The Tuck Shop Project

Orb education

Aim: To decide on a pathway through the Tuck Shop Project.

You will be creating an ordering system for a tuckshop or small store, either in the real world or online. There are lots of resources in this pack that will help you succeed with the project. The choices that affect your path through include:

- The amount of research and planning carried out before you start building;
- The platform used to create the interface (e.g. WordPress, Access, PowerPoint or programming from scratch);
- Whether you are just learning skills or would actually like to use your store in the real world.

Your teacher might have chosen a path for you and checked some of the option boxes below.

Essential | **Informational** | **Interface Options** | **Other Options**

1. Introduction

Task 1 – Keeping a Project Record
Task 2 – First Thoughts
Task 4 – Your Initial Ideas

☐ Task 3 – Collaboration

☐ Task 5 – An Introduction to Systems, Design and Computational Thinking

2. Project Management

☐ Task 1 – Collaborating Online

☐ Task 2 – Project Management
Task 3 – Project Management Tools
Task 4 – Project Management Websites

☐ Task 5 – Trello: A Simple Solution

3. Understanding the Issues

Task 1 – Design Thinking
Task 2 – Empathising

☐ Task 3 – Systems Thinking

☐ Task 4 – Collecting Data

4. Defining the Project Goals

Task 1 – Writing Problem Statements
Task 2 – Ideation
Task 4 – Selecting Ideas

☐ Task 3 – Organising Your Thoughts

Checklist (page 2)

5. Non-Functional Requirements

- Task 1 – Functional and Non-Functional Requirements
- Task 2 – User Friendliness
- ☐ Task 3 – Portability
- ☐ Task 4 – Scope and Boundaries

6. Organisation and Timing

- Task 1 – The Path Forward
- Task 3 – Decomposition and Abstraction
- Task 4 – Creating a Timeline
- ☐ Task 2 – Computational Thinking
- ☐ Task 5 – Creating your Timeline in Trello

7. Interface Solutions

- **Option 1:** Website Builder
- **Option 2:** PowerPoint
- **Option 3:** WordPress
- **Option 4:** Access Database
- **Option 5:** Programmed Website

Appendix 1A:
A. Investigate
- A1 – Action Buttons
- A2 – Slide Masters
- A3 – Reusing Objects
- A4 – Design Themes
- A5 – Smart Art
- A6 – Image Links
- A7 – Button Layering

Appendix 2A:
Investigate
- A1 – Installing WordPress
- A2 – Users
- A3 – Getting Back In
- A4 – Posts and Pages
- A5 – Themes
- A6 – Customising a Theme
- A7 – Editing Pages
- A8 – WooCommerce
- A9 – Setting Up Shop
- A10 – Editing a Product
- A11 – Orders

Appendix 3A:
Investigate
- A1 – Design Themes
- A2 – Displaying the Menu
- A3 – Lookup Fields
- A4 – Calculated Fields
- A5 – Combining Data from Multiple Tables
- A6 – Command Buttons
- A7 – Queries and SQL
- A8 – Adding Items to an Order
- A9 – Deciding when to Add Tables

Appendix 4A:
Investigate
- A1 – Creating a Domain
- A2 – Testing your Website
- A3 – Creating the Database
- A4 – Setting Up a Table
- A5 – SQL Queries
- A6 – PHP
- A7 – Connecting to the Database
- A8 – Retrieving Data
- A9 – HTML in a PHP Page
- A10 – CSS
- A11 – DIVS
- A12 – Forms
- A13 – JavaScript
- A14 – Form Handling
- A15 – Menu Pages

Checklist (page 3)

8. Prototypes and Early Testing

- Task 1 – Functional Design in a User Interface
- Task 2 – Wireframes
- Task 3 – Visual Design
- Task 4 – Creating Designs

- ☐ Task 5 – Accessibility and Usability
- ☐ Task 6 – Further Options
- ☐ Task 7 – Testing your Prototypes

9. Pattern Recognition

- Complete for your chosen interface

10. Flowcharts and Algorithms

- ☐ Create Flowcharts
- ☐ Write Algorithms

11. Building the System

- Task 1 – Checklist
- Task 2 – Building the Interface
- Task 3 – Debugging Code

Option 1: Website Builder

Option 2: PowerPoint

Option 3: WordPress

Option 4: Access Database

Option 5: Programmed Website

Appendix 1B: PP Programming
- B1 – Macros
- B2 – Button Layering
- B3 – Programming a Menu
- B4 – Programming a Better Menu

Appendix 2B: WordPress Skills
- B1 – Images
- B2 – Store Categories
- B3 – Attributes
- B4 – Page Setup
- B5 – Widgets
- B6 – Menus

Appendix 3B: Database Design
- B1 – The Purpose of the Database
- B2 – Primary Keys
- B3 – Identifying Relationships
- B4 – The Data Dictionary
- B5 – Creating the Tables
- B6 – Setting up Relationships

Checklist (page 4)

11. Building the System (Cont.)

Appendix 1C: Going Further
- C1 – Options
- C2 – Browsed at a Kiosk
- C3 – The Developer Tools

Appendix 2C: Going Further
- C1 – Options
- C2 – Using Your Store
- C3 – HTML
- C4 – CSS
- C5 - JavaScript

Appendix 3C: Further Help
- C1 – Structured Menu
- C2 – Adding Calculated Fields to a Table
- C3 – Orders and Order Details Using a Subform
- C4 – Orders and Order Details Using Programming
- C5 – Going Forward from Here

12. Test and Refine

- Task 1 – Personal Testing
- Task 3 – Refining
- Task 4 – Taking Things Further

☐ Task 2 – Wider Testing

Your challenge is to design and build a digital ordering system for a small store. It might be a tuck shop, canteen, uniform shop, stationary store, equipment supplier or any other small business that takes orders for products or services. It might be for your school, club or society. It could be a shop that is already in existence or it might be a new venture. It can be real or imagined. It could be online.

