Saimaa University of Applied Sciences
Technology Lappeenranta
Degree Programme in Information Technology
ICT-entrepreneurship

Antti Mäkelä

First steps into EPiServer CMS 7
Case: Seepia Games

Thesis 2015

Tiivistelmä

Antti Mäkelä
First steps into EPiServer CMS 7 Case: Seepia Games, 21 sivua
Saimaan ammattikorkeakoulu
Tekniikka Lappeenranta
Tietotekniikka
ICT-yrittäjyys
Opinnäytetyö 2015
Ohjaaja: TkT, tietotekniikan lehtori Pasi Juvonen, Saimaan ammattikorkeakoulu

Tämän opinnäytetyön tavoitteena oli kerätä ja reflektoida kirjoittajan kokemuksia oppimisprosessista EPiServer CMS 7 -sisällönhallintajärjestelmän kanssa. Tämä tapahtui käymällä läpi case-projektin vaiheita. Projekti toteutettiin Seepia Games -yritykselle jonka toiminta perustuu mobiilipelien ympärille.

EPiServer on yritys, jonka toiminta keskittyy pääasiassa pohjoismaihin. Heidän tuotteinaan ovat sisällönhallintaan ja sähköiseen kaupankäyntiin keskittyvät ohjelmistot. Sisällönhallintajärjestelmällään he keskittyvät tarjoamaan joustavan ympäristön web -kehitykseen. Pääasiallinen tiedonlähde koskien EPiServer CMS:ää oli EPiServer World -yhteisösivusto.

Tämän opinnäytetyön tarkoitus on olla opas aloittelevalle EPiServer CMS-kehittäjälle joka haluaa päästä sen kanssa alkuun. EPiServer CMS ei ole kovin laajalle levinnyt sisällönhallintajärjestelmä, joten avun löytyminen internetistä on jokseenkin haasteellista. EPiServer World -yhteisösivusto tarjoaa paljon ohjeita kehittäjille, mutta monissa tapauksissa niiden kirjoittajat ovat olleet siinä olettamuksessa, että lukija osaa kaikki tarvittavat taidot. Tässä opinnäytetyössä pyritään tuomaan esille nämä sudenkuopat ja tarjoamaan ratkaisuja erinäisiin haasteisiin.

Avainsanat: EPiServer, sisällönhallintajärjestelmä, opas, käyttökokemukset

Abstract

Antti Mäkelä
First steps into EPiServer CMS 7 Case: Seepia Games, 21 pages
Saimaa University of Applied Sciences
Technology Lappeenranta
Degree Programme in Information Technology
ICT-entrepreneurship
Bachelor´s Thesis 2015
Instructor: D.Sc.(Eng.) Senior Lecturer Pasi Juvonen, Saimaa University of Applied Sciences

The main goal of this thesis was to collect and reflect on experiences from learning process while working with EPiServer CMS 7 content management system. This was done by going through phases of a case project. The project was done for a company called Seepia Games, whose activity is focused on mobile games.

EPiServer is a company whose activity is focused on the Nordic countries. They provide content management and e-commerce software. The focus of their content management system is to provide a more flexible environment for web development. Regarding EPiServer CMS, the main source of information was the EPiServer World community page.

This thesis is aimed to being a guide for a starting EPiServer CMS developer who wants to get started with it. EPiServer CMS is not a very widespread content management system and therefore finding help from internet might be tricky. Although the EPiServer World community page offers a lots of guidance for developers, most of it is written with the assumption that the reader is familiar with all the required skills. In this thesis it is pursued to disclose these pitfalls and to offer answers for particular challenges.

