SPIRIT/21

DHBW
Duale Hochschule
Baden-Württemberg

Darstellung von Benutzerinformationen in einer Webapplikation (programmiert mit C# und ASP.NET)

Modul T3100

Studienarbeit

Studiengang Angewandte Informatik

an der Dualen Hochschule Baden-Württemberg Stuttgart Campus Horb

von

Jens-Christian Willmer

27.02.2012

Bearbeitungszeitraum	12 Wochen
Matrikelnummer, Kurs	8189582, TAI2009
Universität	DHBW Stuttgart Campus Horb
Betreuer der Universität	Schneider Ulrich

Selbstständigkeitserklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit mit dem Thema

„Darstellung von Benutzerinformationen in einer Webapplikation“

Selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Böblingen, den _____ _____

Jens-Christian Willmer

Zusammenfassung

Diese Arbeit befasst sich mit der Erstellung einer Homepage, die dem angemeldeten Benutzer ermöglicht, sein E-Mail-Kontingent sowie sein verbleibendes Kontingent auf der privaten Dateiablage der Universität einzusehen.

Die Arbeit gliedert sich hierzu in zwei Phasen. Zu jeder Phase gibt es eine Ausarbeitung, wobei sich diese mit Phase eins befasst. In diesem Teil der Arbeit wird der Schwerpunkt auf die Grundlagen gelegt und es wird damit begonnen, die Projektstruktur aufzubauen sowie die Anmeldung am Active Directory zu realisieren.

Die Grundlagen umfassen hierbei den Microsoft Windows Server, den Microsoft SQL-Server, das .NET-Framework, die Entwicklungsumgebung Visual Studio 2010 sowie das Entwicklungsmuster Model-View-Controller.

Die Umsetzung des Projekts wird unter Verwendung von Visual Studio 2010 und ASP.NET MVC 3 angegangen. Es wird hierzu beschrieben, an welchen Stellen die Anwendung und der IIS konfiguriert werden müssen, um eine verschlüsselte Verbindung zuzulassen und wo die Anmeldedaten für das ActiveDirectory abgelegt werden. Des Weiteren werden Daten aus dem ActiveDirectory in einer lokalen Datenbank zwischengespeichert, welche ebenfalls beschrieben wird.

Inhaltsverzeichnis

1. Einleitung

1.1 Schilderung des Umfelds

Angestellte und Studierende der Dualen Hochschule Baden-Württemberg (DHBW) Stuttgart Capus Horb besitzen jeweils ein eigenes E-Mail-Postfach sowie einen privaten Speicherbereich auf dem Dateisystem der Universität.

Das E-Mail-Postfach wird von einem Exchange Server gehostet, welcher im Netzwerk der Universität steht. Der Fileserver, der sich ebenfalls im Netz der Hochschule befindet, arbeitet mit einer Unix-Plattform. Die Zugänge zu den Servern werden über ein Active Directory geregelt, welches die Anmeldedaten der Anwender enthält und bei korrekter Authentifizierung den Zugriff auf die Ressourcen erlaubt.

Sowohl das E-Mail-Postfach als auch das Benutzerverzeichnis unterliegen hierbei einer Größenbeschränkung. Eine einfache Möglichkeit den verbleibenden Speicherplatz einzusehen gibt es nicht. Es ist jedoch möglich, den verbleibenden Speicher auf Grundlage des verwendeten Speichers und der im Intranet ausgeschriebene Maximalgröße zu errechnen.

1.2 Problemstellung

Aufgrund des benötigten Aufwandes, der für den Anwender entsteht, eine Übersicht über seinen verbleibenden Speicher zu erlangen, ist es angedacht, hierfür eine Übersichtsseite (Dashboard) zu entwickeln. auf welcher diese Informationen aufbereitet abgerufen werden können. Des Weiteren soll diese Seite noch eine Suche enthalten, mit welcher Benutzer auf Grundlage von Vor- und Nachname einer Person deren E-Mail-Adresse suchen können. Diese Seite soll aufgrund der existierenden Windows Server in ASP.NET und C# geschrieben werden.

Für die Anzeige der nutzerbezogenen Daten muss der Benutzer sich anmelden. Dies geschieht über einen Abgleich mit dem Active Directory. Daraufhin sendet die zu entwickelnde Webapplikation eine Anfrage gegen den Exchange Server, um die aktuelle und maximale Größe des Anwenderpostfachs zu erfahren.

Eine zweite Anfrage versendet das Programm an den Unix Server, um auch hiervon die Größenangaben zu erhalten. Die Kommunikation zwischen Unix Server und dem Dashboard soll hierbei über eine SSH-Verbindung stattfinden. Die von den beiden

Anfragen erhaltenen Daten werden daraufhin aufbereitet und dem Anwender im Dashboard angezeigt.

Die Suche nach E-Mail-Adressen soll über das Active Directory realisiert werden. Hierbei ist angedacht, die Attribute der angelegten Personen zu durchsuchen, um deren kompletten Namen sowie die E-Mail-Adresse zu finden. Werden auf Grund von Suchanfragen, die nur einen Teil des Namens enthalten, mehrere Übereinstimmungen gefunden, sollen die ersten 15 Treffer ausgegeben werden.

Weiterhin wird für die Anzeige der Daten auf dem Dashboard ein Design benötigt, welches mit dem bereits vorhandenen Design des Intranets korrelieren sollte. Hierzu wird ein Designkonzept benötigt.

Die beschriebene Problemstellung ist Inhalt der Seminararbeit. Diese erstreckt sich über zwei Semester, wobei für jedes Semester eine Ausarbeitung anzufertigen ist. Mit dem ersten Teil der Ausarbeitung beschäftigt sich diese Arbeit, wodurch der Schwerpunkt auf den theoretischen Teil gelegt wird. Dieser befasst sich im Grundlagenteil mit dem Aufbau des Windows Server, dem Designkonzept Model-View-Control sowie Grundlagen in der .NET-Programmierung. In Kapitel 3 wird anschließend das praktische Vorgehen beschrieben.

2. Grundlagen

2.1 Microsoft SQL-Server

Der von Microsoft entwickelte SQL-Server ist das führende, serverseitige, Datenbanksystem auf Windowsplattformen. Ein Datenbanksystem ist dazu da, Datensätze abzuspeichern und diese auf Anfrage wieder auszuliefern. Dabei werden nicht die Rohdaten zurückgeliefert, sondern abhängig von der Anfrage eine aufbereitete Ansicht auf die betroffenen Datensätze.

Der Microsoft SQL-Server besteht aus einer relationalen Datenbank (Abbildung 1). Diese besteht aus einer Sammlung von Tabellen, die sogenannten Relationen. Die Zeilen, auch Tupel genannt, enthalten jeweils einen Datensatz. Dieser besteht aus Spalten, welche die Eigenschaften (Attribute) des Datensatzes definieren. Mit dem Relationsschema werden die Anzahl und der Typ der Spalten festgelegt und mit der relationalen Algebra werden Abfragen auf die Relationen beschrieben.[1]

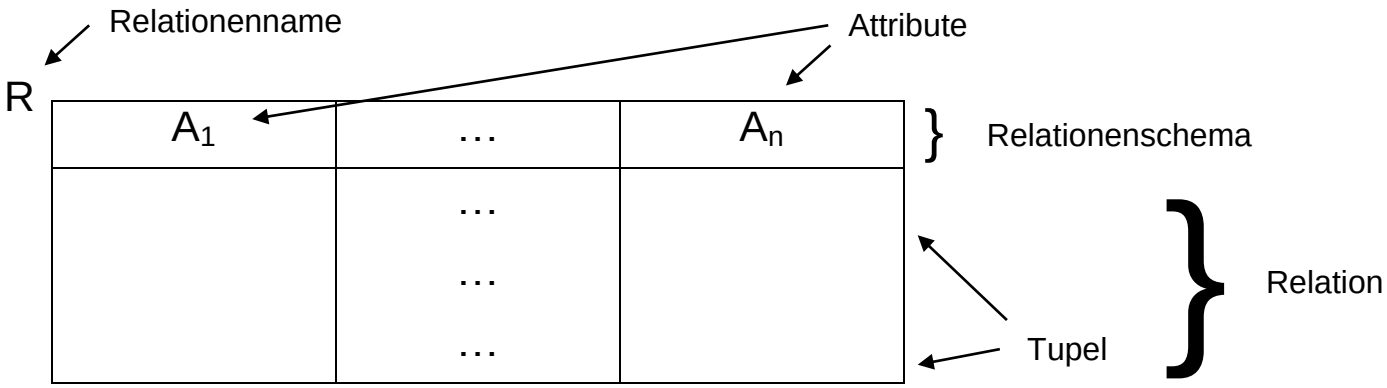

Abbildung 1: *Relationenmodell*[2]

Der SQL-Server ist in drei Schichten unterteilt: die Datenschicht, die Programmschicht und die Präsentationsschicht. Die Datenschicht ist verantwortlich für die Speicherung und Verwaltung der Daten. Die Programmschicht ist für die Abbildung der Logik und des Datenzugriffs verantwortlich. Die Präsentationsschicht

[1] Vgl. Kowarschick, W. (2011)
[2] Enthalten in Kowarschick, W. (2011)

ist für die Anzeige der Ausgaben, sowie die Interaktionen des Anwenders mit der Oberfläche verantwortlich.[3]

2.2 Windows Server 2008

Der Windows Server 2008 ist ein von Microsoft entwickeltes Betriebssystem, welches eine stabile Plattform für den Betrieb von Anwendungen bereitstellt. Windows Server 2008 ist der Nachfolger von Windows Server 2003 und baut auf dem Betriebssystem Microsoft Vista auf. Mit der Veröffentlichung von Windows Server 2008 Release 2 (R2) wurde die Unterstützung für die maximale Anzahl an CPUs erhöht und die Unterstützung der 32-Bitversion eingestellt.

Windows Server 2008 besitzt einen streng modularen Aufbau, der es dem Administrator ermöglich, nur die Rollen und Funktionen zu installieren, die er benötigt. So ist es möglich, eine minimale Installation auszuführen, wodurch lediglich eine Konsole zur Verwaltung des Servers vorhanden ist und auf die grafische Oberfläche und deren Werkzeuge zu verzichten.[4]

2.2.1 Microsoft Internet Information Services

Der Microsoft Internet Information Services (IIS) ist eine Dienstplattform von Microsoft und ist Teil der Windows Server Installation. Mit ihm ist es möglich, Dateien und Dokumente über das Netzwerk zugänglich zu machen. Des Weiteren bietet der IIS umfangreiche Konfigurationsmöglichkeiten und kann ASP.NET, PHP und JSP Anwendungen betreiben (Abbildung 2). Charakteristisch für den IIS ist die Skalierbarkeit des Systems sowie die Sicherheit und Produktivitätssteigerung.[5]

[3] Vgl. Konopasek, K (2010) S.16f
[4] Vgl. Boddenberg, U. (2010) Kapitel 1.2
[5] Vgl. Microsoft Corporation (2012)