Aim: *To start thinking about what the project involves and what is of interest.*

The ordering system should have the following properties:

- *It should allow customers to browse through and select products from a menu;*
- *Orders might be taken through devices at the counter or over the internet;*
- *When the goods or services are provided to the customer, they should be checked off as completed;*
- *When designing the system, you should take account of everyone who uses it. This might include the customers, staff, administrators and other developers.*

In these resources, we will consider the development of a school tuck shop. The shop will take orders from its menu and have the items bagged and ready in time for lunch. You may work with a similar type store or adapt the project to something else that takes your interest. You don't necessarily need to build the whole system; your project might be to develop only part of it - it's all good experience.

Task 1 – Keeping a Project Record

You should keep a record of all your research and activities. This could include any of the following:

- *Notes about the tasks you carry out and the things you learn completing them;*
- *All your planning details, including responses to the questions in this guide;*
- *Information about any apps, websites or programming solutions you discover along the way;*
- *Pictures or drawings of anything that helps describe your ideas.*

We suggest you start by adding some notes to a Word document for now. If you decide to use a project management tool later, you can copy your notes across to this.

Task 2 – First Thoughts

a. Consider your school's tuck shop, uniform shop or other similar business that you have used, either recently or in the past (perhaps even back in primary school). How do they deal with orders? Briefly outline the steps in the process.

b. Make a list of all the people you think are involved in the shop, either on the business side or as a customer.

c. What do you think of the way the shop takes orders? Does it seem to work well?

d. If there are obviously problems, what changes do you think would improve the system?

You should by now have completed your research and written down the goals you aim to achieve in the project. It's time to start looking at modelling the solution (the prototyping stage). We will use some Computational Thinking during this phase of development, breaking down the problem into smaller parts. We can then list all the tasks that need completing and decide who is going to do what.

Aim: *To plan how the project will be carried out.*

Task 1 – The Path Forward

You may by now have a good idea of the steps required of this project. If you need further help with these, then place the tasks below in the order that they might be completed. They won't necessarily all apply to your own project but they will give you a good idea about how things could be planned. It's not an absolute science; there is some flexibility.

a. Create wireframes and test prototypes of the system.

b. Build your user interface based on all the ideas and feedback you have received.

c. Decide on areas of responsibility for the members of your group so that the work is shared.

d. Look at templates and other 'out of the box' solutions that could save you time and improve professionality.

e. Evaluate your finished product and look at ways of implementing it.

f. Design the style of the user interface, selecting colours, background images, buttons etc.

g. Decompose the problem fully and make a step-by-step plan.

h. Consider users with limited physical abilities.

i. Test all possible paths through your user interface and fix anything that doesn't work.

j. Share your ideas with potential users and get some feedback about how you might proceed.

k. Investigate software and apps that might provide a platform for your ordering system.

l. Create flowcharts and algorithms for your interface or any programming required to run it.

Suggested order _____

Task 2 – Computational Thinking

When using computational thinking, we break down the problem, look for ways to make the job easier, then build sets of instructions for completing the tasks. If you need a recap, try and match each term below to its definition.