Keywords: EPiServer, content management system, guide, user experiences

Content

1 Introduction 6
1.1 Introduction of client 6
1.2 Motivation for the thesis 7
1.3 Goals and confining 7
1.4 Introduction of the author 8
1.5 The structure of thesis 8
2 EPiServer content management system 9
2.1 Content management systems 9
2.2 EPiServer 9
2.3 ASP.NET MVC 10
2.4 Architecture of EPiServer 11
3 Course of the project 12
3.1 Project management framework 12
3.2 Setting up working environment 12
3.3 The construction of case web page 13
3.4 Deploying to Azure 16
4 Conclusion 18
4.1 Challenges 18
4.2 Summary 18

Acronyms abbreviations and notations

CMS	Abbreviation of Content Management System
C#	Programming language developed by Microsoft
HTML	Abbreviation of Hypertext Markup Language. Used to formulate web pages
CSS	Abbreviation of Cascading Style Sheet. Used to describe looks and formatting of web page
JavaScript	Programming language commonly used as part of web browser for client-side scripting
jQuery	JavaScript library for simplifying client-side scripting in HTML
PHP	Server-side scripting language for web development
MySQL	Relational database management system
Razor	View engine for ASP.NET
cshtml	File type for ASP.NET MVC Razor C#. Includes code for view

1 Introduction

The aim of this thesis is to introduce the EPiServer web content management system through a case project and reflect on experiences of an author who is working with it for the first time. The author has worked in numerous website projects and he is familiar with the most globally used web development languages and tools. He also has experience with WordPress content management system. Use of EPiServer CMS for this project was requested by Seepia Games.

Development environment for EPiServer content management system is Visual Studio. For EPiServer CMS 7, which was the version used in this project, Visual Studio 2010 or later is required. In this case Visual Studio Express 2013 for Web was chosen since it is free to use.

At the time of this project Seepia Games had temporarily halted the development of their Windows Phone game Pet Shows. This period of time was favorable to develop it for other platforms, Android and iOS, and make preparations for launching the game for said platforms. These preparations include conversion of game code to support these new devices and also updating the games promotional web site.

All sources of information and help needed in this project are from the internet. Mainly, documentation on world.episerver.com was used, but some helpful guides were located in blog posts on tedgustaf.com written by Ted Nyberg.

1.1 Introduction of client

The client in this project is a game development company Seepia Games Ltd. Their focus has mainly been in the development of mobile games. Their most notable game has been Permia Duels for Windows Phone and iPhone. Business premises of Seepia Games are located at Technopolis Skinnarila, Lappeenranta, Finland.

1.2 Motivation for the thesis

The author with two other students from ICT enterpreneurship study path, Ville Jurvanen and Toni Pulkkinen had interest in game development, so it was decided to contact Seepia Games in hopes of finding a project on said subject. Seepia offered a platform conversion project concerning their Windows Phone game Pet Shows alongside with promotional web page update using EPiServer CMS.

During the planning of the project, the author saw opportunities in EPiServer CMS, so he decided to accept the challenge and study something new. This is how EPiServer CMS was selected as subject for this thesis.

1.3 Goals and confining

The first goal of this thesis is to explain what skills and tools are required to get started with EPiServer CMS and where to find information concerning said subject. This is done by reviewing some parts of the case project.

The econd goal of this thesis is to collect and reflect the author's experiences when taking first steps into developing with EPiServer CMS. Based on those experiences, some improvements are presented regarding available tutorial material.

Seepia Games provided clear requirements for the web page. The structure and the visual appearance are based on a concept created by Joram Bosker. Figuring out deployment procedures are also included in the project. In addition, Seepia Games wants documentation about the experiences regarding developing with EPiServer CMS.

The text is aimed to be kept at general level and the amount of coding related matters are minimized. This is to help non-programmers to get a better grasp of this thesis and to keep the text more consistent.

1.4 Introduction of the author

Since a large part of this thesis covers the experiences of author during this case project, it is essential to explain his background concerning web development and programming in general. He has studied programming for 7 years, 3 years in vocational school and 4 in a university of applied sciences. In Saimaa University of Applied Sciences, the majority of programming was learned via customer projects in form of web sites and web applications.

In web development, the author's the most used languages consist of HTML, PHP and JavaScript. The author's skills improved more when customers started to request more complex applications requiring solutions with jQuery libraries and applications requiring interaction with MySQL databases. Simpler web page projects needing features for updating content led to the use of WordPress.