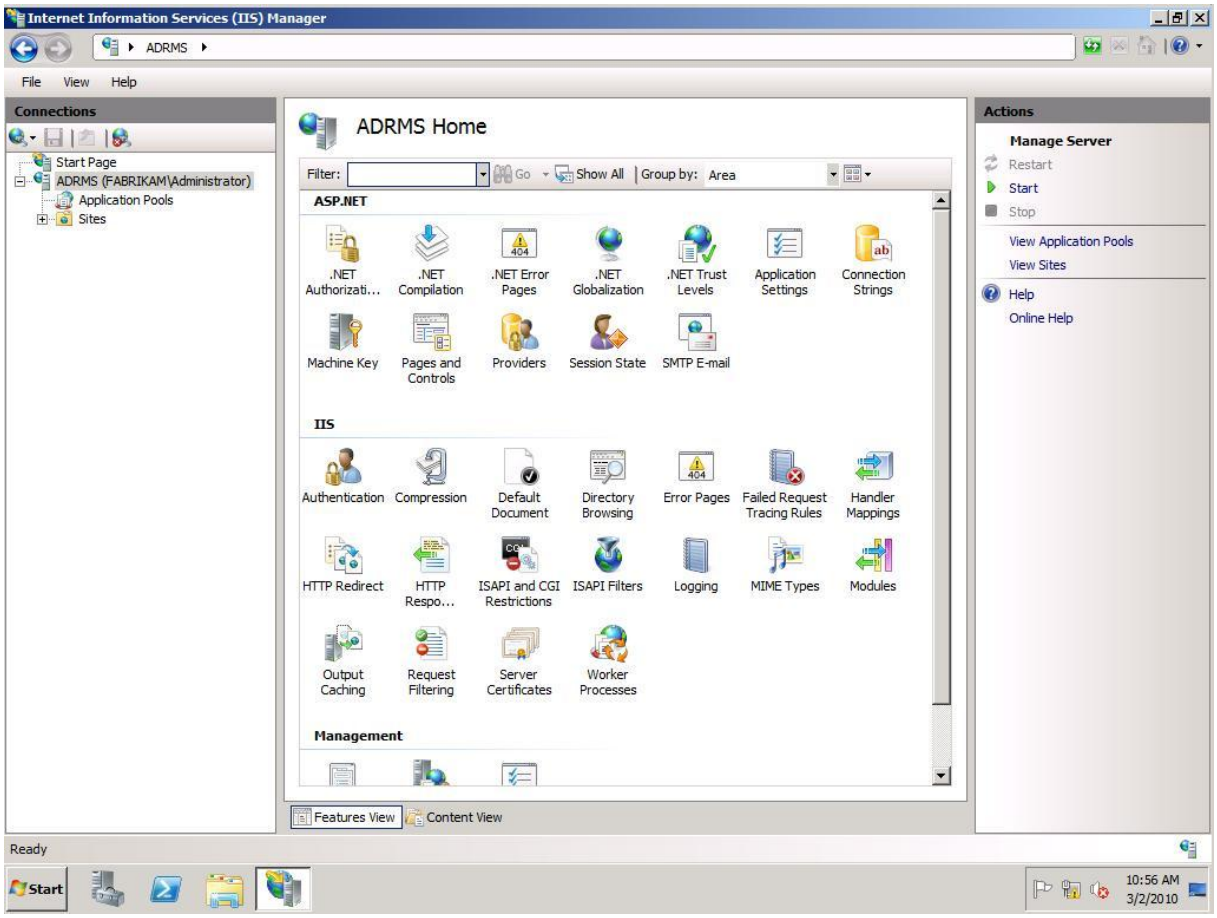

Abbildung 2: *IIS Konfigurationsoberfläche*[6]

2.2.2 Active Directory

Das Active Directory (AD) ist ein Verzeichnisdienst von Windows. Mit ihm ist es möglich, das zugrundeliegende Netzwerk in einer logischen Struktur zu organisieren. Es übernimmt hierbei die Aufgabe der Authentifizierung der im Netzwerk befindlichen Geräte sowie die Regelung der Zugriffsberechtigungen.

Die Struktur des ADs ist gegliedert in Domäne, Tree, Forest, Namensraum und Organisationseinheiten. Die Domäne besteht aus mehreren Objekten, darunter der eigentliche Domain Controller, PCs, Benutzer, Gruppen sowie Server (Abbildung 3). PCs und Benutzer sind hierbei selbsterklärend. Das Server-Objekt beinhaltet beispielsweise den Microsoft SQL Server, Microsoft Exchange Server oder einen Fileserver. In Gruppen können Benutzer, PCs und Server organisiert werden und ihnen über die Gruppenzugehörigkeit Ressourcenberechtigungen vergeben werden.

[6] Enthalten in Microsoft TechNet (2012)

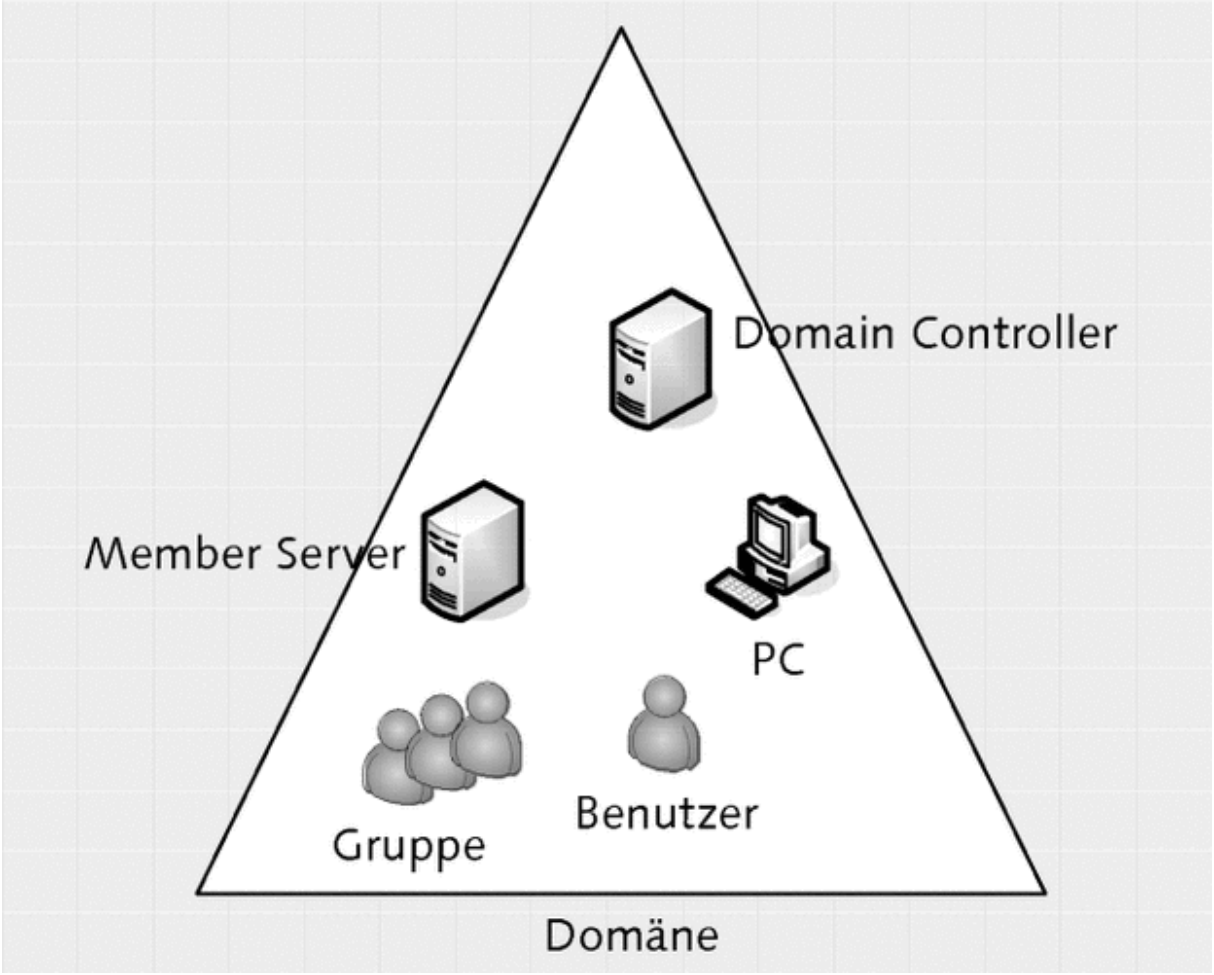

Abbildung 3: *Vorhandene Objekte in einer Domain*[7]

Der Tree stellt einheitliche Namensräume für die Umgebung zur Verfügung. Mehrere Domänen bilden hierbei einen Tree. Ein Tree besitzt den Aufbau einer Wurzel, dieser startet oben mit dem Initialen Namensraum. In Abbildung 4 ist der Aufbau eines solchen Trees und die Verzweigung des Namensraums dargestellt. Anders als der Namensraum werden die Berechtigungen nicht vererbt und müssen, falls benötigt, neu angelegt werden.

In einem Tree befinden sich die Domänen in einem Vertrauensverhältnis. Dies bedeutet, dass, wenn sich ein Benutzer oder Computer bei einer Domäne authentifiziert, die anderen Domänen keine weitere Authentifizierung mehr verlangen, wenn der Anfragende Zugriff auf eine Ressource der anderen Domäne

[7] Enthalten in Boddenberg, U. (2010) Kapitel 8.1.1

anfordert. Dies ist allerdings unabhängig von den Berechtigungen. Diese werden in jedem Fall überprüft, um festzustellen, ob der Anfragende überhaupt das Recht hat, auf die angeforderte Ressource zuzugreifen.

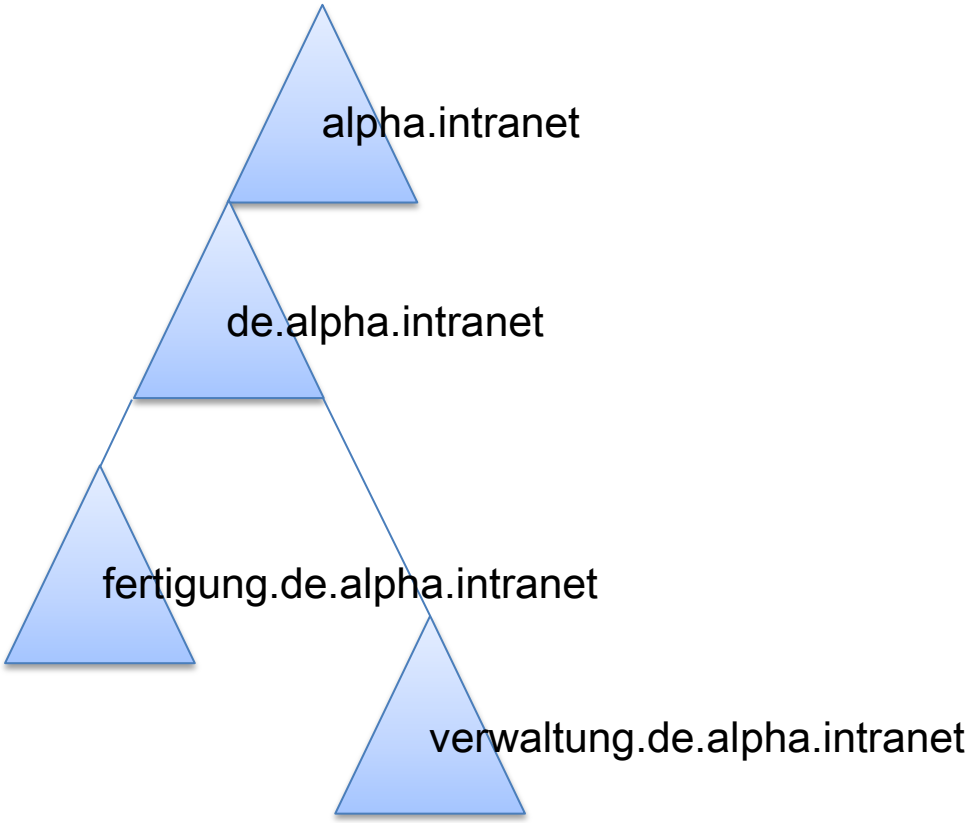

Abbildung 4: *Mehrere Domänen bilden einen Tree*[8]

Unter dem Forest wird die Gesamte Struktur des Verzeichnisses beschrieben. In einem Forest können mehrere Trees hängen, wobei sich die Namensräume der Trees unterscheiden. Eine solche Struktur wird eingesetzt, wenn zum Beispiel die einzelnen Lokationen zentral verwaltet werden sollen, die Richtlinien und Strukturen aber unabhängig voneinander aufgebaut und verwaltet werden.

Mit Organisationseinheiten, sogenannten OUs (Organisational Unit) können die gebildeten Domänen in einer Struktur abgebildet werden. Dadurch kann die Struktur der eigenen Firma auf die Domänen angewendet werden. Dies hilft der

[8] Enthalten in Boddenberg, U. (2010) Kapitel 8.1.1

Übersichtlichkeit und vermindert den Verwaltungsaufwand, indem der Administrator Gruppenrichtlinien auf OUs anwendet, anstatt diese für jeden einzelnen Anwender zu definieren.