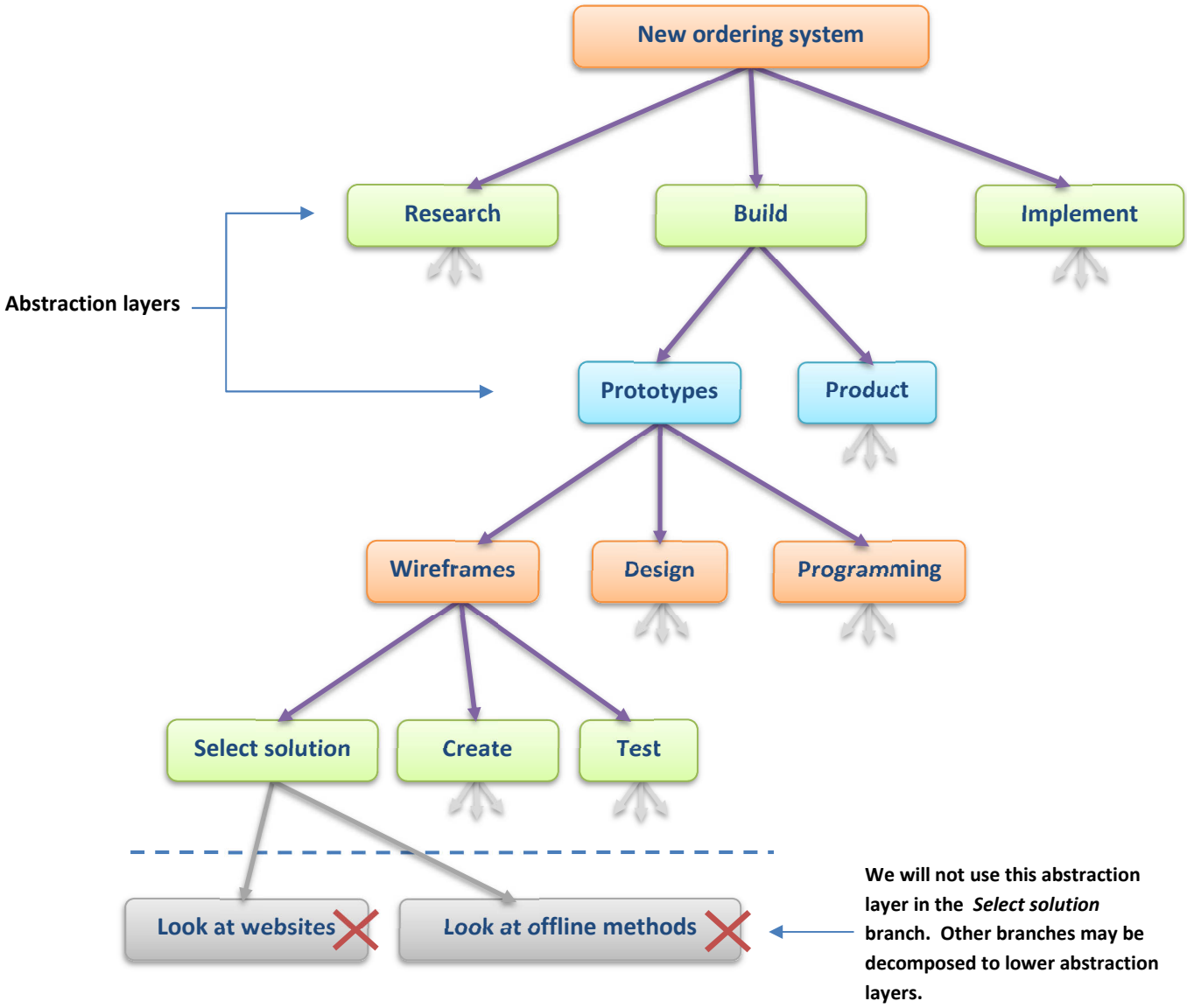
	Term				*Definition*
1	**Decomposition**	•	•	a	Identifying sections of the problem that are similar to each other or have been tackled before.
2	**Pattern recognition**	•	•	b	Providing step-by-step solutions to different parts of the problem.
3	**Abstraction**	•	•	c	Breaking down a difficult problem into more manageable pieces.
4	**Algorithms**	•	•	d	Hiding the complicated bits that we don't need to consider.

Organisation and Timing (page 2)

Task 3 – Decomposition and Abstraction

Decomposition means breaking down a difficult problem into manageable pieces. A common way to tackle this process is to use a tree diagram showing increasing levels of detail. The diagram below is just an example of how you might decompose this project. You may build on our diagram or start your own from scratch.

Abstraction is about deciding on the level of detail required to describe the tasks people will carry out. In the example below, we have ignored the abstraction layer below *Select solution* because we felt this was unnecessary detail. You should make decisions about the where to finish decomposing each branch of the tree.



More about Abstraction

In our tree, we have decided to stop decomposing the *Wireframe* branch at *Select solution*. But deciding on the solution you use to create your wireframes is a definite step; it might involve investigating websites that enable you to share wireframes as well as looking at other methods of creating them. However, for our project, we have decided that no more detail is required in our plan. It will be one person's job to research wireframes; it won't be split up and shared between different people. Therefore, in our case, no more decomposition is required. You can make your own decisions as the abstraction layers you decompose to.

Building the interface is obviously a huge part of this project. How you achieve this depends on the experience of your team and the skills you would like to practice and develop. Before going any further, you should make a decision about the tools you will be working with. We have provided some suggestions below, although you are free to choose any solution that works.

Aim: *To decide which platform to use for development then learn how to use it.*

Solution	Difficulty	Suggested functionality	Support offered here	More Information
Website Builder	Easy (completing a template)	Fully featured online web store.	Very little support. They are easy to use.	Web builders allow you to create a fully functioning, professional store in a flash. Having said this, there is very little design or technology involved; you would simply be substituting text and images in a template. It will be good ecommerce experience but offer little in the way of coding, design or thinking skills. This option is really only for those coming late to the party.
PowerPoint	Easy/Fair/ Difficult, depending how far you take things.	Navigate store menu. Select and add items to an order (with some coding).	Basic PP skills required. Instructions for adding functionality offered.	PowerPoint may not be the best platform for an ordering system, but it is an easy way to get started and still have the option of a fully personalised design. Plus, if you would like to try a little VBA programming, you can insert buttons which add products to a list and perform other ordering functions. There will be no data security and no storage or forwarding of orders, but there's the possibility of very good design and technical experience.
WordPress	Easy/Fair/ Difficult, depending how far you take things.	Fully featured web store. Access to coding for tailored functionality.	Enough support provided to build a working system.	WordPress is a platform used to build and run millions of websites around the world. It has lots of menus and *drag and drop* facilities, plus thousands of plugins for added functionality. However, it can still offer a considerable technical challenge, offering full access to the HTML, JavaScript and PHP coding for the addition of tailored features.
Access Database	Fair/Difficult, depending how far you take things.	Interactive menu; select items; add up costs; store and retrieve details.	Lots of support given. Basic Access skills required.	Access can be used to build a solution with menu items displayed, selected and totalled. Customer, product and sales information may be edited using forms. Orders can be displayed in a report. This is an excellent choice if you would like to build your database and VBA/SQL skills but perhaps not implement your solution in the real world (a website provides the most obvious solution if you are looking to put your work to use).
Programmed Website	Difficult	Anything is possible if you can program and utilise free tools.	We provide the support required to set up your website and get started with your code.	If you are super technical minded and have considerable programming skills, then why not start from scratch and code a website yourself? It can be more difficult to produce a polished looking interface but the programming experience will be fantastic.