Visual Studio and C# were familiar environments to the author, but he has not used them actively before this project. The author had only used them in two programming courses in the first year of studying in Saimaa University of Applied Sciences. No knowledge of .NET or EPiServer was acquired before this thesis project.

1.5 The structure of thesis

This thesis begins with an introduction to the project. The intro includes information about the project and the customer. The second chapter covers the introduction to EPiServer CMS and the explanation of its main differences with other web content management systems. In addition, some terms and technologies are clarified.

The third chapter describes the course of the project starting from the formation of team rules and project management framework. In addition, the third chapter includes information about what procedures were taken to create a web page for this project and to deploy it in Microsoft Azure cloud service. The last chapter covers the conclusion and reflection, i.e. what was achieved, what was learned and what the author would like to see improved regarding tutorials and guides of EPiServer CMS.

2 EPiServer content management system

2.1 Content management systems

Some key features of content management systems are streamlining content administration over web site, implementing Web-forms-based content management and separating content from layout and design (Powel & Gill, 2003).

Content management system, or CMS in short, makes setting up web pages an easy task. In some cases, the knowledge of HTML may not be needed at all. CMS is a great choice for web pages including content, that requires frequent updating and management.

At the moment, the most used CMS in the world is WordPress, which is free to use due to it being open source. These factors lead to a large developer base, which makes finding help from the internet simple.

2.2 EPiServer

EPiServer CMS is a web content management system created by the software company EPiServer, originating from Sweden. The company is focused on providing web content management and e-commerce software. Since the company and CMS share the same name, in this thesis it is aimed to state if it is referred to company or content management system.

EPiServer CMS is developed in Visual Studio, therefore main programming language is C#. XHTML and CSS are used for templates and styling as in any other web page. EPiServer CMS web sites consist of pieces called pages and blocks. Both can be used in the same way, but after the release of EPiServer CMS 7 and introduction of blocks, they are advised to be used for construction of the site.

While EPiServer CMS is a content management system and it is used to do same the action as other similar systems, it should not be compared directly to

other content management systems. The same could be said about other content management systems, too (Nyberg, 2011).

Since the author has experience only from WordPress and EPiServer CMS, it would be convenient to cover those. As described previously, direct technical comparison would not tell much, but some differences can be noted. WordPress is simple and easy to use, but it is built on many assumptions on how a web page should be structured. However, for example a small budget and simple web site needs could be source for choosing WordPress (Nyberg, 2011).

EPiServer would be a good choice for a large company with a multilingual client base and plans for more complex website design needing customizable backend features. It all comes down to the requirements for a web site. The strength of EPiServer CMS is its flexibility, but this feature can be perceived as a double edged sword. EPiServer CMS does not provide very many ready solutions and it requires a greater amount of work to put into it. That is why it is more like a framework for creating a content management system for any need than simpler content management systems.

2.3 ASP.NET MVC

ASP.NET is a web application framework created by Microsoft. It is used to create dynamic web sites and web services and MVC (Model-View-Controller) is the architectural pattern for it.

The ASP.NET MVC framework offers an alternative to the ASP.NET Web Forms environment. ASP.NET Web Forms does not require handling raw HTML code during development. ASP.NET MVC takes the developer slightly deeper, so knowledge about HTML is required (Shepherd 2010, 449). Since full support of MVC is a new feature for EPiServer CMS 7 most of the tutorials and guides were based on this new version of EPiServer CMS.

MVC consists of model, view and controller components. Every object in web pages created with EPiServer requires all three of these components. The model component includes properties for objects. The view component defines how to display properties given in the model component. The controller compo-

nent handles user interaction and selects view for rendering. Interaction of these components are shown in figure 1.

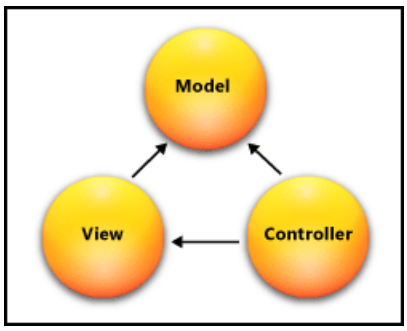

Figure 1. MVC design pattern. Microsoft Developer Network - ASP.NET MVC Overview

Every element on a web page created with EPiServer CMS requires all of these three components. In some cases it is a good practice to create a base model class including recurring properties, which can be inherited to other models.