Weitere Vorteile sind die Möglichkeiten auf Basis von OUs Anmeldeskripte anzulegen, welche die Benutzer beim Login automatisch mit Druckern oder Netzwerkfreigaben verbinden. Des Weiteren ist es möglich, die Verteilung von neuer Software über die Zugehörigkeit zu OUs zu definieren oder in Anwendungen Funktionen nur für bestimmte Anwender freizuschalten, indem die Software die Zugriffsrechte aus dem Active Directory bezieht.

Ein Beispiel einer solchen Anwendung ist SharePoint 2010. Dies ist ein Portal für die Zusammenarbeit. In ihr kann konfiguriert werden, dass die Berechtigungen im Portal von den Berechtigungen im AD abhängig sind und auch automatisch aktuell gehalten werden beziehungsweise ein regelmäßiger Abgleich stattfindet.

Die Informationen, die aus dem Active Directory über einen Benutzer oder eine Maschine bezogen werden können, sind über ein Schema geregelt. Dieses bildet die Datenbankstruktur des ADs ab und ist über Hilfsmittel erweiterbar. Dadurch können beliebig viele Details zu den Anwenderkonten hinzugefügt werden (Abbildung 5). Es ist allerdings dabei zu beachten, dass das Hinzufügen von Feldern nur mit Einschränkungen wieder rückgängig zu machen ist und dass die Änderungen an den Feldern Auswirkungen auf den gesamten Forest haben.

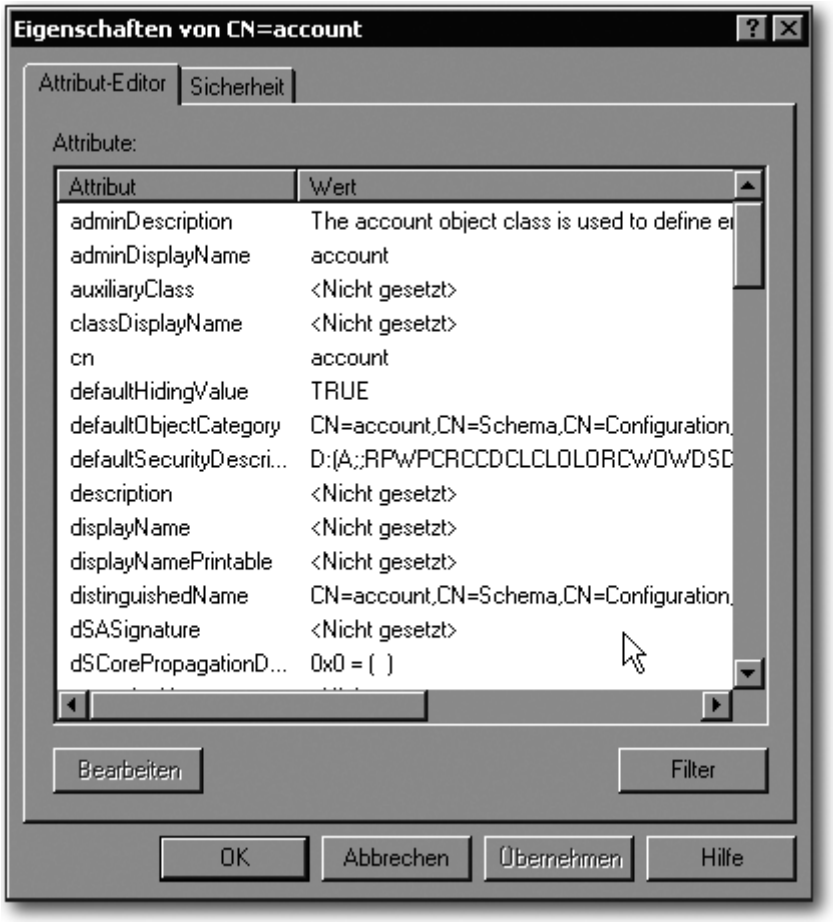

Abbildung 5: *Auflistung von Kontoeigenschaften*[9]

2.3 Model-View-Controller

Die Absicht vieler Systeme ist es, Daten von einem Datenspeicher zu erhalten und diese dem Anwender anzuzeigen. Nachdem der Benutzer die Daten gesichtet und verändert hat, werden diese wieder zurück in den Datenspeicher gesendet und dort gespeichert. Deshalb liegt die Idee nahe, den Aufwand der Programmierung zu minimieren, indem der Datenspeicher und die Benutzeroberfläche zusammen programmiert werden. Das Problem, das sich dabei ergibt, ist, dass sich bei den meisten Anwendungen die Benutzeroberfläche öfters ändert als die der Datenverarbeitung zugrundeliegende Logik.

Gründe dafür können das Erstellen oder Anpassen von Webseiten bei einer Webanwendung sein sowie die Anpassung der Anwendungsoberfläche an verschiedene Geräteklassen, wie zum Beispiel Smartphones oder Tablets.

[9] Enthalten in Boddenberg, U. (2010) Kapitel 8.1.2

Um bei dem Auftreten solcher Änderungen nicht das ganze System verändern zu müssen, wurde das Model-View-Controller (MVC) Modell entwickelt, welches durch Modularisierung versucht die Oberflächenlogik von der Anwendungslogik zu trennen (Abbildung 6).

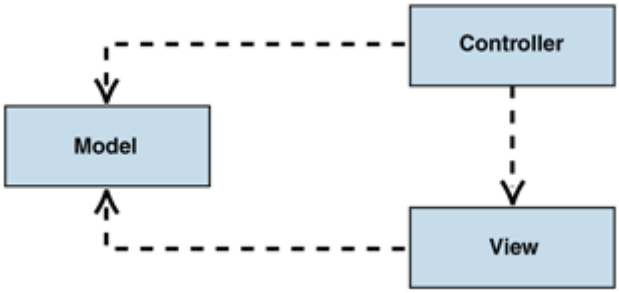

Abbildung 6: *MVC Struktur*[10]

Das Modell beinhaltet die darzustellenden Daten und ist von der Präsentation (View) sowie der Steuerung (Controller) unabhängig. Die Präsentationsschicht und die Steuerungsschicht beobachten das Modell über das Beobachter-Entwurfsmuster (Oberver), um über Aktualisierungen informiert zu werden.

Der Observer ist ein weiteres Entwurfsmuster aus dem Bereich der Softwareentwicklung, welches definierte Schnittstellen zur Verfügung stellt. Durch diese ist es den Beobachtern möglich, sich beim Observer an-, beziehungsweise abzumelden. Dadurch werden diese über Änderungen, der zu beobachtenden Objekte, auf dem Laufenden gehalten.

Der Vorteil dieses Entwurfsmusters ist es, dass eine lose Kopplung zwischen Beobachter und dem zu beobachtenden Objekt besteht, wodurch Abhängigkeiten vermieden werden und keine Informationen über die Struktur des zu beobachtenden Objekts für den Beobachter von Nöten sind. Das zu beobachtende Objekt muss lediglich die Schnittstelle des Observers implementieren.

Die zweite Komponente im MVC Modell ist die Präsentationsschicht (View). Diese ist dafür zuständig die Daten aus dem Modell auf der Oberfläche darzustellen. Des Weiteren ist sie für die Interaktion mit dem Anwender zuständig und leitet dessen

[10] Enthalten in Microsoft Patterns & Practices (2012)

Eingaben an die Steuerung (controller) weiter. Die Präsentationsschicht benötigt zur Anzeige der Daten und Weiterleitung der Anwenderinteraktionen Kenntnisse über das Modell und die Steuerung. Änderungen am Modell bekommt sie durch die Anmeldung beim Observer mit.

Genau wie die Präsentationsschicht nutzt auch die Steuerung den Oberver um über Änderungen am Modell informiert zu werden. Die Steuerung ist zuständig für die Umsetzung der Benutzereingaben. Sie nimmt diese von der Präsentationsschicht entgegen, wertet diese aus und ändert auf Grundlage der hinterlegten Logik die Daten im Modell.[11]

2.4 .NET-Framework

Der folgende Abschnitt über das .NET-Framework sowie die Grundlagen zu C# wurden aus der von Jens Willmer erstellten Praxisarbeit (T300) „Evaluierung der App-Entwicklung für iOS 5“ entnommen.

Das .NET-Framework ist eine in Windows integrierte Komponente, welche das Entwickeln und Betreiben von Programmen und Webservices ermöglicht. Das Framework wurde erstellt, um eine objektorientierte Entwicklung zu ermöglichen und Konflikte bei der Versionierung zu minimieren. Außerdem ist eine sichere Ausführung des Programmcodes sowie das Beheben von Geschwindigkeitsproblemen ein Ziel der Entwicklung. Eine weitere Absicht des Frameworks ist es, die plattformunabhängige Ausführung von Programmen zu ermöglichen, wodurch sich die mit dem .NET-Framework entwickelten Programme auf allen Windowsplattformen ausführen lassen. Durch das Miteinbeziehen industrieller Kommunikationsstandards bei der Entwicklung des Frameworks wurde sichergestellt, dass die Integration von Fremdcode, bei der Programmentwicklung, keine Probleme verursacht.[12]

Das .NET-Framework besteht aus vier separaten Produktgruppen. Die erste Gruppe umfasst Entwicklungstools und Bibliotheken. Die Bibliotheken bestehen aus objektorientierten Sammlungen von wiederverwendbarem Quellcode. Sie beinhalten Quellcode für die grafische Entwicklung von Anwendungen, für kommandozeilenbasierte Programmierung und die Programmierung für Internetseiten auf Basis von ASP.NET. ASP.NET bietet zur Entwicklung WebForms

[11] Vgl. Microsoft Patterns & Practices (2012)
[12] Vgl. Microsoft MSDN (2012)

sowie XML-Webservices. Das .NET-Framework betreibt hierfür eine skalierbare Laufzeitumgebung.[13]

Des Weiteren befinden sich in dieser Gruppe Schnittstellen für Windowsdienste, Webservices, Windowsanwendungen, sowie Tools wie die Entwicklungsumgebung Visual Studio. Weiter sind die Sprachen C#, J# und VB.NET darin enthalten, genauso wie die Common Language Runtime (CLR). [14]

Die CLR ist für die Verwaltung von Quellcode während der Ausführung zuständig. Darunter zählt das Verwalten von Threads und Arbeitsspeicher genauso wie das Überwachen von Typsicherheiten und das akkurate Erstellen des Quellcodes, um Stabilität und Sicherheit zu gewährleisten. Das CLR gewährleistet, dass alle Programme, welche in einer .NET-kompatiblen Sprache geschrieben wurden, auf dem Computer ausgeführt werden können. Hierzu wird der Quellcode in ein assemblerähnliches Format konvertiert, woraufhin der Code auf Sicherheit und Stabilität getestet wird. Der Assemblercode wird beim Starten des Programms zur Laufzeit "just in time" (JIT) kompiliert. Hierbei werden die jeweiligen Anforderungen des Hostsystems beachtet, wodurch das Programm auf jeder Plattform, die .NET unterstütz, lauffähig ist.[15]

In der zweiten Gruppe befinden sich kommerzielle Webservices zur Unterstützung zwischen verschiedenen Anwendungsprogrammen. In der dritten Produktgruppe sind spezialisierte Server, wie der Microsoft SQL Server oder Microsoft Exchange Server, enthalten. Diese stellen spezielle Funktionalitäten für die relationale Datenspeicherung zur Verfügung. In der letzten Gruppe befinden sich neue Geräte wie Mobiltelefone und Spielkonsolen.[16]

Zur Abfrage beliebiger Datenquellen stellt das .NET-Framework Language Integrated Query (LINQ) zur Verfügung. Dabei handelt es sich um eine Komponente, mit der Abfragen auf XML-Dateien oder Datenbanken direkt in der .Net-Programmiersprache erstellt werden können. Die Anfragen ähneln hierbei SQL-Abfragen und haben den Vorteil, dass durch die Integration in .NET die Abfragen durch den Compiler auf Fehler geprüft werden können, bevor sie abgesetzt werden.

