



CLDV6211 POE

PORTFOLIO OF EVIDENCE BY KIAAN MAHARAJ

Table of Contents

POE (Publishing Web App)	2
Database and Web Application Published	7
Task 1- Revised.....	8
Lecturer Feedback.....	8
Revised Entity Relationship Diagram (ERD)	9
Feedback on implementation of correction	10
Task 2 – Revised	11
Lecturer Feedback.....	11
Feedback on implementation of correction	12
Self-Evaluation	13
What have I learnt?.....	13
Task 1	13
Task 2	13
Task 3	13
What I could have done better?	13
What would I do differently next time?.....	13
Overall.....	13

POE (Publishing Web App)

The following screenshots, indicate the process to publish a web application.

- i. The developer is first required to open up an IDE (Visual Studio), and then open up the correct application that is required to be published. You should then locate the server explorer of the application on the right-hand side navigation tab. Right-click on the project name “Domingo Roof Works” and select the “Publish” option as shown in Figure 1.

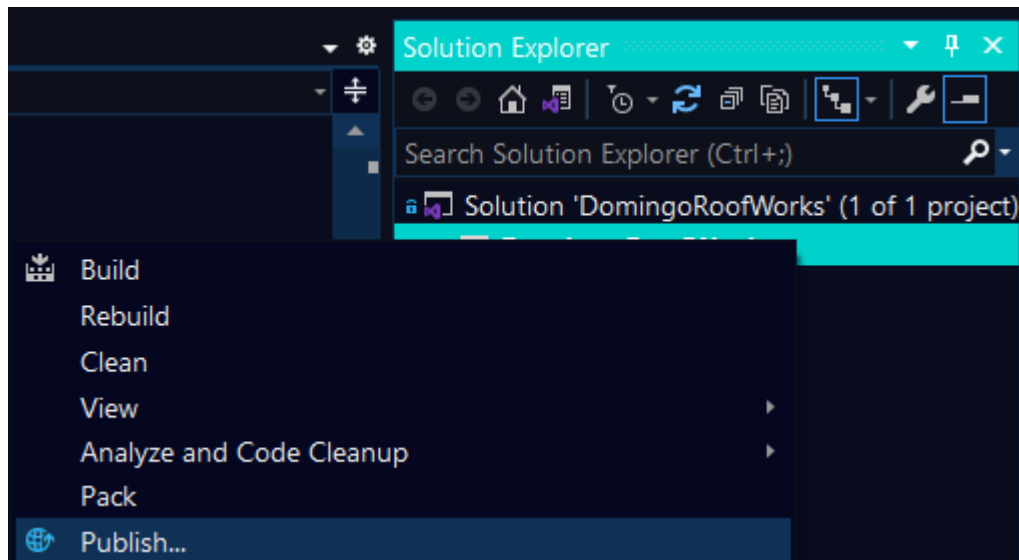


Figure 1

- ii. The next screen prompted to the user is shown below in figure 2, the user is required to select the option “Azure” and click on the button “Next”.