2.4 Architecture of EPiServer

Figure 2 shows the structure of EPiServer platform, which consists of the EPiServer framework with a core user interface, CMS and commerce for content management and e-commerce, and other supporting modules for marketing and social interaction features.

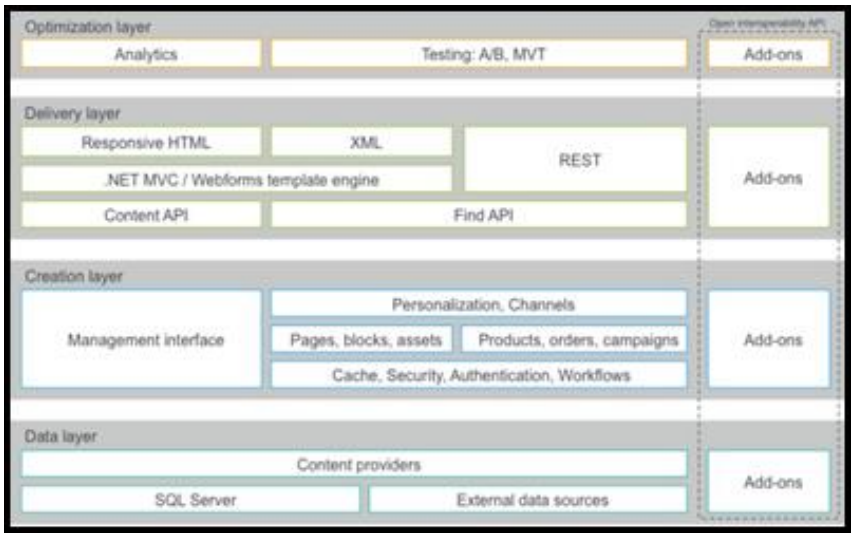

Figure 2. Overview of EPiServer platform. EPiServer World documentation - Architecture

Information originating from database, content providers and integrated external data sources are provided by the data layer. The creation layer manages content of pages and blocks or e-commerce content. The delivery layer holds the Web Forms and MVC based design and templates as well as support for advanced search and filtering features. Measuring, analyzing and optimizing of performance are provided by the optimization layer.

3 Course of the project

3.1 Project management framework

This project was done by a small team, rather than individually, so it felt appropriate to settle some rules and practices since the whole team was not able to be present at all times. It was decided to use an agile development model, Scrum. Only some elements of it was used, since our team size was three and the minimum team size of Scrum is five (Cohn 2010, 178).

It was decided to have meetings two times a week. The purpose of these meetings was to map the progress of every individual between meetings and set goals for the next meeting. In addition, if any problems were to appear, they were solved together during the meetings. The meetings were documented alongside with a backlog in case of need for retrospective examination. All documentation were shared among team members via Google Drive.

Since most of the time the team was not able to work together in a same location, an appropriate messaging system was required. Skype was chosen, because it has instant text messaging and VoIP capabilities with a group creation feature.

3.2 Setting up working environment

Microsoft Visual Studio is a working environment for EPiServer projects and for the newest version of EPiServer, EPiServer CMS 7, Visual Studio 2010 or later is required. Microsoft is offering a free Visual Studio Express 2013 for Web, so it was used. Visual Studio also needs EPiServer extensions and NuGet package control references before it is possible to start an EPiServer CMS project.

EPiServer provides getting started documentation on their EPiServer World page. This guide gives instructions to creation of EPiServer project and starting page without template (Start developing with EPiServer CMS. http://world.episerver.com. 12 August 2015).

3.3 The construction of case web page

The web page created in this project was based on a concept seen in figure 3. It is designed by Joram Bosker. Ready to use images were not available, so they were created with Photoshop using assets from game files of Pet Shows. Alongside text and images, the page was also to contain an embedded Pet Shows trailer video from YouTube and an interactive game character introduction section with clickable elements.