[13] Vgl. Thai, T. L./Lam H. (2003) S. 2ff
[14] Vgl. Microsoft MSDN (2012)
[15] Vgl. Dotnet-Guide.com (2012)
[16] Vgl. Thai, T. L./Lam H. (2003) S. 2ff

Programme, welche auf .NET aufbauen, können grundsätzlich nicht in einer Mac OS-Umgebung betrieben werden, da hier das Framework fehlt.

2.4.1 C#

Bei C# handelt es sich um eine strukturierte, komponentenbasierte, sowie objektorientierte Programmiersprache. Sie besitzt circa 80 Schlüsselwörter und ist dennoch sehr ausdrucksfähig bei der Implementierung moderner Programmierkonzepte. Für die Programmierung von C# werden keine Header-Dateien oder Interface-Definitionen, wie von C++ bekannt, benötigt. Eine in C# geschriebene Klasse kann nur von einer anderen Klasse abgeleitet werden (keine Mehrfachvererbung), kann dafür jedoch mehrere Schnittstellen implementieren.

Des Weiteren besitzt C# Structs. Structs sind in C# ein leichtgewichtiger Datentyp, der nicht erben oder geerbt werden kann. Structs können jedoch Schnittstellen implementieren und benötigen nur geringe Systemressourcen.

Darüber hinaus unterstützt C# Delegates. Delegates sind Referenztypen, welche Methoden mit ihren Signaturen und Rückgabetypen kapseln. Mit ihnen lassen sich Methoden über Verweise aufrufen.

Auch unsicherer Code lässt sich mit C# schreiben. Dabei handelt es sich um Quellcode, welcher nicht vom CLR überprüft werden kann. Dadurch werden mögliche Fehler vor der Ausführung eventuell nicht entdeckt. In einem unsicheren Codesegment können zum Beispiel Zeiger auf den Arbeitsspeicher gesetzt werden. Bei unsicherem Code ist der Entwickler dafür zuständig, dass der Code keine Sicherheitsrisiken oder Zeigerfehler beinhaltet.

Schlussendlich wird der in C# geschrieben Quellcode zu einem Assembly kompiliert, welches eine Sammlung von Dateien darstellt und im Betriebssystem als Dynamic Link Library (DLL) oder Executable (EXE) angezeigt wird. Diese stellen die Grundeinheit für die Wiederverwendung, Versionierung, Sicherheit sowie Verbreitung dar.[17]

2.4.2 ASP.NET MVC

Mit ASP.NET MVC hat Microsoft eine neue Möglichkeit geschaffen Webapplikationen zu programmieren. Vor ASP.NET MVC wurde ASP.NET Web Forms entwickelt,

[17] Vgl. Liberty, J. (2005) S.7f

welches den Ansatz verfolgt, bekannte Bausteine aus der Softwareentwicklung in der Webentwicklung verfügbar zu machen.

Dieser Ansatz sorgt allerdings mit der Weiterentwicklung des Webs zunehmend für Probleme. Einerseits wurde bei der Entwicklung nicht beachtet, dass automatische Softwaretests heutzutage ein essenzieller Teil der Programmierung sind. ASP.NET Web Forms bietet nur sehr eingeschränkte Möglichkeiten automatische Tests durchzuführen. Andererseits versteckt ASP.NET Web Forms HTML und HTTP so gut es geht, wodurch das Implementieren von speziellen Funktionen den Entwickler dazu zwingt, Postback-Funktionen nachzukonstruieren und damit die Abstraktionsschichten von ASP.NET Web Forms zu verlassen.

ASP.NET MVC ist der Ansatz von Microsoft die Probleme durch den Einsatz des MVC Musters zu lösen. Die Basis von ASP.NET MVC ist genau wie bei ASP.NET Web Forms, ASP.NET. ASP.NET ist eine Technologie, die auf das .NET-Framework aufbaut und Klassen zur Entwicklung von Webanwendungen bereitstellt (Abbildung 7).[18] [19]

[18] Vgl. Esposito, D (2009)
[19] Vgl. hierzu auch Wenz C. u. a. (2004) S.19ff

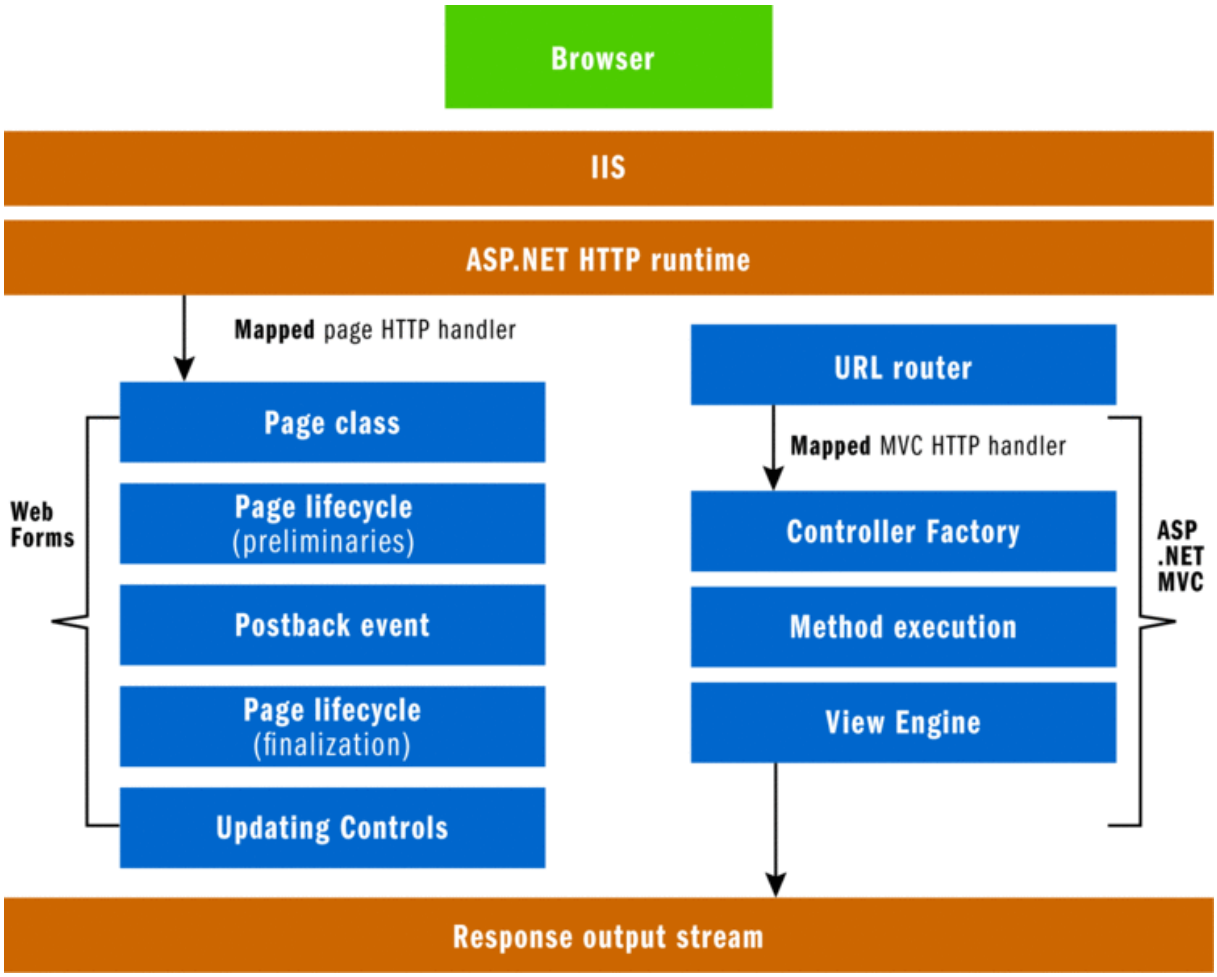

Abbildung 7: *Vergleich von Web Forms und MVC zur Laufzeit* [20]

Durch das Verwenden des .NET-Frameworks besitzt ASP.NET MVC auch dessen Vorteile. Es bietet vereinheitliche Operationen, welche beim Wechsel auf eine andere, von .NET unterstützte Sprache, weiter benutzt werden können. Genauso wie eine verbesserte Geschwindigkeit gegenüber interpretierten Sprachen wie PHP.

Des Weiteren muss sich der Entwickler nicht um die Speicherverwaltung kümmern, um diese kümmert sich das System und sorgt automatisch dafür, dass nichtmehr benötigte Objekte aus dem Arbeitsspeicher entfernt werden.

Ein weiterer Vorteil, den das .NET-Framework bietet, ist die Versionssicherheit, wodurch Anwendungen, die für unterschiedliche Framework-Versionen geschrieben wurden, parallel auf einem Server betrieben werden können ohne Probleme zu verursachen.

[20] Enthalten in Esposito, D (2009)

2.5 Visual Studio 2010

Visual Studio 2010 (VS) ist eine integrierte Entwicklungsumgebung (Integrated Development Environment, IDE) von Microsoft zur Entwicklung von Applikationen auf Basis des .NET-Frameworks. Ohne eine IDE müsste ein Texteditor zum Schreiben des Programmcodes geöffnet werden, anschließend eine Kommandozeile aufgerufen werden, um daraufhin einen Compiler zu starten, um das geschriebene Programm zu kompilieren. VS hilft dem Entwickler durch einen strukturierten Aufbau, umfangreiche Tools und eine weite Automatisierung von Entwicklungsabläufen, die Produktivität zu steigern.

Eine weitere Funktion von VS ist, dass es beim Anlegen eines neuen Projekts ein Template erstellt, welches Standardquellcode bereithält, womit das Skelett der Anwendung bereits geschrieben ist. Dieser Quelltext kann sofort kompiliert und ausgeführt werden, sodass sofort mit der Programmierung der Logik begonnen werden kann.

Der geschriebene Quellcode wird farblich, in Abhängigkeit des Typs, hervorgehoben, um dem Entwickler eine bessere Übersicht zu gewährleisten. Des Weiteren werden Funktionsvorschläge bei Eintippen von Funktionen angezeigt. Dadurch reicht es aus, die Anfangsbuchstaben einer Funktion zu schreiben und diese daraufhin aus den Vorschlägen auszuwählen, was das versehentliche Vertippen minimiert.

Späteres Ändern einer Variablen oder Funktion kann durch eine eingebaute Automatik auf alle Vorkommnisse dieser angewendet werden. Dadurch reicht es aus, die Variable oder Funktion nur an der Stelle, an welcher sie definiert wurde, zu ändern, um die Änderung auf das gesamte Projekt anzuwenden.

Ein weiteres Tool, welches VS bietet, ist die Toolbox. Sie beinhaltet häufig benötigte und ausführlich getestete Controls, wodurch der Programmierer diese nicht selbst entwickeln muss und seine Produktivität dadurch gesteigert wird.

Um auf die individuellen Bedürfnisse eines jeden Entwicklers eingehen zu können, bietet VS eine große Auswahl an Anpassungsmöglichkeiten. Der Entwickler kann nicht benötigte Tools ausblenden, Makros schreiben, die ihm Arbeit abnehmen, welche noch nicht durch VS automatisiert wurde, die verschiedenen Fenster neu

anordnen oder über einen Plug-In-Manager Erweiterungen für VS herunterladen, um neue Funktionsmöglichkeiten zu erhalten. [21]

3. Vorgehen

In diesem Kapitel wird, mit Bezug auf den Grundlagenteil, das Vorgehen bei der Entwicklung der Oberfläche sowie dem Verbindungsaufbau zum Active Directory und das Auslesen dessen beschrieben.

