search Criteria Builder give your new query the name "FillByCity" and add the following query text:

```
SELECT CustomerID, CompanyName, ContactName, ContactTitle,
Address, City, Region, PostalCode, Country, Phone, Fax
FROM dbo.Customers
WHERE City = @City
```

Then click the OK button. You should notice the fillByCityToolStrip appear on the tool bar.



Figure 35: Fill by City Tool Strip Widget

Try running the program again. How does the tool strip modify the data in the window? Can you filter by different cities? How do you refresh the data?

Today's Exercise

Now that you have some experience with adding data to a form we have supplied a SQL script (ParkwayCommute.sql) with generated data from Parkway Commute. Modify your application that you started on last week to add the data and view it. If you are having any issues with adding information then ask a staff member for assistance. They will be able to help you construct the necessary SQL queries in order to get the correct data displayed.

Once the data has been added and can be viewed, review your requirements and use cases. Which of these can you implement using this process?

For an advanced exercise, try creating a database to store usernames and passwords following the format in the ParkwayCommute.sql script. How would you ensure that this worked?

Summary

Today we have covered some of the basics of backend development, specifically looking at the Model-View-Controller design pattern. By the end of today's session you should have a project which has a user interface and connection to a database. Next session we will start exploring

more complex functionalities that you could add to your application to implement more of the initial project requirements. Be sure to save your application in a safe spot (e.g. Google Drive) so that you can open it again next time.

Useful Resources

- MVC Magic!: <u>https://avelonpang.medium.com/mvc-magic-13e37d782cf7</u>
- Create a Windows Form to search data: <u>https://docs.microsoft.com/en-</u> us/visualstudio/data-tools/create-a-windows-form-to-search-data?view=vs-2022
- SQL queries syntax: <u>https://www.w3schools.com/sql/sql_syntax.asp</u>