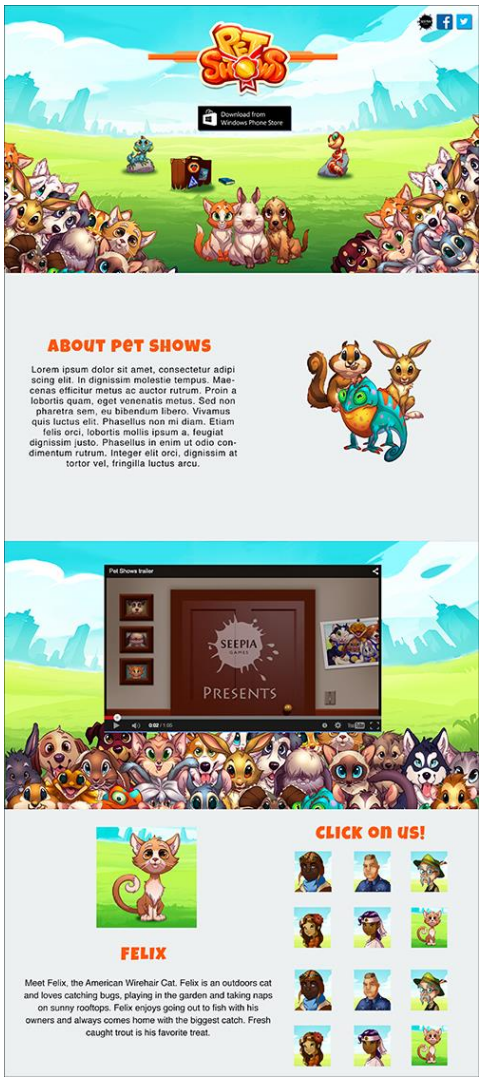

Figure 3. Concept for Pet Shows web page designed by Joram Bosker

The first task was to set up the structure for web page. This was achieved by creating extremely simplified template as a cshtml file using basic HTML elements. The command for rendering main content area was added inside the wanted HTML element within template file. Then, EPiServer CMS also needed a file which declared where the web page finds template file.

The next step was to create blocks. Blocks are elements having properties declared in assigned model file. A good practice when designing block pattern is to cut layout into pieces as shown with red squares in figure 4. Each square represents one block. This way it is easier to perceive the web page structure created under rules of EPiServer CMS.

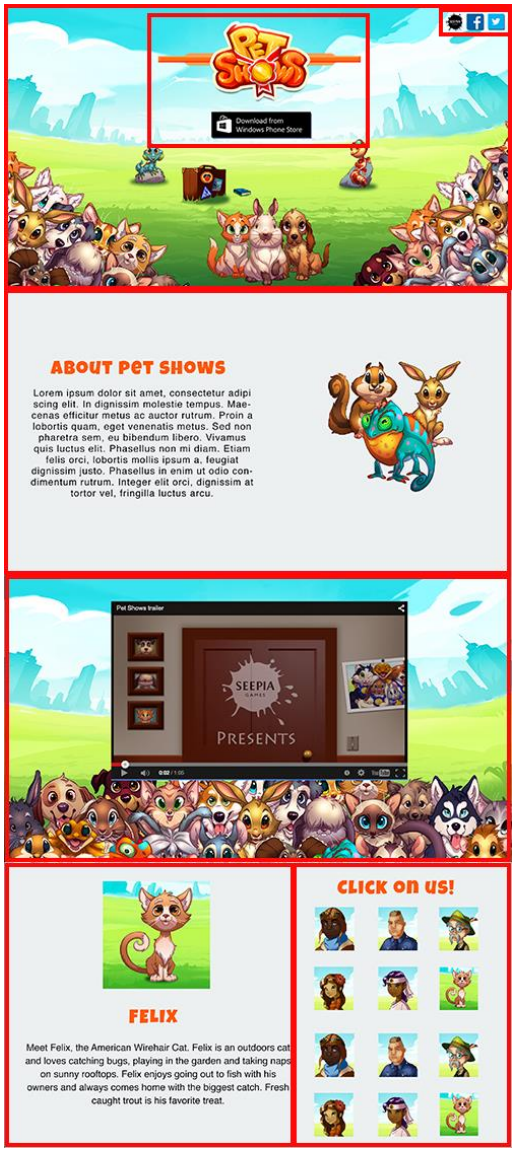

Figure 4. Block placement highlighted with red squares.