3.1 Erstellen eines Designkonzeptes

Bevor mit der Arbeit begonnen werden kann wird ein Designkonzept benötigt. Für die Anfertigung des Konzepts wurde Microsoft Expression Blend 4 ausgewählt, ein Bestandteil der Produktfamilie Microsoft Expression Studio, welches über die Universitätslizenz bezogen wurde. Expression Blend ist ein Programm, das Microsoft für die Entwicklung von Oberflächen auf Basis von .NET entwickelt hat. Mit ihm ist es möglich, in kurzer Zeit umfassende Oberflächen für Web und Desktopanwendungen zu gestalten.

Die mit Expression Blend erstellten Oberflächen lassen sich anschließend in Visual Studio importieren, wodurch eine Trennung zwischen Oberflächenerstellung und Logikprogrammierung erreicht wird. Die in diesem Abschnitt gezeigten Designs wurden hierbei mit der Funktion SketchFlow erstellt, mit welcher rudimentäre Layouts entwickelt werden können, die noch keine Interaktionslogik enthalten.

Für eine produktive Nutzung von Expression Blend ist eine hohe Einarbeitungszeit erforderlich, da der Funktionsumfang ähnlich umfangreich wie bei Adobe Photoshop ist. Deshalb wurde auf die Erstellung des kompletten Designs mit Hilfe von Expression Blend verzichtet und das Programm lediglich zur Erstellung von Entwürfen verwendet. Für die anfängliche Entwicklung wird deshalb auf die Standard-Weboberflache des MVC-Templates zurückgegriffen, welches am Ende über CSS an das Design des DHBW-Intranets angepasst wird.

Für den Aufbau des Dashboards hat sich der Student eine Seite mit zwei Reitern überlegt. Der erste Reiter (Abbildung 8) dient der Anzeige der wichtigen Informationen wie Mailquota und Filequota und der zweite Reiter der Suche nach E-Mail-Adressen von bekannten Personen (Abbildung 9).

[21] Vgl. Mayo, J. (2010) S.4f

Das Mailquota sowie das Filequota werden im Dashboard-Entwurf über Kuchendiagramme visuell dargestellt. Die eigentliche Größe des jeweiligen Speichers steht hierbei in der Mitte des Kreises. Die zwei Kuchenstücke, die das Verhältnis zwischen benutztem und unbenutztem Speicher darstellen, werden über Striche mit den Detailinformationen verbunden. Hierbei ist zu überlegen, ob es noch weitere sinnvolle Dateiinformationen anzuzeigen gibt. Möglicherweise könnte bei der Anzeige des Mailquotas noch die Gesamtmenge an E-Mails, ein Liste der ungelesenen sowie der größten E-Mails angezeigt werden.

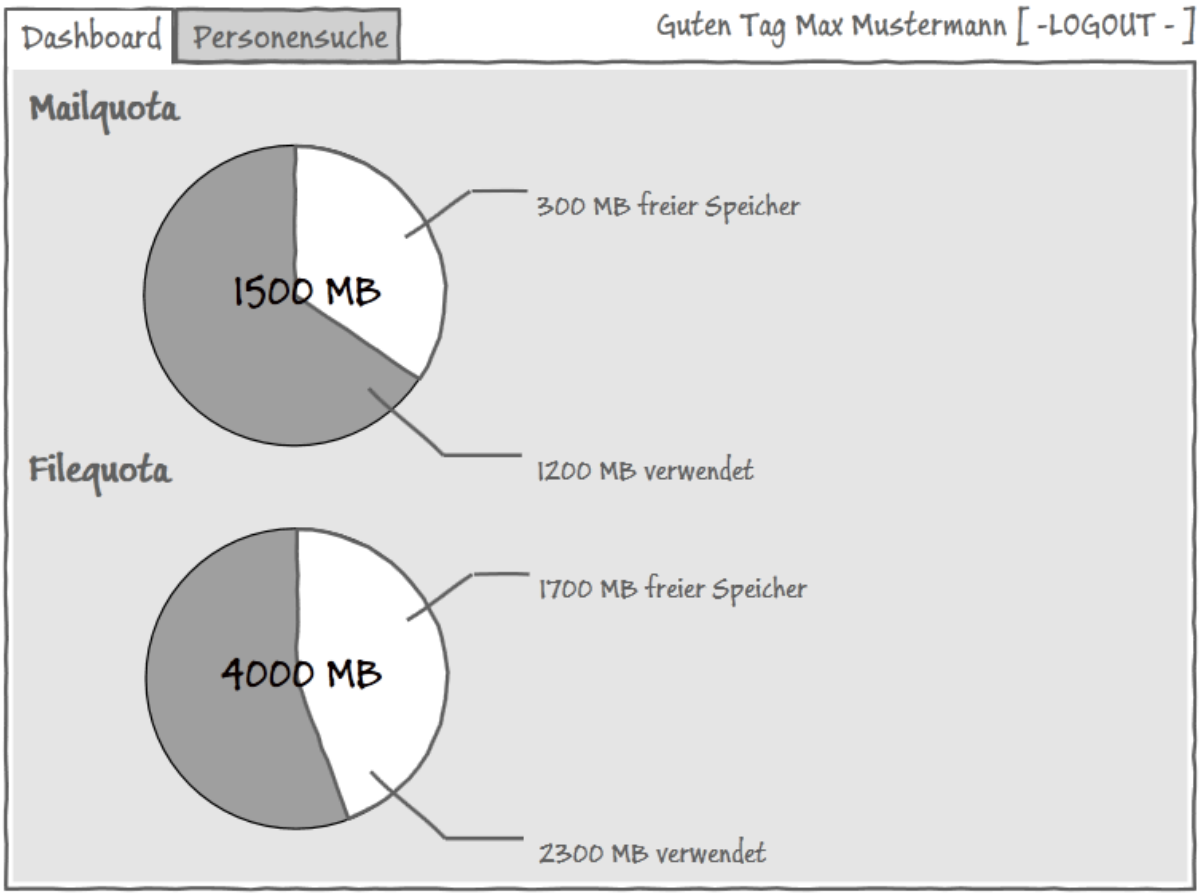

Abbildung 8: *Designkonzept für das Dashboard* [22]

Der Tab für die Personensuche ist klassisch aufgebaut. Die Eingabefelder für die Suche sind ganz oben angeordnet, darunter erstreckt sich die Ergebnisliste mit den gefundenen Personen und deren Mail-Adressen. Unter dieser Liste verbirgt sich der

[22] Entwickelt mit Microsoft Expression Blend 4

Startknopf für die Suche. Darauf zu achten ist, dass die Ergebnisliste klein genug bleibt, um auch auf kleinen Bildschirmen ohne Verschieben an den Startknopf zu gelangen. Dies sollte jedoch gegeben sein, da in der Problemstellung die maximale Anzahl an Suchergebnissen bereits beschränkt wurde.

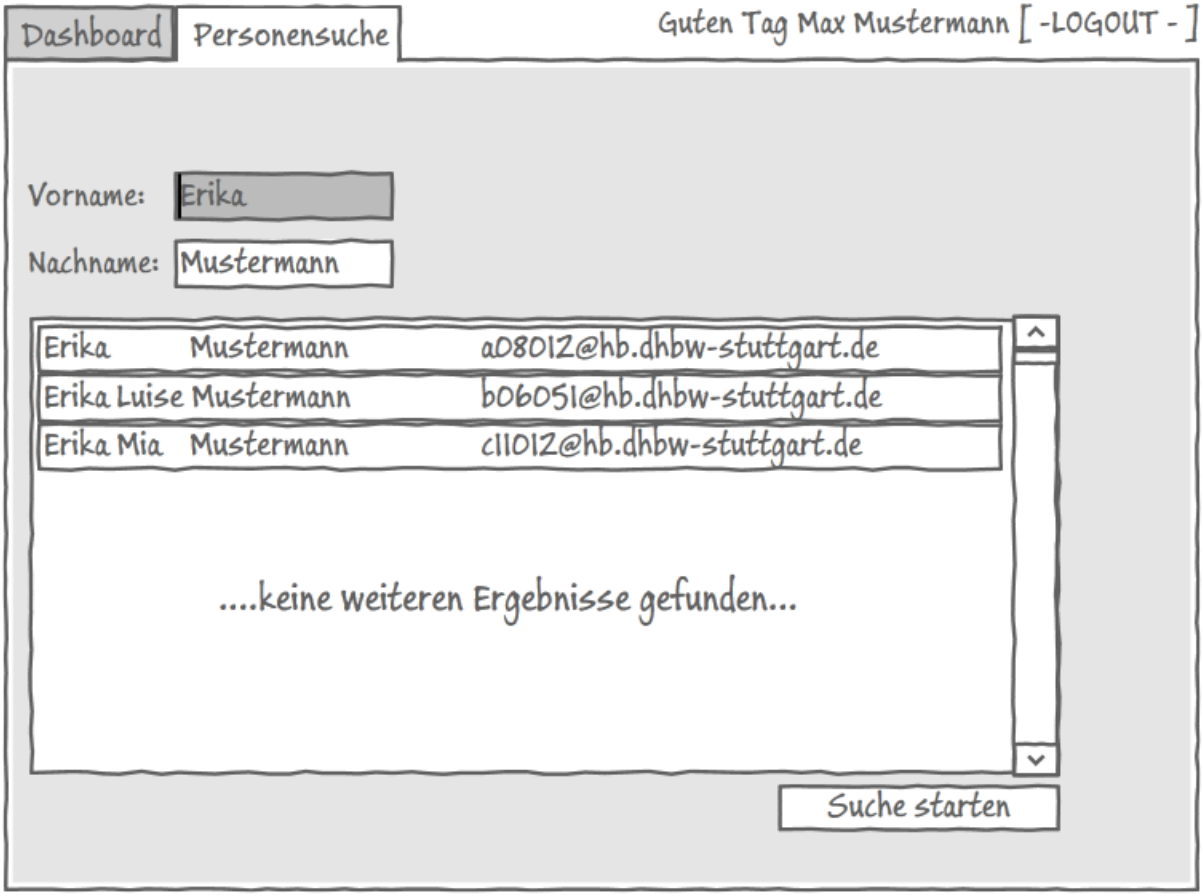

Abbildung 9: *Designkonzept für die Personensuche* [23]

Am oberen rechten Rand wird der angemeldete Benutzer begrüßt und findet auch die Option sich abzumelden. Die Anmeldung könnte ebenfalls über den oberen rechten Bereich geschehen, es ist allerdings zu überlegen, ob es sinnvoll ist, eine eigene Seite für die Anmeldung zu erstellen, welche das Anmeldefenster, zentral, in der Mitte positioniert und bei Erfolg auf die eben vorgestellten Seiten weiterleitet.

[23] Entwickelt mit Microsoft Expression Blend 4

3.2 Erstellen der Projektstruktur

Zum Erstellen der Projektstruktur wurde als Erstes VisualStudio 2010 und das ASP.NET MVC 3 Framework über den von Microsoft bereitgestellten Webinstaller (http://asp.net/mvc) installiert.

Über die Menüleiste wurde daraufhin ein neues ASP.NET MVC 3 Projekt ausgewählt (Abbildung 10) und über den daraufhin erscheinenden Wizard angegeben, dass es sich bei dem neuen Projekt um eine Internetanwendung handeln soll. Dadurch wurde automatisch die Grundstruktur für die formularbasierte Autorisierung erstellt.