Aim: *To investigate how to create a fully functional store using WordPress.*

WordPress started life as a blogging platform but has grown into the content management system of choice for many millions of websites and stores. In its barest form, WordPress helps you create websites using a series of menu pages and tools. However, this functionality can be increased almost indefinitely with the addition of themes and plugins developed by people all over the world. Some of these addons are completely free, some cost from the outset and others charge only for advanced features. You can go a long way without spending any money.

Technically, you can take WordPress to almost any level. It's quite possible to build a shopping interface without looking at a single line of code. However, if you wish your website to work in a very specific way, WordPress offers you access to the programs running in the background. Your use of this code might range from minor edits in the HTML to the addition of JavaScript functions and other applications. The choice is yours.

Things that can be achieved in WordPress:

- *Build a web store without coding;*
- *Add functionality, selecting from a massive range of themes and plugins;*
- *Use HTML, CSS, PHP and JavaScript programming to tailor features.*

Things that are a little difficult in WordPress:

- *There is always a dizzying array of options. WordPress can be a challenge to get your head around;*
- *The WordPress database that stores your information is complex, so it won't be edited directly. The store setup, processing of orders and the analysis of sales are all conducted through webpages. There will be a lot of menus.*

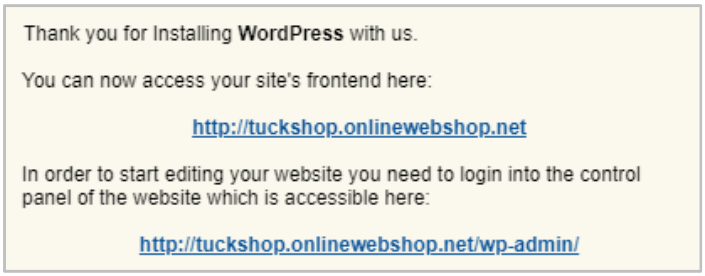
WordPress is a great option for anyone who is interested in online marketing experience, plus there are plenty of technical possibilities for those who want to take things further. It can be a little frustrating at times. If something doesn't work as it should, it can take quite a while to find the problem; there are so many possible causes.

This help guide will start by looking at the setup of a basic storefront. You are advised to work through the introductory tasks first with a small amount of data. Once you have a good idea about how your interface might work, return to the planning activities and decide how to proceed. This guide then offers suggestions and support for additional functionality.

Task A1 – Investigation: Installing WordPress

You will need a webhost to host your WordPress site. A webhost stores your data and serves up webpages to your website visitors. When you install WordPress, your webhost will make a copy of all the original WordPress files for you to work with. It will also secure these files and give you permission to edit, delete and add to them. The rest is basically up to you.

Follow the instructions in the introductory section (Interface Solutions) to get up and running. This involves setting up a free hosting account with https://www.awardspace.com/ and using the *Zacky Installer* to install *WordPress*. You should end up with an AwardSpace account, the addresses of your website and the WordPress *Control Panel*, plus the username and password required to access it. Make a note of all this information in your log.


Thank you for Installing **WordPress** with us.

You can now access your site's frontend here:

http://tuckshop.onlinewebshop.net

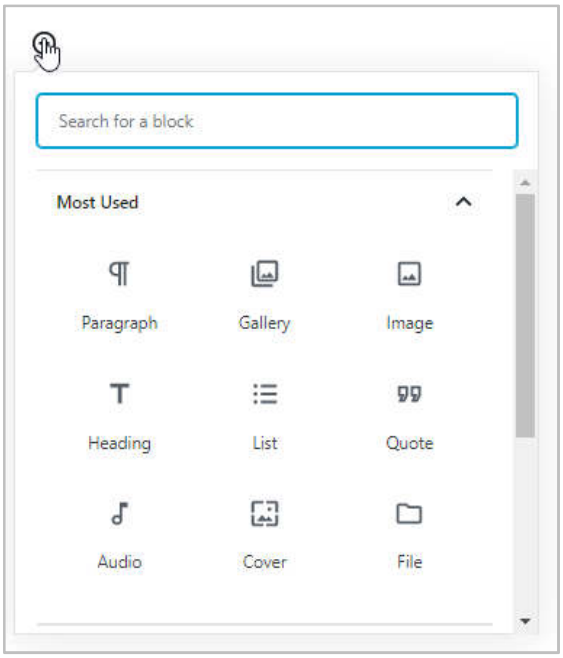
In order to start editing your website you need to login into the control panel of the website which is accessible here:

http://tuckshop.onlinewebshop.net/wp-admin/

Task A7 – Investigation: Editing Pages (Cont.)