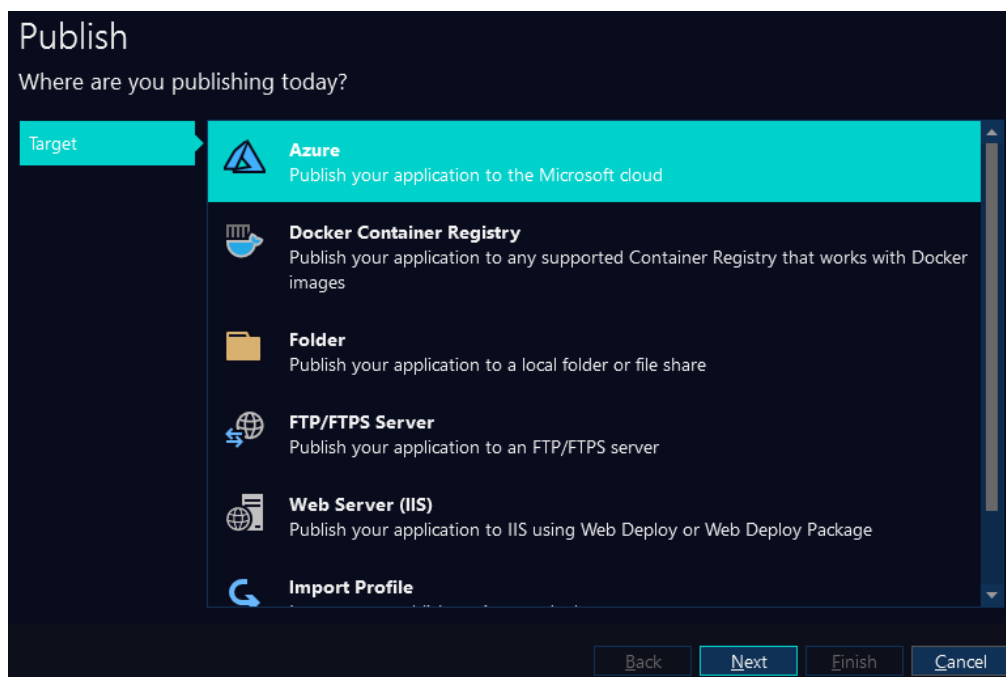


Figure 2

- iii. The user is then prompted with the screen shown below in figure 3. The user is required to select the option “Azure App Service (Windows)” (for this task) and thereafter click on the next button.

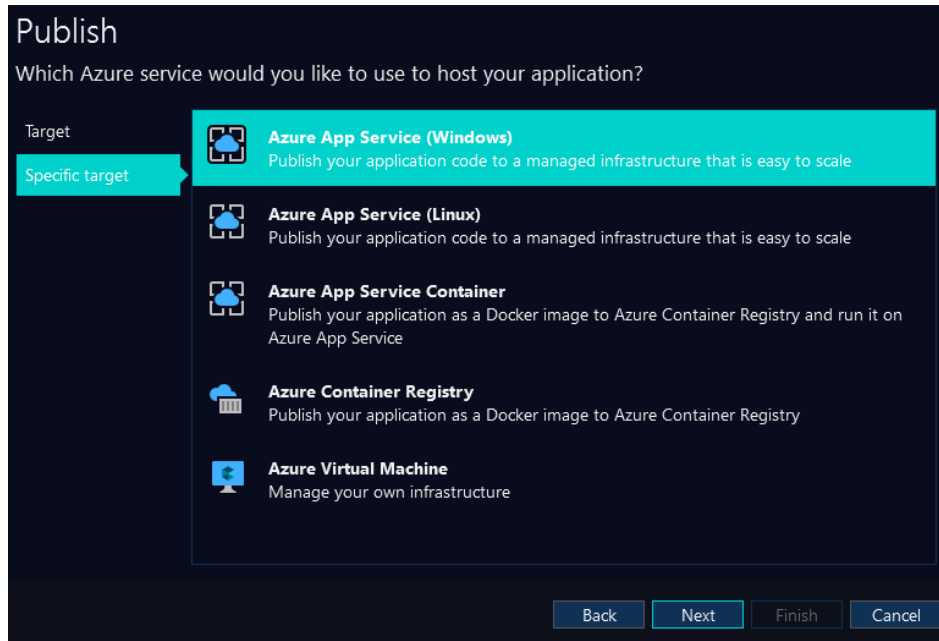


Figure 3

- iv. The user will then be displayed with the App Service(Windows) screen. The user is required to fill in the required details as shown in Figure 4 below. And then required to click on the create button.



Figure 4

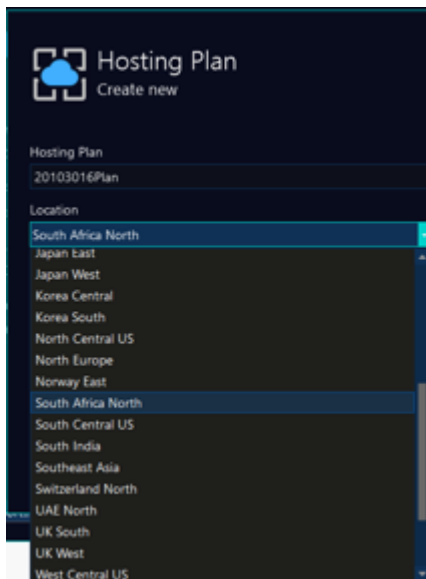


Figure 4.1



Figure 4.2

- v. Once the process has been completed the following screen will show as seen in figure 5. The user will then be required to select the finish option provided below.

