



Highway 125 Case Study

Inception

Highway 125 Approached BondiGeek in late September 2010 with a development enquiry that was rather intriguing.

- *standard info pages - text images and video*
- *music library - digital tunes for sale by digital download, and some physical CDs for purchase.*
- *shop - merchandise for purchase*
- *live shows - information and ticket booking/purchasing*
- *blog - (eg wordpress) - this serves as our frequently updated news page*
- *A plug-in off-the shelf solution for all the e-commerce - eg Shopify - seems the best idea, but I'd rather work with the web architect on this phase of the decision making.*
- *We are currently in the design/wireframing stage.*

After a series of meetings at local coffee shops in Bondi and at Highway 125 digs at Trackdown Studios it was apparent that this was going to be a fun project with a group of people that were both passionate and committed to what would grow into a brilliant and exciting project.

During the discovery phase it soon became apparent that a product like Shopify, although great for a standard eCommerce site, would not meet the complex requirements of Highway 125.

In addition, rather than go with a static beta site initially, Highway 125 made the decision to go straight to the full blown dynamic solution.

And thus was the beginning of the birth of a new child in the form of the Highway 125 web site.

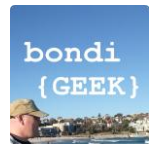
The Early Days

Right from the start, working with Highway 125's Digital Producer, it was clear that this was not going to be a "fly by the seat of your pants" job but instead a slick and professional operation.

Over the course of the next couple of months Highway 125 refined their design document that evolved in to a 55 page brief complete with screen layouts and functional requirements, Colour Specifications, Font Specifications and Site Map. Everything a web developer needs to build and quote on a professional system.

And so it was that agreement was reached and a quote supplied to start work on December 17th 2010.

Of course being the Christmas period, and with travel plans already in place until mid-January, full on development didn't start straight away but an agreement had been reached and the path forward was clear and unambiguous.



The Foundations

The development began with basic database design work being undertaken, based on the design brief supplied by Highway 125, in SQL Server 2008 R2.

BondiGeek decided early in the process that EntityFramework 4.0 would be the foundation of the Highway 125 backend. The reasons for this were clear:

- A flexible and type safe database modelling framework.
- Database changes could be implemented quickly and safely and reflected in the Entity Model.
- Seamless integration with WCF Data Services.

It was also decided early on in the build that WCF Data Services would be the means of communication between the browser and the backend and again the reasons for this were clear:

- Seamless integration with EntityFramework 4.0
- Powerful and Flexible querying capabilities from the client via the OData protocol.
- Powerful and Flexible querying and access from the server

By utilising the above technologies it was possible to get up and running very quickly with a live database solution that could feed real test data in to the web site.

By doing the initial phase of the development in this manner BondiGeek could focus purely on the front end design allowing the client to see the site evolve very quickly and catch any design omissions early on in the development lifecycle.

The web-site is chock full of dynamic content with loads of friendly urls so the ASP.NET routing engine was used extensively. In addition jQuery templates are responsible for rendering virtually all of the content on the client via json data retrieved from Ajax calls through jQuery.

And so it was that the delivery of the skeleton site to Highway 125 was completed bang on schedule on the 24th of January.

Heavy Lifting Commences

The next phase of the development was when the real work began on the backend. This was the finalising of the database schema, design and build of the Shopping Cart and Customer Administration and the build of the CMS for Highway 125.