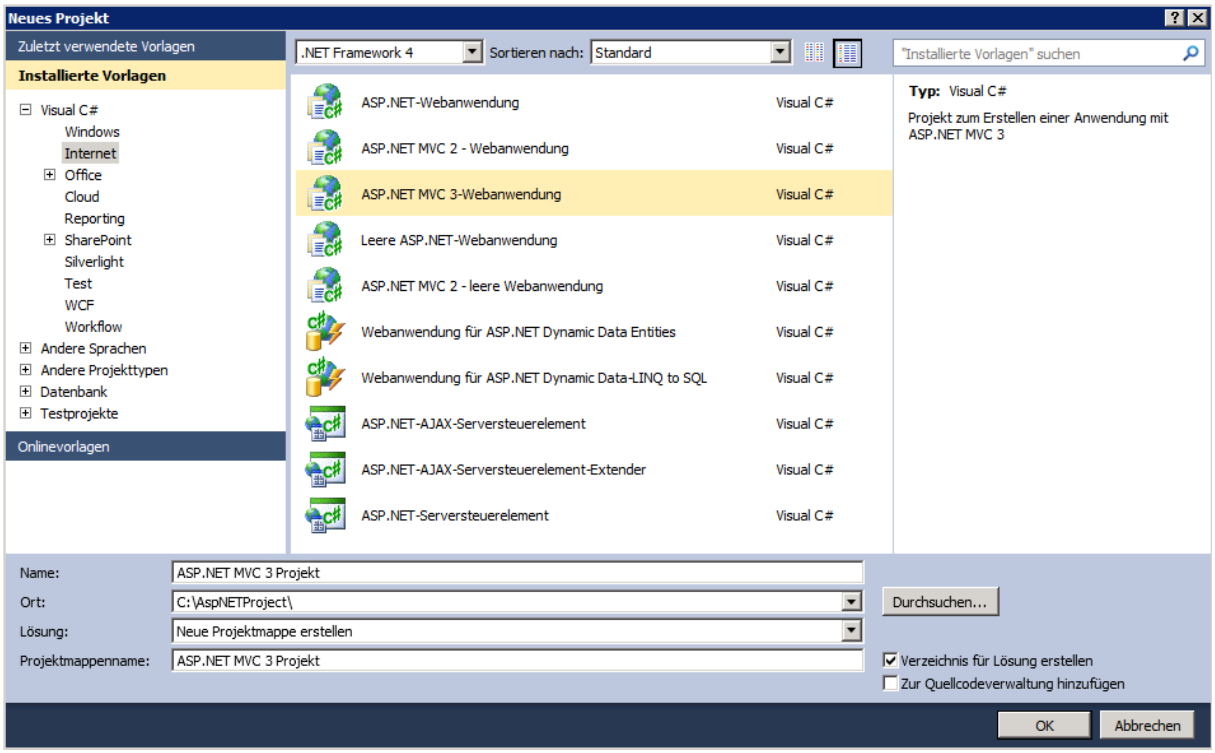

Abbildung 10: *Auswahlmenü für Projekterstellung*

Als Ansichtsmodul wurde Razor ausgewählt. Razor ist ein Ansichtsgenerator (View Engine), welcher auf die Syntax von C# zurückgreift. Dadurch fällt es dem C#-Entwickler leicht, die Scriptsprache zu erlernen. Mit ihr ist es möglich, HTML-Seiten zu erstellen, welche mit serverseitigem Quellcode versehen werden können. Dieser wird von Razor erkannt und ausgeführt.

Nach Fertigstellung des Wizards ist im Projektmappen-Explorer die Aufteilung des Projekts zu sehen (Abbildung 11). Es gibt Ordner für die Models, die Views sowie die

Controller. Des Weiteren gibt es bereits ein Verzeichnis für Skripte, in dem sich einige JavaScript-Dateien befinden. Diese übernehmen clientseitige Validierungsfunktionen. In dem Ordner Content befinden sich die Vorlagen für das Aussehen. Darunter sind CSS-Dateien sowie Bilder.

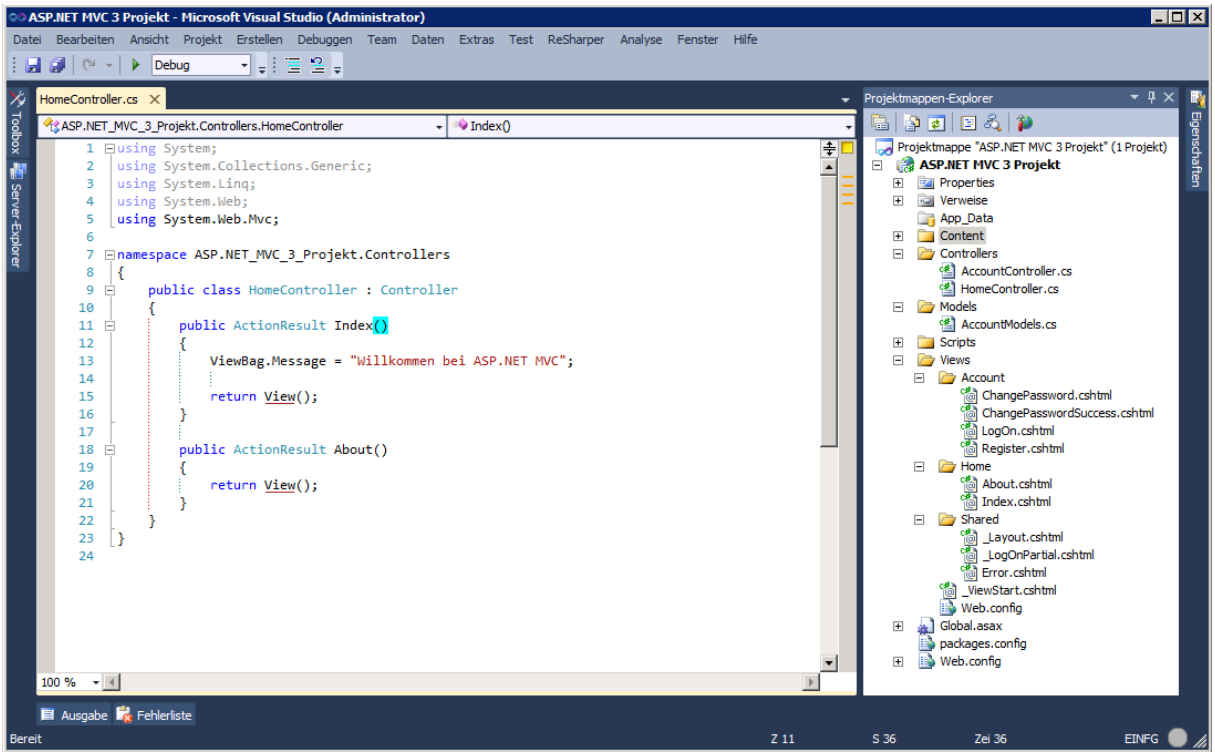

Abbildung 11: *Projektstruktur*

Auf Abbildung 11 ist zu erkennen, dass es bereits ein Model im Ordner Models gibt. Zudem sind schon einige Ansichten angelegt und es existieren zwei Controller. Diese wurden automatisch beim Anlegen des Projekts erstellt, wodurch der Entwickler zu diesem Zeitpunkt in der Lage ist, das Projekt zu kompilieren und die Homepage zu betrachten (Abbildung 12).

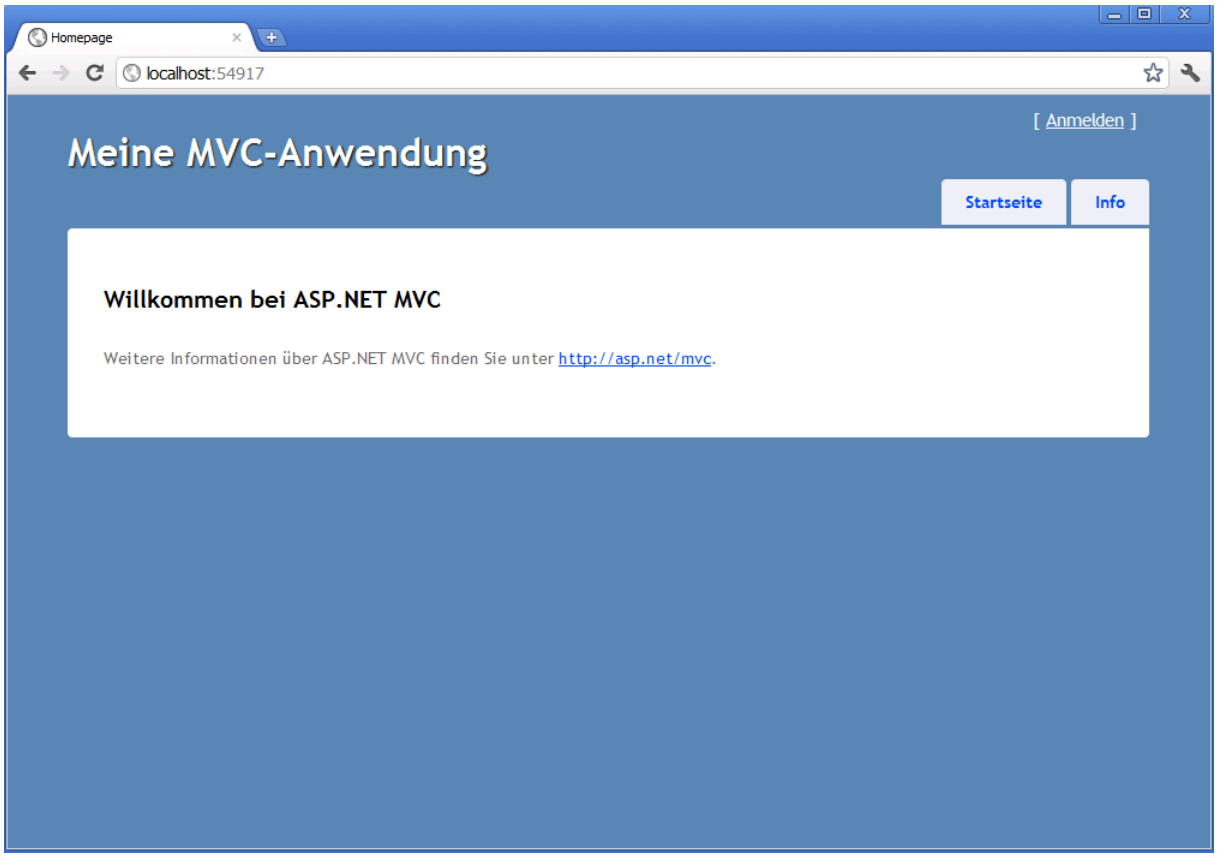

Abbildung 12: *Startseite der Homepage*

Die generierte Ansicht ist ein Resultat des Home-Controllers. Diese besitzt zwei Reiter mit statischem Text als Inhalt. Über den Punkt Anmelden landet der Betrachter auf dem in Abbildung 13 dargestellten Dialog, welcher zu einer Anmeldung oder Registrierung auffordert. Die Logik, die dahinter steht, wird hierbei vom Account-Controller und dem im Ordner Model angelegten Account-Model bereitgestellt.