Investigate the options for adding a new *block* below the first. Depending on your setup, the options available may include:

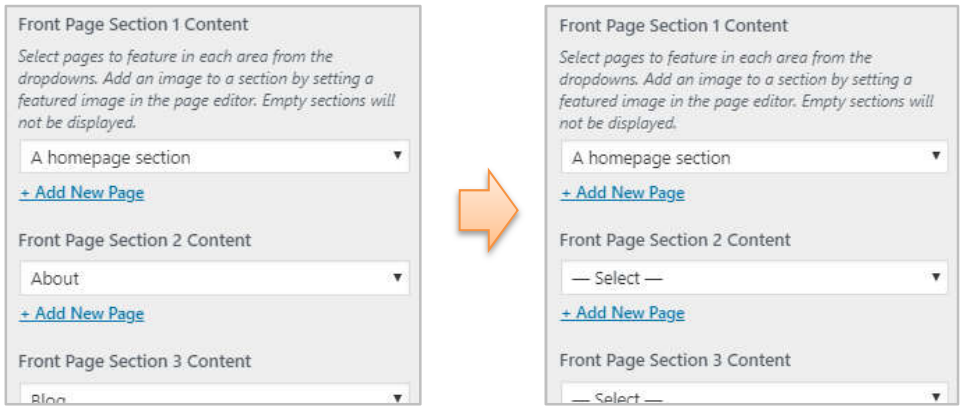
- **Layout objects** *e.g. paragraphs, lists, columns and tables.*
- **Multimedia objects** *e.g. images, audio and video.*
- **Widgets** *e.g. calendars and search boxes.*
- **Embedded objects** *e.g. YouTube, Facebook and SoundCloud.*
- **Coding** *e.g. shortcodes and custom HTML blocks.*

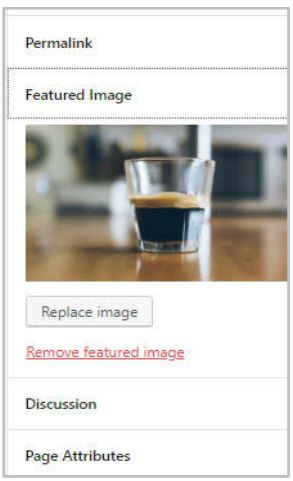


It's worth remembering that the options available for editing can vary tremendously. They depend on your WordPress version, the theme you have selected, the plugins you have installed and your choice of editor. Once under way, we can really only offer general guidance and point you in the direction of tools that may be of use.

Open the Homepage in *Edit* mode (in the Twenty Seventeen theme, it's called 'Home – Front Page'). Again, there might appear to be little in the way of content when compared to the one viewed in your live website. One reason for this is that the homepage includes the common content such as titles and menus. The other is that there are specific homepage settings in the theme customiser. In the case of the Twenty Seventeen theme, the front page (i.e. the homepage) can draw in content from other pages, such as the *About* and *Contact* pages. This information is duplicated, appearing both on the homepage and in their own pages.

You may of course decide whether this is how your website works. In this case, the repeated content can be switched on and off in the '**Appearance / Customize / Theme Options**' section. Simply set any sections to '–Select—'if you don't want them to appear.





Other changes for individual pages can be made in the panel on the right when a page is open for editing. Again, the options available here differ depending on the theme you are using. Have a look at one of your pages in *Edit* mode and see what's available in this group of settings.

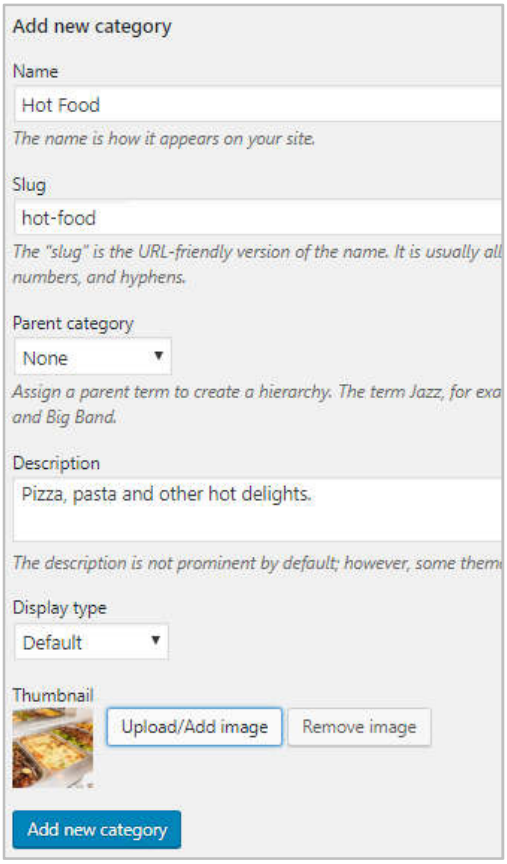
Hopefully, you will now have a little understanding of the convoluted set of menus and options WordPress uses to create websites. It's time to shake things up with a look at the ecommerce tools required for your store.

Task B2 – WordPress Skills: Store Categories

You might have decided to include some structure in your menu, such as Hot Food, Cold Food and Drinks. These sections can be set up easily using WooCommerce categories.

a. Select '**Products / Categories**' in the main menu.

b. Use the form on the left of the screen to complete details for a new category. You can ignore the *Parent* category – this can be used later if you want to add more levels to your categories.