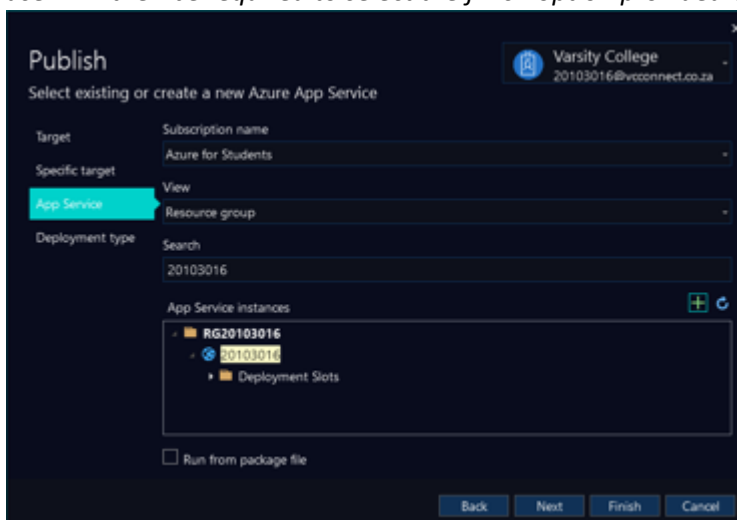


Figure 5

- vi. The user will then be prompted with the screen shown below in figure 7. The user is required to click on the configure button and select the Azure SQL Database to be configured and then to select the database that has been migrated onto azures platform.