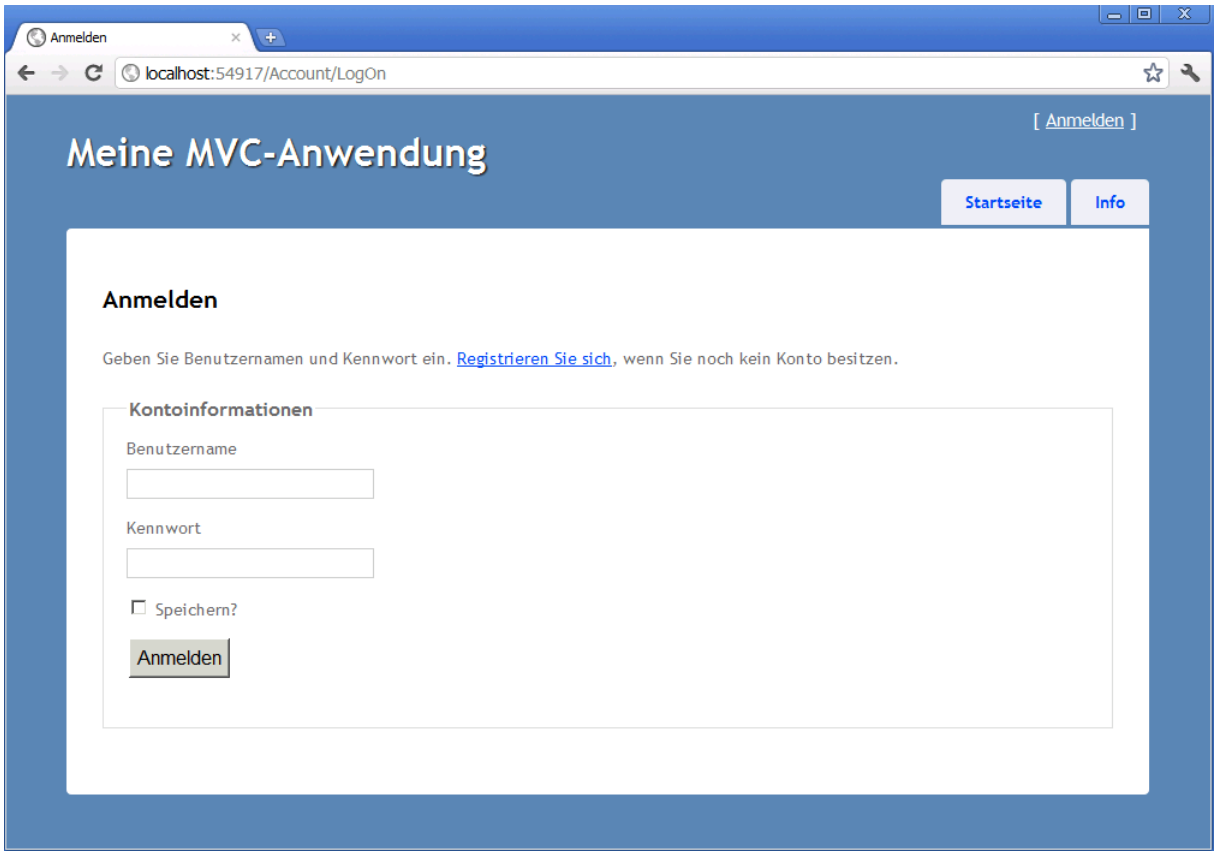

Abbildung 13: *Anmeldeformular der Homepage*

Die Funktion zur Registrierung neuer Anwender wird für dieses Projekt nicht benötigt, die Anwenderinformationen werden aus dem ActiveDirectory bezogen und die Anwendung soll keine Möglichkeit bieten, weitere Anwender im ActiveDirectory anzulegen. Deshalb wurde diese entfernt.

3.3 Einrichten einer sicheren Verbindung

Nach Anpassung der Projektstruktur an die Anforderungen der anstehenden Anwendungsentwicklung wurde die Anmeldefunktion durch eine verschlüsselte Verbindung vor unberechtigtem mitlesen geschützt. Dies geschah mittels Secure Sockets Layer (SSL).

SSL ist ein kryptografisches Protokoll, welches eine sichere Verbindung über das Internet gewährleistet. Um SSL einsetzen zu können, wird ein Zertifikat benötigt. Dieses kann entweder selbst erzeugt werden oder von ausgewiesenen Zertifikatsstellen aus dem Internet angefordert werden. Diese verlangen allerdings eine jährliche anfallende Gebühr.

Der Vorteil eines Zertifikats aus einer autorisierten Zertifikatsstelle liegt allerdings darin, dass der Browser keine Warnmeldung anzeigt, wenn ein Anwender die entwickelte Seite besucht. Diese Warnmeldung klärt den Besucher darüber auf, dass das eingesetzte Zertifikat keine Vertrauensstellung im Internet geniest und der Anwender selbst die Rechtmäßigkeit des Zertifikats bewerten muss. Akzeptiert er dies, wird er auf die eigentliche Seite weitergeleitet. Die Warnung über das selbst erstellte Zertifikat des Herausgebers ist allerdings weiterhin über eine rot eingefärbte Adressleiste sichtbar (Abbildung 14).

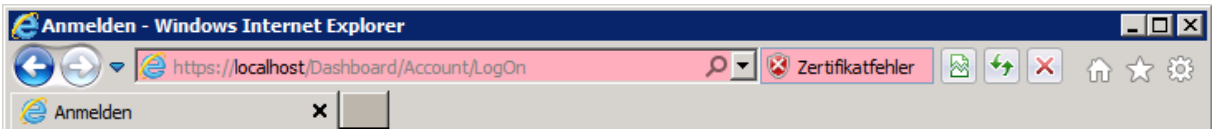

Abbildung 14: *SSL Zertifikatsfehler*

Für die Entwicklungsphase wurde ein selbst ausgestelltes Zertifikat verwendet. Dies kann bei der Produktivnahme gegen ein bestehendes Zertifikat der Universität ausgetauscht werden.

Ein eigenes Zertifikat ist über den IIS erstellbar (Abbildung 15). Bei der Erstellung müssen die Funktionen des Zertifikates angegeben werden und die Anwendung, für die es ausgestellt werden soll.

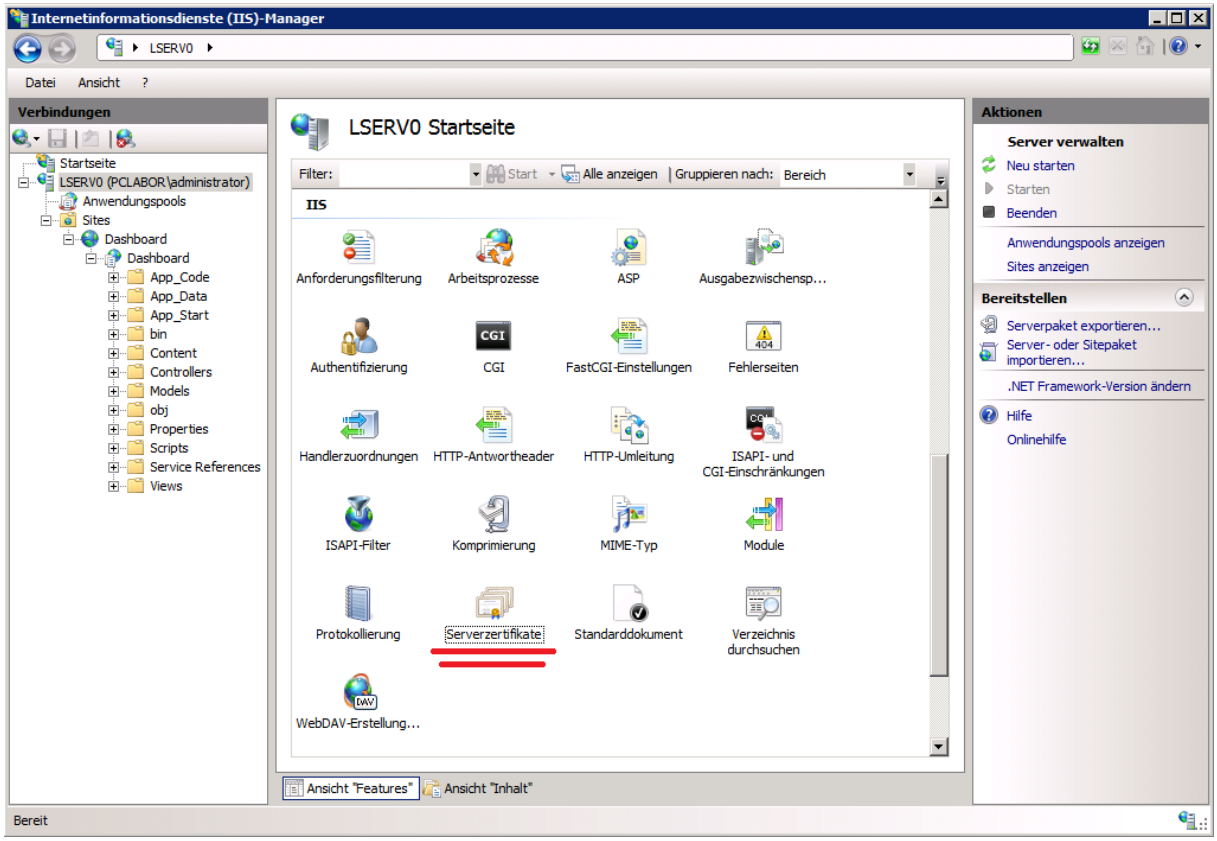

Abbildung 15: *Zertifikatseinstellungen im IIS*

In Abbildung 16 ist das Zertifikat für die zu entwickelnde Seite zu sehen. Dieses besitzt mehr Rechte als benötigt. Dies ist für die Entwicklung zu vernachlässigen. Nach Erstellung des Zertifikats muss in den Einstellungen des IIS der Port 443 für die Anwendung freigegeben werden. Über diesen Port laufen standardmäßig die verschlüsselten Verbindungen.[24]

[24] Vgl. ScottGu (2007)

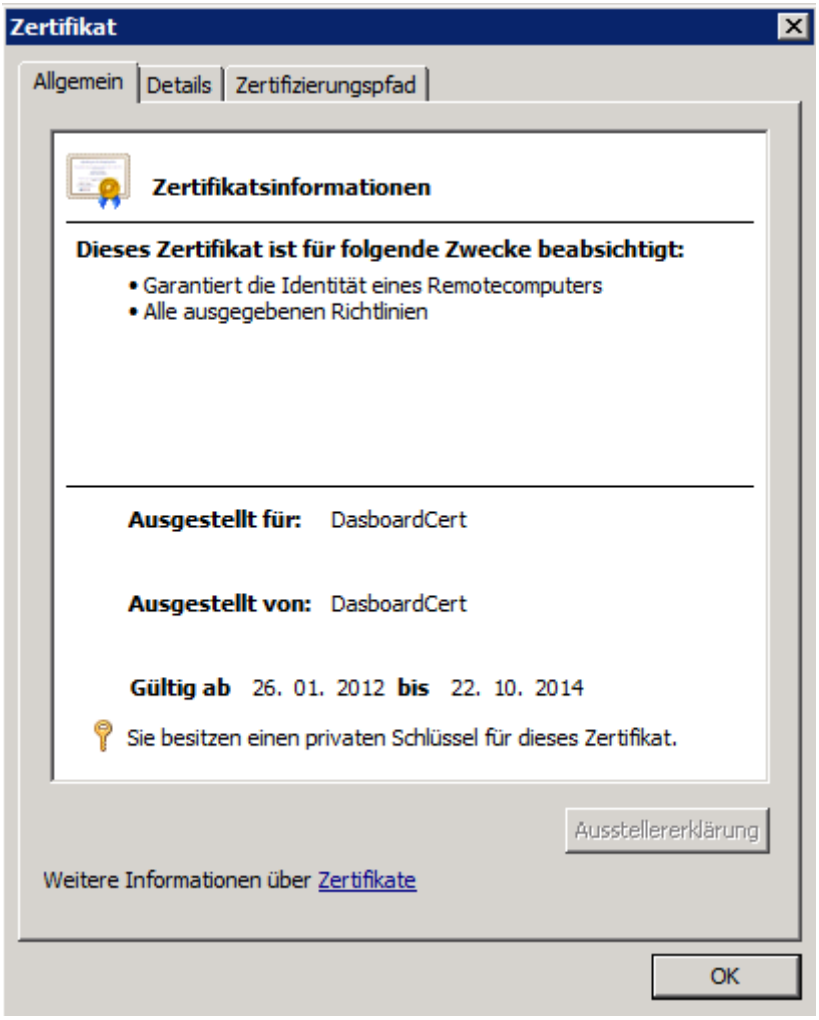

Abbildung 16: *Ausgestelltes Zertifikat*

Als letzter Schritt zum Einrichten einer sicheren Verbindung wird im Projektverzeichnis auf oberster Ebene die Datei Web.config geöffnet. Hier werden die in Abbildung 17 gezeigten Abschnitte hinzugefügt. Der in Zeile 21 startende Abschnitt legt die formularbasierte Authentifizierung fest und bewerkstelligt eine Umleitung auf die Anmeldeseite, wenn der Besucher nicht angemeldet ist. Des Weiteren wurde durch das Setzen des Attributes requireSSL dafür gesorgt, dass zur Authentifizierung das SSL-Protokoll verwendet wird.

In Zeile 27 wird dafür gesorgt, dass alle Verzeichnisse für nicht autorisierte Benutzer gesperrt sind und in der darauffolgenden Zeile werden die Zugriffsrechte für autorisierte Benutzer gesetzt. Diesen ist es erlaubt auf alle Verzeichnisse zuzugreifen.