Some of the functionality and challenges involved in the Site and CMS were:

- Site: A custom shopping cart solution integrating with:
 - eWay for credit card processing
 - Australia Post for postage calculations
- Site: A shopping cart that handles physical, digital and event purchases simultaneously
- Site: A customer administration area to manage purchases and download digital content



- Site: Friendly URL system
- Site: iTunes like preview of tracks
- Site: Integration of YouTube content
- Site: Extendable Social Network Integration for each Artist.
- Site: Internal linking mechanisms for Artist products sold in the Shop
- Site: Tickets Sales and Ticket Production.
- Site: Artist specific Banner Ads
- Site: Integration of a Wordpress Blog seamless with the site design
- CMS: Easy uploading of site images in varying sizes
- CMS: Automatic Resizing and cropping of images
- CMS: Easy uploading of multiple, large .mp3 files
- CMS: Sampling of .mp3 files for previewing on the site
- CMS: Management of both Album and Singles Sales
- CMS: Management of Digital Products other than music
- CMS: Management of Physical Products for shipping
- CMS: Shop Categories & Sub-Categories
- CMS: Management of Banner Ads
- CMS: Management of Friendly URL's
- CMS: Management of Meta Tags
- CMS: Management of user maintained HTML content
- CMS: Product and Event Management
- CMS: Event seat allocation between Highway 125 & Moshtix
- CMS: Order Management and Processing

This phase of the development was to be delivered as a Beta Release by the 28th of February 2011 and this was indeed the case with a presentation to the customer on the 28th.

All through the development the solution had been built with future planning in mind for expansion and the necessity to deploy across a Web Farm if required. Towards the end of the build process this was discussed further with the client and it was decided that the step to future proofing the solution would be better taken sooner rather than later. Thus the decision to move the entire solution to The Cloud (apart from WordPress blog) and to Microsoft Azure was taken.

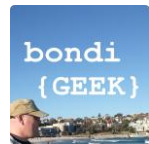
The Road to Azure

Moving to Windows Azure and SQL Azure was a relatively straightforward move and an extremely cost effective one for the client who had been quoted solutions running in to the several thousands of dollars per month. None of these solutions could offer the future proofing offered by Azure and the ease of scalability.

What would have cost 1000's of dollars per month was reduced to 100's of dollars per month and with the ability to scale when required along with reliability and recoverability backed by a 99.9% uptime guarantee.

Some of the key factors and benefits to moving to Azure were:

- Ability to switch on a global CDN solution when required to reduce latency and increase scalability
- Ability to distribute the solution globally when required to reduce latency and increase scalability



- Ability to switch to a distributed cache when required.
- Ability to scale up the existing solution from 3 servers to 10,20, 100 or even 1000, if necessary, with zero downtime
- Easy management of SSL Certificates from the Azure portal across all servers.
- Cheap monthly cost of SQL Azure database
- Cheap monthly cost of blob storage for images and media
- Ability to deploy a new version of the solution and perform a VIP Swap from Staging to Production with zero downtime.

All of these benefits were afforded to the client with very little extra work required.

The challenges that needed to be overcome were:

- Creation of a custom service and installer for the Azure deployment package to handle encoding and sampling of .mp3 files utilizing FFMPEG. A blog post on how this was achieved can be found [here](#)
- Redirecting all media and image files to Blob storage and creating custom routines to push them to Blob storage from the CMS
- Customizing the SquishIt Framework, for script combining and compression, to work with Azure Blob storage. A blog post on how this was achieved can be found [here](#)
- Altering the SQL Azure database to work with the ASP.NET Membership provider
- Custom Ticketing creation and store in Blob Storage

This was all achieved within the client's budget and delivered on schedule to the client by April 1 2011.

Since the beginning of April the client has continued to test the solution and a few modifications have been made and tweaks here and there. However, additional time was required by the client to prepare for launch and get certain legal and contractual arrangements in place.

Solution Components

| Client Side | | Server Side | |
|------------------|-------------------------|-----------------------------|--------------------|
| HTML | QapTcha - jQuery Plugin | ASP.NET | WCF Data Services |
| CSS | Sound Manager | ASP.NET Membership Provider | Windows Azure |
| jQuery | Uploadify | C# | SQL Azure |
| jQuery Templates | CMS: TinyMCE | MemCache | FFMPeg |
| jScrollPane | CMS: jqPlot | EntityFramework 4.1 | SquishIt Framework |