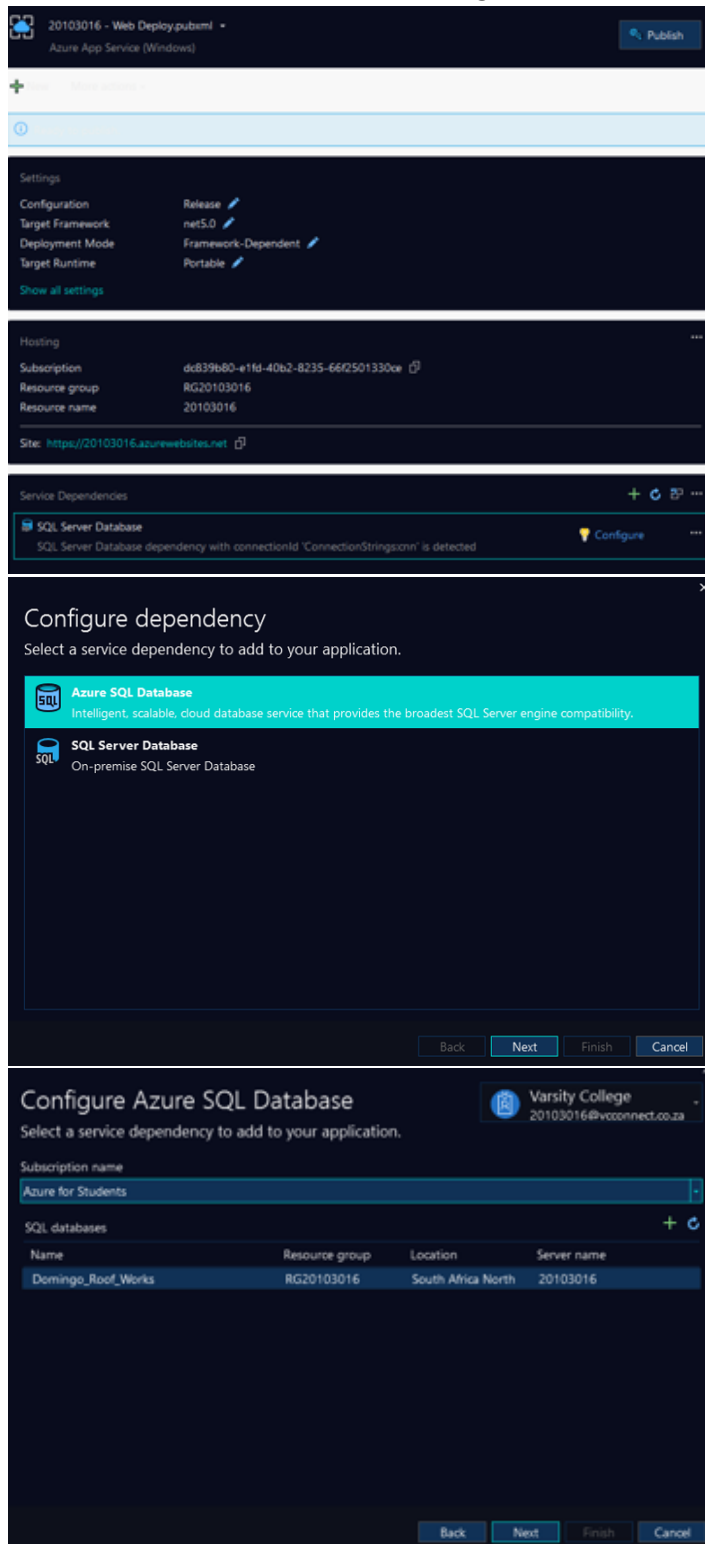


Figure 7.1-7.3

- vii. The user will then be displayed with the configure azure SQL Database, and the user is then required to fill in all the correct details below as shown in Figure 8 and then click on the next button.

Figure 8

- viii. Once all has been configured, the user will be shown the following screen as seen in Figure 9. The user is then required to click on the close button.

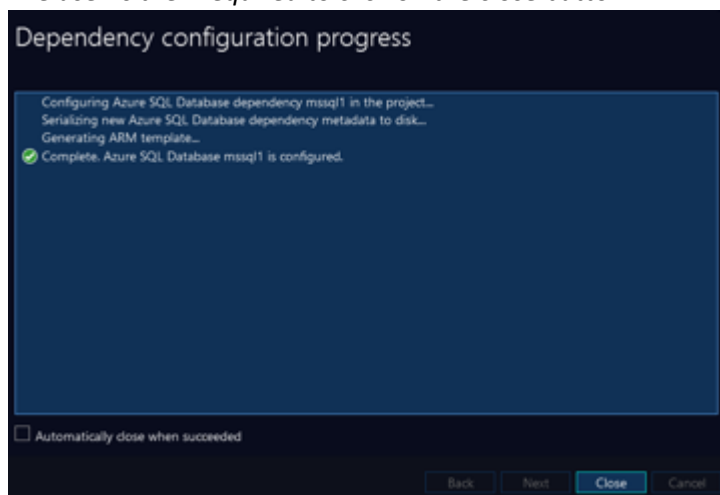


Figure 9





- ix. Lastly the user will be brought back to the screen shown in figure 7.1. The user is then required to click on the Publish button on the top right hand side corner as shown in figure 10.



Figure 10

Database and Web Application Published

As can be seen below is proof that under my resource group, we have my server, database, app service, and app service plan published to the Azure Cloud.

 20103016	SQL server	RG20103016	South Africa North	Azure for Students
 20103016	App Service	RG20103016	South Africa North	Azure for Students
 20103016Plan	App Service plan	RG20103016	South Africa North	Azure for Students
 Domingo_Roof_Works (20103016/Domingo_Roof_Works)	SQL database	RG20103016	South Africa North	Azure for Students

Link to Published application: <https://20103016.azurewebsites.net/>

Task 1- Revised

Lecturer Feedback

After being given the following feedback for my Task 1 submission as shown below:

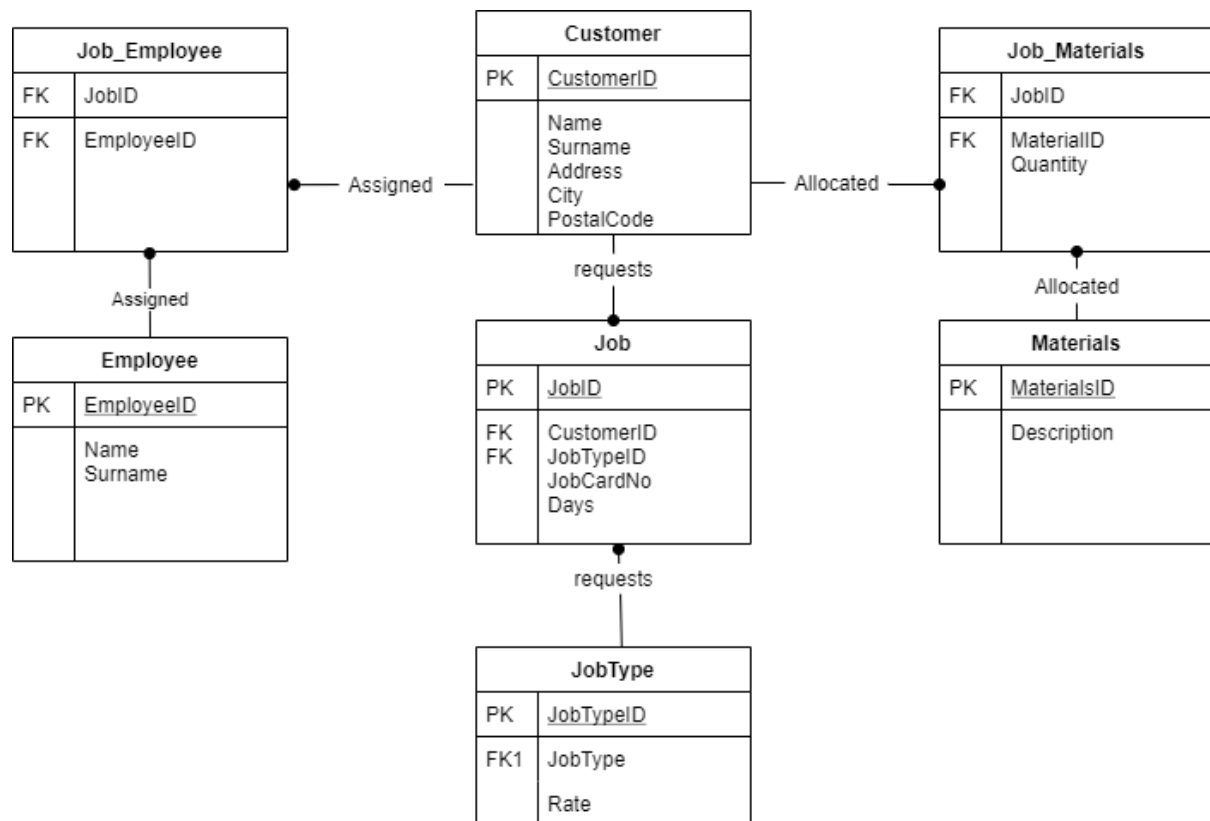
		Notation: Diagram uses an appropriate E-R notation. The notation is used correctly for all elements of the diagram.	Professionalism: Diagram presents a professional appearance. It could be shared with a "real-world" customer without changes.	Entity Sets: Diagram captures all entity sets necessary for a database that would satisfy the initial problem statement.	Attributes and Keys: Diagram captures all attributes and primary keys necessary for a database that would satisfy the initial problem statement.	Complexity: The required number of tables and foreign key relationships will be needed to implement the database.	Constraints: Diagram captures all cardinality and participation constraints necessary for a database that would satisfy the initial problem statement. (Recognising that if all relationships are legitimately many-many with partial participation, then no constraint annotations are necessary.)	TOTAL
20103016	Maharaj, Kiaan	8	6	15	18	18	17	82
		no arrows	neatness				check job-jobtype	

I have done more self-study to understand better the concepts of an Entity-Relationship Diagram. Thanks to the feedback given and the resources that have helped me understand where I have went wrong and how critical it is for the mappings of an ERD to be mapped correctly in order to design a database which would then be required to build in task 2. On the next page, is the revised diagram that I have created, using the feedback provided as my guideline.

Revised Entity Relationship Diagram (ERD)

Assumptions:

- A job can have one or many employees.
- A job can have one or many materials.
- A job can have multiple job types.



Version No. 2.0

Last Updated: 06/06/2021

Feedback on implementation of correction

<i>Feedback</i>	<i>Correction</i>
<i>no arrows</i>	<i>I have removed the arrows and use lines with solid dots to represent the correct cardinality. My diagram uses appropriate E-R notation. The notation is used correctly for all elements of the diagram.</i>
<i>neatness</i>	<i>I have decided to go with the plain and simple approach and design by using Draw.io and was able to design a neat and elegant model that presents a professional appearance. It could be shared with a “real-world” customer without changes.</i>
<i>check job-type</i>	<i>I have noticed my mistake and had fixed the job-type entity. I have also included an attribute called Rate. My diagram captures all cardinality and participation constraints necessary for a database that would satisfy the initial problem statement.</i>
<i>overall</i>	<i>Overall, I am much happier with the new ERD that I have fixed and designed, and I can confidently say that it meets all requirements with regards to Attributes and Keys, Constraints, Complexity Notation, Professionalism and Entity Sets.</i>

Task 2 – Revised

Lecturer Feedback

After being given the following feedback for my Task 1 submission as shown below:

		Use SQL statements to create and change databases, tables and indices. 30	Use the INSERT, UPDATE and DELETE statements to modify the data in a table. 20	Construct SELECT statements to solve data requirements by coding the SELECT clause, the WHERE clause and the ORDER by clause. 30	Demonstrate proper coding format for SQL statements. 10	Database Migration to Windows Azure. 10	TOTAL (Round Up)
20103016	Maharaj, Kiaan	29	20	30	10	10	99
	Masterful!						