Durch das komplette Sperren des Zugriffs auf Ordner im Projekt, für Gäste, wird allerdings auch verhindert, dass die Gäste auf die Style-Informationen und Skripte zur Validierung zugreifen können. Dadurch sehen diese nur eine unformatierte Seite, wenn sie die Homepage besuchen. Um dies zu vermeiden, wird von Zeile 5 bis Zeile 18 den Gästen explizit der Zugang zu diesen Ressourcen gewährt.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <configuration>
3    ...
4    .
5    <location path="Content">
6      <system.web>
7        <authorization>
8          <allow users="*" />
9        </authorization>
10     </system.web>
11   </location>
12   <location path="Scripts">
13     <system.web>
14       <authorization>
15         <allow users="*" />
16       </authorization>
17     </system.web>
18   </location>
19   ...
20   .
21   <system.web>
22     <authentication mode="Forms">
23       <!--requireSSL!-->
24       <forms loginUrl="~/Account/LogOn" name=".ADAuthCookie" requireSSL="true" />
25     </authentication>
26     <authorization>
27       <deny users="?" />
28       <allow users="*" />
29     </authorization>
30   </system.web>
31   ...
32   .
33 </configuration>
34
```

Abbildung 17: *Änderungen in der Web.config*

3.4 Autorisation mit dem ActiveDirectory

Für die Authentifizierung des Anwenders wird eine Verbindung mit dem ActiveDirectory benötigt. Durch diese Verbindung soll anschließend überprüft werden, ob der Username existiert und ob das Passwort korrekt ist. Diese

Verbindung wird, wie die Einstellung der SSL-Verbindung, durch die Datei Web.config konfiguriert. In Abbildung 18 ist in Zeile 6 der Verbindungsstring zu sehen. Über die hier definierte Adresse wird später versucht eine Verbindung herzustellen.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <configuration>
3    ...
4    .
5    <connectionStrings>
6      <add name="ADConnectionString" connectionString="LDAP://lserv1.ba-horb.de/OU=BA Horb,DC=ba-horb,DC=de" />
7    </connectionStrings>
8    ...
9    .
10   <system.web>
11     <membership defaultProvider="ActiveDirectoryMembershipProvider">
12       <providers>
13         <clear />
14         <add
15       name="ActiveDirectoryMembershipProvider"
16       type="System.Web.Security.ActiveDirectoryMembershipProvider, System.Web, Version=4.0.0.0, Culture=neutral,
17             PublicKeyToken=b03f5f7f11d50a3a"
18       connectionStringName="ADConnectionString"
19       attributeMapUsername="sAMAccountName" />
20       </providers>
21     </membership>
22   </system.web>
23   ...
24   .
25 </configuration>
```

Abbildung 18: *ActiveDirectory Verbindung in der Web.config*

Ab Zeile 11 wird hierzu ein MembershipProvider festgelegt. Dies ist eine Schnittstelle, auf die aus den Projektklassen zugegriffen werden kann. Der in Zeile 6 geschriebene String wird über seinen Namen auf den in Zeile 14 definierten MembershipProvider referenziert. Zu beachten ist, dass durch das Attribut attributeMapUsername der Benutzername, welchen der MembershipProvider liefert, auf den sAMAccountName verlinkt wird.

Über den folgenden Quellcode wird schließlich der Anwender gegen das ActiveDirectory validiert.

```
bool validUser = Membership.ValidateUser(model.UserName, model.Password);
```

3.5 Auslesen des ActiveDirectory

Nachdem der Anwender erfolgreich validiert wurde, wird eine weitere ActiveDirectory-Abfrage gestartet, welche sich nicht auf das Validieren des Anwendernamens beschränkt, sondern den Zweck verfolgt, das UserModel mit Inhalt

zu füllen. Dieses wurde angelegt, um Datenbankzugriffe zu dem entfernten ActiveDirectory-Server zu minimieren, alle erforderlichen Informationen auf einmal aus der Datenbank auszulesen und in einer lokalen SQL-Datenbank vorzuhalten (Abbildung 19).

```
35      // get user data from active directory
36      DirectoryEntry root = new DirectoryEntry("LDAP://lserv1.ba-horb.de/OU=BA Horb,DC=ba-horb,DC=de");
37      DirectorySearcher dSearch = new DirectorySearcher(root,
38                                                        string.Format("sAMAccountName={0}",
39                                                        model.UserName));
40      PropertyCollection collection = dSearch.FindOne().GetDirectoryEntry().Properties;
41
42      // create UserDataModel
43      UserDataModel dataModel = new UserDataModel();
44
45      // fill dataModel
46      try
47      {
48          dataModel.Guid = Guid.NewGuid();
49          dataModel.FirstName = collection["givenName"].Value.ToString();
50          dataModel.LastName = collection["SN"].Value.ToString();
51
52          //...
53          //.
54      }
55      catch (Exception)
56      {
57          // handle errors..
58      }
```

Abbildung 19: *Auslesen des ActiveDirectorys*

Um die Applikation mit der SQL-Datenbank zu verknüpfen, wurde das Entity Framework heruntergeladen und installiert. Dieses erlaubt dem Entwickler die Daten über ein Objektmodell anstelle eines relationalen Modells zu entwerfen und zu bearbeiten (Abbildung 20).[25]

[25] Mehr dazu unter Microsoft MSDN (2012)

```
9   // derive from System.Data.Entity
10  public class EntityDB : DbContext
11  {
12      // add entity UserDataModel
13      public DbSet<UserDataModel> UserData { get; set; }
14
15
16  }
17
18  // model that will be saved in db
19  public class UserDataModel
20  {
21      public int ID { get; set; }
22      public Guid Guid { get; set; }
23      public string FirstName { get; set; }
24      public string LastName { get; set; }
25      public int MailQuota { get; set; }
26      public int MailSpaceUsed { get; set; }
27      public int FileSpaceQuota { get; set; }
28      public int FileSpaceUsed { get; set; }
29      public int PrintQuota { get; set; }
30  }
```

Abbildung 20: *Entity-Datenbank*

Ein großer Vorteil dieses Frameworks ist es, dass die dahinterliegende Datenbank problemlos ausgetauscht werden kann, indem der Verbindungsstring verändert wird. Zudem kann der Entwickler sich auf die Entwicklung konzentrieren und benötigt keine Datenbankbefehle, um mit den in der Datenbank hinterlegten Daten zu arbeiten.

Das Hinzufügen von Daten in die Datenbank, genauso wie das Auslesen dieser, geschieht über die Instanziierung der Datenbank im Quellcode. In Abbildung 21 ist das Abspeichern der Daten aufgezeigt. Nach der Instanziierung wird das erzeugte Datenmodell in der Datenbank abgespeichert und daraufhin die Veränderungen in der Datenbank gespeichert.

```
60      // add data informations to database
61      using (EntityDB db = new EntityDB())
62      {
63          db.UserData.Add(dataModel);
64          db.SaveChanges();
65      }
```

Abbildung 21: *Datenbankänderungen abspeichern*

4. Zusammenfassung

4.1 Ausgangssituation

An der DHBW Stuttgart Capus Horb gibt es für jeden Studierenden und Angestellten eine private Dateifreigabe, auf welcher diese ihre Daten ablegen können. Dieses sowie ihr von der DHBW gestelltes E-Mailpostfach unterliegen hierbei einer Maximalgröße, die nicht überschritten werden darf.

Um den Anwendern eine einfache Möglichkeit zu bieten, den verbleibenden Platz in ihrem Postfach und ihrer Dateifreigabe einzusehen, wird eine Webseite benötigt, die diese Informationen anzeigt und aufbereitet darstellt.

Diese zu entwickeln und mit einer E-Mail-Adressen-Suche zu erweitern, ist das Ziel der aus zwei Teilen bestehenden Studienarbeit.

4.2 Zwischenziel

Mit dem in Kapitel 3 beschriebenen Vorgehen wurde die erste Phase der Entwicklungsarbeit abgeschlossen. Es wurde ein Designentwurf angefertigt, welcher in der kommenden Projektphase umgesetzt wird. Des Weiteren wurde der Grundaufbau des Projekts angelegt sowie eine Serververbindung zum ActiveDirectory hergestellt. Durch diese Verbindung wurde im Folgenden eine Anmeldung des Benutzers mit seinen im ActiveDirectory gespeicherten Daten am System ermöglicht. Diese Verbindung von Anwender zu der Website wurde zusätzlich mit einer Verschlüsselung versehen, um das Mitlesen Dritter zu verhindern.

4.3 Ausblick

In der kommenden Projektphase liegt der Fokus auf der praktischen Seite der Arbeit. Es wird ein Kapitel über Grundlagen zu Microsoft Exchange Server sowie zu Secure Shell geben.

Die Grundlagen für den Exchange Server werden benötigt, um das E-Mail-Kontingent der einzelnen Anwender auszulesen. Mit der Secure Shell wird der Verbindungsaufbau zum Dateiserver hergestellt, welcher unter Linux läuft und über welchen der verbleibende Speicherplatz der einzelnen Anwender im Dateisystem abgefragt werden soll.

Des Weiteren wird über die in dieser Arbeit erworbene Fähigkeit das ActiveDirectory auszulesen eine Suche realisiert, über welche nach E-Mail-Adressen anhand des Namens gesucht werden kann.

Zum Schluss wird es noch eine kleine Zusammenfassung geben und einen Vortrag, in welchem das Projekt vorgestellt und den Anwesenden die Verwendung demonstriert wird.

Abkürzungsverzeichnis

MVC	Model-View-Controller
AD	Active Directory
IIS	Microsoft Internet Information Services
OU	Organisational Unit
VS	Visual Studio
IDE	Integrated Development Environment
CLR	Common Language Runtime
JIT	Just In Time
LINQ	Language Integrated Query
DHBW	Dualen Hochschule Baden-Württemberg
SSL	Secure Sockets Layer

Literaturverzeichnis

Boddenberg, U. (2010)	Windows Server 2008 R2 , http://openbook.galileocomputing.de/windows_server_2008/windows_server_2008_kap_08_001.htm, 2010, EBook
Konopasek, K (2010)	SQL Server 2008 R2- Der schnelle Einstieg, http://books.google.de/books?id=JbMZBEHJcesC, 2010, EBook
Kowarschick, W. (2011)	Aufbau relationaler Datenbanken, http://glossar.hs-augsburg.de/Aufbau_relationaler_Datenbanken, 2011, Einsichtnahme: 31.01.2012
Microsoft Corporation (2012)	Internet Information Services (IIS) , http://www.microsoft.com/en-us/server-cloud/windows-server/internet-information-services-iis.aspx, 2012, Einsichtnahme: 31.01.2012
Microsoft TechNet (2012)	Restart IIS and AD RMS Logging Service, http://technet.microsoft.com/en-us/library/ff625744(WS.10).aspx, 2012, Einsichtnahme: 31.01.2012
Microsoft Patterns & Practices (2012)	Model-View-Controller, http://msdn.microsoft.com/en-us/library/ff649643.aspx, 2012, Einsichtnahme: 25.01.2012
Esposito, D (2009)	Comparing Web Forms And ASP.NET MVC, http://msdn.microsoft.com/en-us/magazine/dd942833.aspx 2009, Einsichtnahme: 01.02.2012
Wenz, C. u. a. (2004)	Jetzt lerne ich ASP.NET, http://books.google.de/books?id=LkyWdwIn_R8C, 2004, EBook

Mayo, J. (2010)	Microsoft Visual Studio 2010: A Beginner's Guide, http://books.google.de/books?id=unV37pUZdpQC, 2010, EBook
ScottGu (2007)	Tip/Trick: Enabling SSL on IIS 7.0 Using Self-Signed Certificates, http://weblogs.asp.net/scottgu/archive/2007/04/06/tip-trick-enabling-ssl-on-iis7-using-self-signed-certificates.aspx, 2007, Einsichtnahme: 07.02.2012
Microsoft MSDN (2012)	Schnellstart (Entity Framework), http://msdn.microsoft.com/de-de/library/bb399182.aspx , 2012, Einsichtnahme: 07.02.2012

Abbildungsverzeichnis