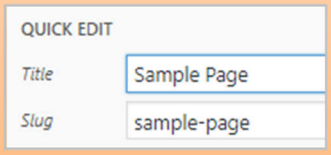


Slugs

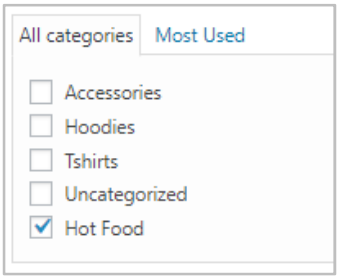
In WordPress, a slug forms part of a user-friendly URL. For example, the slug 'hot-food' in a category will result in the URL below:

tuckshop.onlinewebshop.net/?product_cat=hot-food

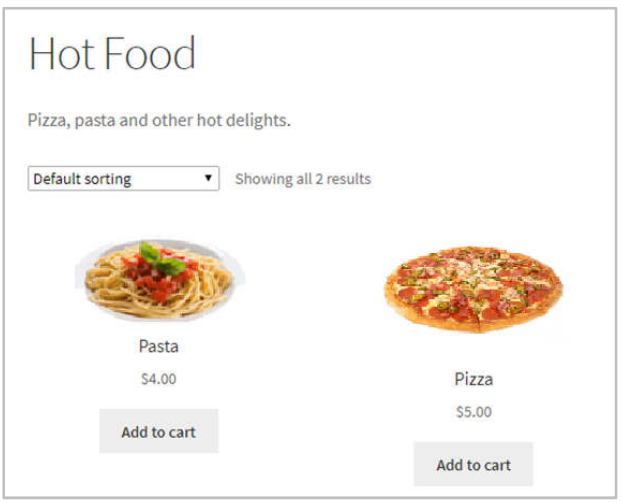
Slugs can be set for categories, pages, product, posts and other objects. They give context to the URL and help with search engine optimisation (SEO). Try the *Quick Edit* mode for the easy editing of slugs.



c. You may assign a product to a category using either the *Edit* or *Quick Edit* modes. The available *Product Categories* will be listed on the right.



d. The category can be viewed in your live store by clicking on the *View* link alongside the category name. Note that the page is viewed in the same window, so remember to navigate back to the control panel rather than closing the browser.



e. Delete the (unused) sample categories and products from your store as they are no longer required. You may check several products and select *Move to Trash* from the *Bulk Actions* drop-down menu, then *Apply*.

Access has long been used for ordering systems, stock keeping and the general processing of business information. It's possible to get almost all the functionality of an ordering system in a single, standalone database, but there are some limitations.

Aim: *To investigate ways of creating a functional ordering system using MS Access.*

Things that can be achieved in Access:

- *Product, customer and order information stored for later use;*
- *Financials, such as record keeping, stock taking, day's sales, best sellers and other reports;*
- *Advanced features such as calculations and programming to add functionality to forms and reports.*

Things that are a little difficult in Access:

- *Sophisticated design effects - the interfaces tend to be clean and simple (and often clunky);*
- *Online collaboration (whilst building, you'll probably need to pass the database file between you);*
- *Remote placement of orders (it would be better to use a database linked to a website).*

In reality, Access is often used for the business side of things rather than being the interface with which customers interact. It can, however, be used with websites or other applications that provide the possibility of better-looking interfaces.

The first sections of this help guide help you get to grips with MS Access as a platform for development. You are advised to work through all the investigatory tasks first then return to the planning activities. When you are ready to build your ordering system, use the later sections of the guide to design a professional database solution.

Task A1 – Investigation: Design Themes

We will create a simple database to investigate tools that might be of use in your interface. You will then be able to return to your planning and create prototypes to test.

Menu

ID	MenuItem	Price		Description
1	Pizza	5	(1)	Ham and pineappl
2	Pasta	4	(1)	Spaghetti with tom
3	Salad	3	(1)	Tomatoes, cucumb

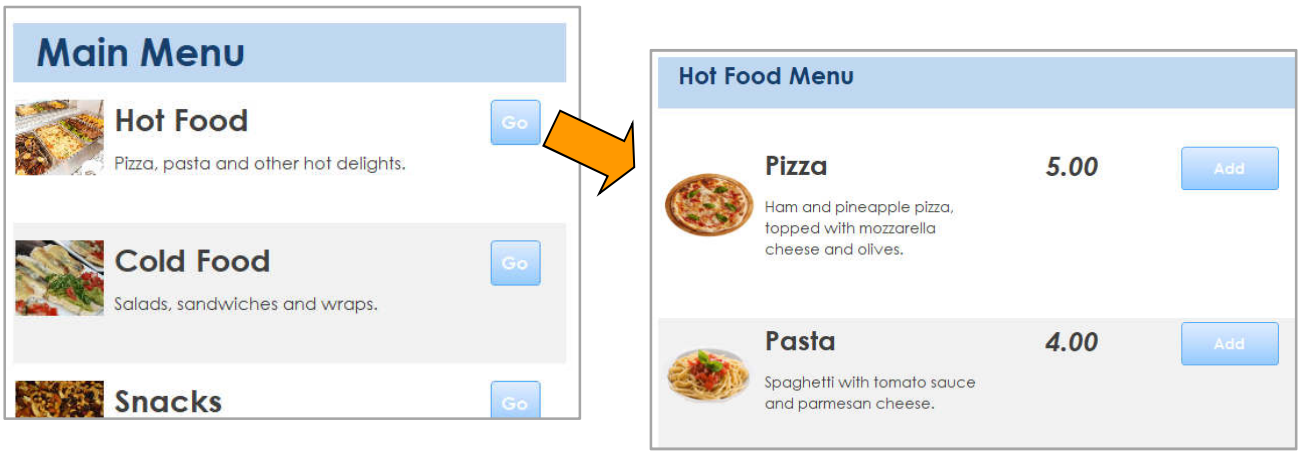
a. Set up a new Access database with the table shown above, saved as *Menu*. The *attachments* field should contain images of the menu items. Double-click on the field to add an attachment.