Feedback on implementation of correction

I have made some changes to my database in order to have a successful web app that would easily be scaffolded in terms of the models, views and controllers. The changes I have made are as follows:

I. Job Table:

⇒ Added a new primary key called JobID, and gave it an auto increment function which is IDENTITY(1,1). I changed JobCardNo to a normal property.

II. Employee Table:

⇒ Edited the primary key EmployeeID to an auto increment function which is IDENTITY(1,1)

⇒ Added a default employee called N/A that are allocated to all jobcards that don't have an employee in order to display an invoice.

III. Material Table:

⇒ Edited the primary key MaterialID to an auto increment function which is IDENTITY(1,1).

IV. Job-Type Table

⇒ Edited the primary key JobTypeID to an auto increment function which is IDENTITY(1,1)

With all these changes being made, it made my database to work more efficiently with my ASP .Net web application , as when a user creates a new instance of example JobCard, the JobCardID will now auto increment and not ask the user to enter the value in.

Self-Evaluation

What have I learnt?

Task 1

- ⇒ During my Task 1, I have learnt along the way and gained a better understanding of what an Entity Relationship Diagram is and what it is used for.
- ⇒ I have also learnt how critically important it is for the ERD to be designed, as it really is the foundation of the database that has to be build and how that later affects the design and code for the web applications that needs to be designed.
- ⇒ I have also learn that it allows you to you to produce high-quality database design to use in database creation.
- ⇒ I have learnt how to draw the different types of ER Models and the different types of notations that can be used in different business environments.

Task 2

- ⇒ In task 2 I have learnt how to create, delete, and alter databases and tables.
- ⇒ How write complex SQL queries using joins and being able to also reduce the redundancy in a database.
- ⇒ I have learnt database normalization and how critically important it is to normalize a database in order to avoid problems at a later stage. It is also good practices.
- ⇒ I have learnt how to migrate a database to the cloud using azures platform in multiple ways.
- ⇒ I learnt how to run quires on cloud databases as well.

Task 3

- ⇒ During this task, I have expanded my knowledge on the .NET Core and .NET Framework.
- ⇒ I learnt a lot of HTML and JAVASCRIPT.
- ⇒ I enhanced my skills and knowledge as a programmer and learnt a lot about Models, Views and Controllers.
- ⇒ Another cool and exciting feature that I learnt was how to use bootstrap.

What I could have done better?

- ⇒ In all honest the only thing that I would have done better was ensured that I could have created a much better ERD and scored a higher mark.
- ⇒ I could have done a better user interface for my Task3 , but due to my limited skills in MVC. I did the best I could in order to meet all functional requirements.

What would I do differently next time?

- ⇒ Next time, I will make sure that my skills have been developed in web design to ensure that I could create a much more powerful application that can implement a lot of features like a search bar. I have attempted to add a register and login page which did not work so it would be my goal to attempt and knock that out of the park and ensure that I can implement those features. I would also try using a different platform and experiment what the world of IT has to offer.

Overall

- ⇒ This has been such a fun and learning experience of a task that I am positive in my ability to now go and build more ERDs, databases and web applications and show pride in my work as I really enjoyed this topic of research.

References

Bootstrap, n.d. *Bootstrap*. [Online]

Available at: <https://getbootstrap.com/>

[Accessed 06 July 2021].

IT, F. o. W., n.d. *How to Create Folder / Image folder in ASP.Net Project*. [Online]

Available at: <https://www.youtube.com/watch?v=bhDGw8P5fW4>

[Accessed 05 July 2021].

Microsoft, n.d. *Part 4, add a model to an ASP.NET Core MVC app*. [Online]

Available at: <https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-mvc-app/adding-model?view=aspnetcore-5.0&tabs=visual-studio>

[Accessed 06 July 2021].

Wallpaperaccess, n.d. [Online]

Available at: <https://wallpaperaccess.com/>

[Accessed 06 July 2021].

Image References used in source code (Wallpaperaccess, n.d.)

Aesthetics added to source code(Bootstrap, n.d.)

Guideline used to scaffold the database , the controller and views (Microsoft, n.d.)