For example, the section shown in figure 5 has the header, text and image highlighted with red squares. All those elements are declared in a model file. View file alongside CSS style sheet, helps to adjust elements to their right places and set fonts and colors. In this phase, a controller is also required. In most cases the controller does not need more than automatically generated code by Visual Studio. Only parts requiring modifications are referred class names.

Figure 5. Highlighted elements of one section from web page.

Most elements in this web page were created the same way as described above. Embedding a video from YouTube required some creativity. Some discussion pages at EPiServer World suggested creating a plugin for embed videos, but in this case the author's range of knowledge alongside with limited time forced to creation of simpler solution. It is possible to create a XHTML string property within model. This property has a feature to paste an iframe -element to a text field. This feature was used to copy and paste an iframe -element found from each YouTube page.

Two of the sections were to have background images. The simplest way to implement this would have been defining wanted images in a css file, but this solution requires access to said file and it is against the aim of creating a highly customizable web page. To achieve this goal, the property for background image must be defined in model, but since model of MCV does not have this op-

tion, some creativity was needed. The solution was to use a normal property for image, but in a view file it was wrapped inside a class. The values declared in css made this class act as a background element and since it was created as a block element, it was possible to use the same element with different images in multiple places on page.

The functionality of character introduction was created with jQuery. Any apparent solution using features of EPiServer CMS was not found, so it was done in a traditional way by writing the script within a template file. The basic idea of this functionality is to show the information regarding first character in the list at page load. When icons of other characters are clicked, previously shown information is hidden and new one shown corresponding clicked icon. This required creation of two blocks, the other containing information and the other an icon image. Both blocks were provided with a properties for identifier string, which were used for linking icon and information. This solution combined with a simple jQuery script leads to a functional element.

At this point the entire web page was getting complete and working locally. Brief testing with content management systems editor was performed to make sure everything was working as intended. When everything seemed to be in order, it was time to start the procedures for deployment.

3.4 Deploying to Azure

EPiServer World has specific instructions for deploying a web site to Windows Server, Amazon Web Services and Microsoft Azure. In this project, the focus is on Azure, as requested by Seepia Games. A free trial of Azure and a demo license of EPiServer CMS were used to test deploying in practice. This means that not all features were available during this project.

The first steps of deployment are to create Azure resources and by following the documentation of EPiServer World, this is quite easily done. A basic site requires a web app, storage and service bus to be created within the Azure management portal (Deploying to Azure websites. http://world.episerver.com. 18 August 2015).

While EPiServer deployment instructions describe creating a web site, a new developer should not get disoriented, since the newest version of Azure uses name web app, instead of web site. It should also be noted, that a default domain name is in the form domain.azurewebsites.com, but this can be changed through the web app configuration. While creating web app, SQL database should be created as well.

All media on a web page should be stored in the Azure binary large object storage, or commonly known, the BLOB storage. As stated in the deployment guide, the storage name should be in lowercase for DNS compatibility. The next step is the creation of the service bus. The purpose of the service bus is to provide an ability to share data between decoupled systems. It had no major role in this project, so the thesis does not focus more on that.

When all resources are ready, it is time to update configurations in Visual Studio. Before continuing, the developer has to install the NuGet package EPiServer.Azure in Visual Studio. When this is done, the developer only needs to follow the steps of the deployment guide and add required connection strings to web.config file. The author likes to note that it is advisable to take a backup copy of this file, or at least leave changed lines commented. This makes transition to local environment simple in case of error situations in deployment process.

Making the configurations explained in the previous step opens a new feature in Visual Studio. Right clicking the project name in solution explorer allows publishing. In this part the deployment guide stumbles slightly, since it never describes where to find required publish settings files. These files can be found from under created web app in Azure. There user can find link named download the publish profile.