b. Save the table and create a form ('**Create / Form**'). Add 3 or 4 items to your menu (no more for now).

c. Open the form and select '**View / Layout View**'. Play around with the themes and other options in the *Design* tab. Are the theme styles applied to the table as well? You should also investigate the background images and formatting options.

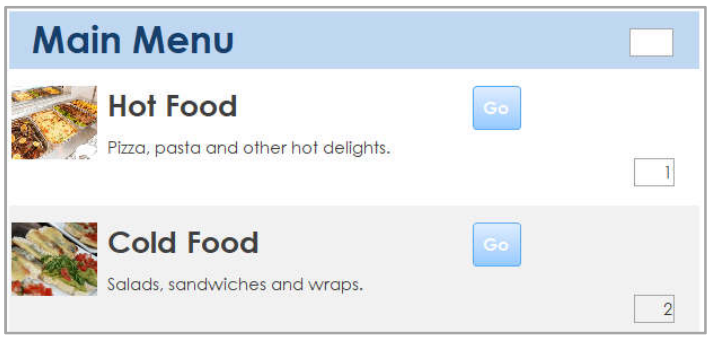
Task C1 – Further Help: Structured Menu

You might have decided to include a structured menu in your system, so that clicking on (say) Hot Food in the main menu displays only a list of hot food items. The best solution for this would store the menu data in two tables; one for the menu sections (or categories) and one for the individual items. We will now look at ways of setting up an interactive menu.



There are many ways to achieve this goal. We will walk you through a very visual method which can be tested frequently as you go along. You may choose a different method if you know one.

a. Create a report for your menu sections - something like the one on the right. It should include:



- a *Go* button to take you to the next screen.
- A text box showing the *MenuSectionIDs*. Use the *Property Sheet* to name this box *txtMenuSectionID*. This box can be hidden later.
- Another text box in the report header to display the ID of the selected section. Use the *Property Sheet* to name this box *txtSelectedSection*. It can also be hidden later.

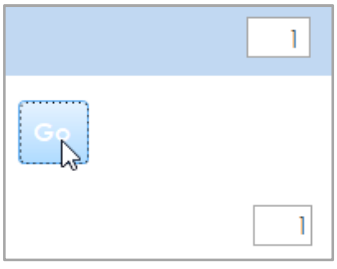
b. Whichever box we click, we want the relevant *Section ID* to be copied to the box in the header. Create an onclick event and use the code builder to create the subroutine on the right. Save the code.

```
Private Sub cmdGo_Click()

    txtSelectedSection = txtMenuSectionID

End Sub
```

c. Open the menu in *Report View* and check that whichever button you click, the associated ID is copied to the top text box. The value will be stored here so that it can be accessed from a query. Close the report.



d. We will now use a query to create a list of menu items to display. The query will dig out our selected Section ID from the last step and use it to filter the data.

First select your *Menu* table (the one with all the individual products) in the panel on the left and open the Query Designer ('**Create / Query Design**').

Task C4 – Further Help: Orders and Order Details using Programming

The problem with the form and subform is that it's a solution for employees rather than for customers. You would not, in real life, give customers access to your tables and expect them to set up and complete an order in this way. Customers should be able to click an *Add* button and everything happens for them. This is going to require some programming.

To help us clarify what is required, we should first think through all the steps that must take place and create a flowchart to build from. Think about the situation when a customer first clicks on the *Add* button. And then clicks on another *Add* button, or even the same one again. We need a plan.

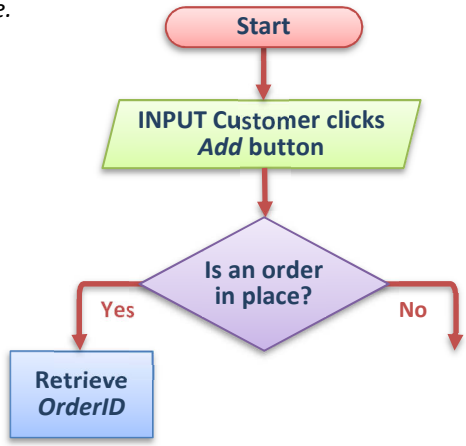
a. Instructions

Place the events below in the order that you think they should happen. A little variation is possible.

1. *If an order is not in place, then one is set up and the OrderID remembered.*
2. *The customer clicks on the Add button.*
3. *If the item is not in place, then it is inserted with a Quantity of 1.*
4. *The database checks to see if an order is in place in the Orders table.*
5. *The order information is displayed in some way.*
6. *If the customer is in the middle of an order, then the OrderID is retrieved and remembered.*
7. *If the item is in place, then 1 is added to the Quantity.*
8. *The database checks to see if the item is in place in the OrderDetails table.*

b. Flowchart

Create a flowchart for the events. We have started one for you.



c. Algorithm

Write an algorithm using structured English. This will become the outline for your programming.

Search for order from this customer

IF order is in place THEN

Retrieve and store OrderID

ELSE

Add a new order to the Orders table

Retrieve and store OrderID

d. Complications

Although your procedure might be coming together, there may be lots of details that you haven't yet considered. For example, when searching the database for an order from this customer, what are you actually going to look for? Here are a couple of the issues:

- The customer might have added products to their shopping cart at an earlier time and then walked away. Do you consider these items?
- There might be orders that have been completed and paid for before. Obviously you would want to start a new order in this situation.

All digital content involves programming at some level. A range of languages are commonly used in websites, including HTML, CSS, JavaScript, SQL and PHP.

Aim: *To set up an online database and connect with it through PHP webpages.*

Things that can be achieved with programming:

- *Anything.*

Things that are a little difficult with programming:

- *Everything.*

A programmed website is a great choice for those with programming experience who want to find out how they might use their skills in the real world. Not only will each of the languages mentioned be required, you will also discover more about databases and data security.

This guide offers enough help to get you started. We will show you how to build an online database and connect with it using SQL. The queries will then be set up in a webpage using PHP and the results presented with HTML and CSS.

Task 1 – Investigation: Creating a Domain

Every website has an IP address. This is a long, ugly string of numbers separated by some dots. For example (at the time of writing) you could visit the Google search page by entering ***172.217.167.78*** into the address bar of a browser. A domain is a user-friendly name that means you can type ***google.com*** instead. Domain name servers working behind the scenes on the internet look up the IP address of the website and forward your request to it.

To set up a top-level domain like ***mywebsite.com*** requires some money, but you may create a less friendly subdomain such as ***mywebsite.onlineshop.net*** free of charge. You will first need a *webhost*.

A *webhost* will store your files and display your webpages to your website visitors. We have suggested using AwardSpace as a host, although there are many others providing small scale services free of charge.

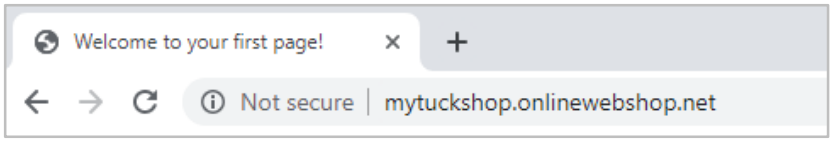
a. Navigate to https://www.awardspace.com/ and set up a free hosting account. You will need to confirm your email.

> ***Note:*** *AwardSpace only allows one free database per account. If you have tested WordPress previously you will need to delete that database first or use a different AwardSpace account.*

b. Once in the control panel, select the *Domain Manager* from the *Website Manager* section.

c. Click *Create a Free Subdomain* and decide on a URL for your website. Click *Create*.

d. Copy or type your new website address into your browser. You should see the default AwardSpace placeholder webpage.



Task 9 – Investigation: HTML in a PHP Page

In the previous examples, we were working mainly with PHP and outputted unformatted text to the screen using the echo function. The next step is to use PHP to control large chunks of HTML in a page. For this, you might have different sections of HTML that are used at different times, or you might have sections of HTML that are used repeatedly.

a. Controlling HTML with IF statements

Page headers

HTML - top section

PHP - If this condition is met…

HTML - Include this content

PHP - Else

HTML - Include this content

PHP - End If

HTML - bottom section

The program on the right uses the *rand* function to generate either the number 1 or 2 randomly. If a 1 is generated, then some text is displayed on the screen. If a 2 is generated, some different text is displayed.

A simplified view of what is happening is shown below. It's the same program without the HTML. It's often easier to set up and test a program in this way first, then add the HTML. It can be less confusing.

```
<?php
$randNo = rand(1,2);
if ($randNo == 1) {
 //Display some text
} else {
 //Display some other text
}
?>
```

```
<!DOCTYPE html>
<html>
<head>
<title>Test Page</title>
</head>
<body>

<h1>Random 1s and 2s</h1>
<p>Controlling HTML using PHP</p>

<?php
//Generate a random number of either 1 and 2
$randNo = rand(1,2);


 if ($randNo == 1) {
//Stop using PHP and return to HTML
?>

<p>The random number generated was 1</p>
<p>BTW - This is HTML</p>

<?php
//Back to the PHP if statement
} else {
//Stop using PHP and return to HTML
?>

<p>This time it was 2</p>

<?php
//Now close the if statement
}
//Stop using PHP and return to HTML
?>

</body>
</html>
```

Create the program and save as **test4.php**. Once the page is open in your browser, refresh it repeatedly using the *F5* key. You should get a random mixture of the two different pieces of text displayed.

As a step towards the next task, place the text in a table. The table should be set up before the *if statement* and closed using a </table> tag after the programming. <tr> and <td> tags can be used to create rows and data cells. We have added a border using HTML, which is not actually good practice (but nor is using tables in this way either; we are just practicing). Save as **test5.php**

```
<table border="1" id="TestTable">
<?php
//Generate a random number of either 1 and 2
$randNo = rand(1,2);


 if ($randNo == 1) {
//Stop using PHP and return to HTML
?>
 <tr>
  <td>The random number generated was 1</td>
  <td>BTW - This is HTML</td>
 <tr>
<?php
```