When working locally, the developer can use his Windows username and password, but after uploading the webpage to Azure, it is recommended to create new login information. Allowing access of source IP address is required first. This is done in Azure navigating to database section and by selecting database under server tab. Now that Visual Studio is able to access the content man-

agement system on Azure server, it is only required to follow introductions to create new login information and deployment is complete. Since all this was done with a free trial version of Azure, it was not possible to change the URL.

4 Conclusion

4.1 Challenges

Although the author has decent skills in traditional web development, working with EPiServer CMS was not an easy task. EPiServer World provided good guides and documentation to get started with project, but it lacks instructions for creating anything concrete. Also there seems to be no clear mention about what skills the developer should have before beginning, so a trial and error style development might lead to several dead ends. If the developer is not familiar with .NET framework, he should explore the learning section at asp.net, especially tutorials concerning MVC and Razor.

When the working environment was set up and building the web page was to be started, world.episerver.com did not seem to contain instructions for the next step. At this point, the blog posts of Ted Nyberg on tedgustaf.com stepped in. He has a good guides regarding EPiServer CMS 7, though his style is one step more advanced than a beginner would like. Anyways, his posts gave some direction and helped to move forward with this project.

4.2 Summary

EPiServer CMS can be a powerful tool for creating web pages when used right, albeit it requires many times more work to get the website running than other content management systems. The biggest challenge for new developers might come from the lack of instructions and tutorials which might be a result of a small developer community compared to other more popular technologies and systems.

The main source of information and instructions to work with the EPiServer CMS is EPiServer World, but alone that is not enough for a starting developer, since most of the instructions seems to be written with the presumption that the

reader is familiar with all required technologies and tools. Tutorial projects starting from scratch and advancing step-by-step may be the personal preference of author, but EPiServer could benefit from having that kind of a document included in their EPiServer World site. The author was able to find this kind of page for EPiServer CMS 6 from an external source, but it was not too helpful due to major changes between versions 6 and 7.

While studying ASP.NET tutorials, the author noticed a great practice of listing required skills at the beginning of tutorial, so the reader knows exactly how to prepare for the tutorial in question (Learn. http://asp.net. 10 August 2015.). Since EPiServer CMS is a complex system which requires understanding of multiple different technologies, it would be delightful to see same kind of tutorial solutions implemented for EPiServer CMS.

All in all the author's inexperience of systems like EPiServer CMS may have played its part in making this project challenging, but some improvements could be made to tutorials and instructions provided EPiServer company itself. Beginners may have hard time for finding information from the internet, due to a seemingly small developer community. Tutorial improvements could result in more new developers, which in turn could generate more, and above all, beginner friendly tutorials.

Figures

Figure 1. MVC design pattern. Page 11
Figure 2. Overview of EPiServer platform. Page 11
Figure 3. Concept for Pet Shows web page. Page 13
Figure 4. Block placement highlighted with red squares. Page 14
Figure 5. Highlighted elements of one section from web page. Page 15

References

Cohn, M. 2010. Succeeding with Agile: Software Development Using Scrum, Boston: Addison-Wesley.

Deploying to Azure websites. n.d. http://world.episerver.com/Documentation/Items/Developers-Guide/EPiServer-CMS/75/Deployment/Deployment-scenarios/Deploying-to-Azure-websites/. Accessed on 18 August 2015.

Learn. asp.net. http://www.asp.net/. n.d. Accessed on 10. August 2015.

Nyberg, T. 2011. WordPress vs. EPiServer CMS http://tedgustaf.com/blog/2011/2/comparison-of-episerver-and-wordpress/. Accessed on 10 August 2015.

Powel, W. & Gill, C. 2003. Web Content Management Systems in Higher Education http://net.educause.edu/ir/library/pdf/eqm0325.pdf. Accessed on 12 August 2015.

Shepherd, G. 2010. Microsoft ASP.NET 4 Step by Step. Redmond, Washington: Microsoft Press.

Start developing with EPiServer CMS. n.d. http://world.episerver.com/documentation/cms/get-started-with-cms/. Accessed on 12 August 2015.