SEMINARARBEIT

im Studiengang Telekommunikation- und Internettechnologien

Lehrveranstaltung Projektarbeit

Xamarin Framework und Microsoft Azure – Photo Sharing App

Ausgeführt von: Christian Pipp

Begutachter: Dipl. Ing. Dr. Thomas Polzer

Wien, 01.02.2017

Inhaltsverzeichnis

Einführung 3

1 Xamarin.Forms 3

1.1 Architektur von Xamarin.Forms 3

1.2 Funktionen der Applikation 4

2 Azure 5

2.1 App Services anlegen 5

2.2 Erstellen von Datenbanktabellen 8

Literaturverzeichnis 10

Abbildungsverzeichnis 11

Tabellenverzeichnis 12

# Einführung

Xamarin bietet die Möglichkeit mit Xamarin.Forms Plattformübergreifend zu entwickeln. Dabei werden die bekanntesten Plattformen wie Android, iOS oder WindowsPhone unterstützt. Dieses Framework wird von Microsoft unterstützt und es gibt dazu zahlreiche Dokumentation welche frei von Microsoft zur Verfügung gestellt wird. Zu Xamarin gehören auch die zahlreichen Möglichkeiten der verfügbaren Schnittstellen, die man benutzen kann. Eine davon ist die Unterstützung von Azure. Damit wird es ermöglicht Smartphone Daten persistent in der Cloud zu speichern.

In diesem Projekt war es Ziel einen Photosharing App zu entwickeln die es ermöglicht über verschiedene Accounts eigene Fotos zu verwalten bzw. diese auch für andere User freizuschalten. Eine zusätzliche Kommentarfunktion solle eine soziale Komponente in das Projekt einführen. Zusätzlich war es wichtig, dass diese App über zumindest über Android und iOS laufen sollte.

# Xamarin.Forms

Für unsere Applikation wollen wir Xamarin.Forms verwenden um mit einer Codebasis die App auf allen Plattformen laufen zu lassen. Für das Projekt wurde „*Visual Studio for Mac*“ verwendet damit wird es ermöglicht Apps für iOS und Android zu entwickeln, aber nicht für das WindowsPhone, da für die Entwicklung der unterschiedlichen OS das darunterliegende betriebssystemspezifische Framework vorhanden sein muss. So muss zur Entwicklung von iOS zum Beispiel XCode installiert werden um dieses in Xamarin.Forms zu verwenden.

Der Vorteil von Xamarin.Forms liegt auf der Hand: Es ermöglicht das Entwerfen einer GUI, die dann wieder von allen OS verwenden werden kann. Die Programmierung dieser GUI kann über C# erfolgen oder aber auch über die eigene Beschreibungssprache XAML die an XML angelehnt ist und dem Beschreibungscode in der Android Entwicklung ähnelt. Leider gibt es keinen XAML Designer in der derzeitigen Visual Studio Version und somit das Design etwas erschwert.

Es gibt auch Unterschieden zwischen der Visual Studio für Mac oder Windows. Es gibt zum Beispiel nicht die volle Azure Funktionalität unter der Mac Version.

## Architektur von Xamarin.Forms

Die prinzipielle Architektur von Xamarin.Forms baut darauf auf das sich Xamarin.Forms on top des jeweiligen Betriebssystems setzt und als Compiler für diese Betriebssysteme fungiert. Das heißt es wird tatsächlicher ausführbarer Code für die jeweilige Applikation generiert und Xamarin.Forms agiert nicht als Interpreter. Am besten veranschaulicht untenstehende Abbildung den Aufbau.

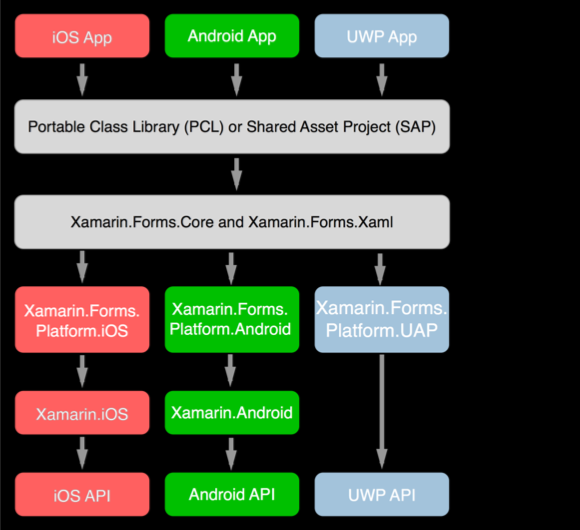


Abbildung : Architektur Xamarin.Forms (Quelle: [1])

Dabei erkennt man auch dass es in Xamarin.Forms zwei Möglichkeiten für ein Xamarin.Forms Projekt gibt;

* Portable Class Library
* Shared Asset Project

Diese unterscheiden sich in einigen wenigen Details, wobei die Portable Class Library die ältere Version davon ist und nach wie vor verwendet wird.

## Funktionen der Applikation

Die Photos Applikation sollte dabei einige rudimentäre Funktionen beinhalten.

# Azure

Zum persistenten Speichern der Daten wurde Azure gewählt und dazu ein Studenten Account über Imagine Microsoft erstellt um dort entsprechende SQL Datenbanken anzulegen und ein sogenanntes App Service zu hosten. Damit konnte man die Datenbankzugriffe direkt in das Xamarin.Forms Projekt einbinden. Diese erfolgte ein paar Schritten direkt im Portal von Azure um alle notwendigen serverseitigen Einstellungen vorzunehmen wichtig dabei war das sich alle Applikationen in der selben Ressourcengruppe befinden müssen. Folgender Abschnitt soll nun die notwendigen Schritte zeigen die notwendig waren um Azure in Xamarin.Forms einzubinden.

## App Services anlegen

Im ersten Schritt wird ein App Service angelegt um damit die Verbindung von der Clientsoftware – die Xamarin.Forms App – zum Server abzubilden. Dies funktioniert relativ einfach und man muss sich dazu mit einem gültigen Microsoft Account in das Azure Portal einloggen. Dort wählt man den im linken Menü den Punkt *App Services* aus wie in der unteren Abbildung gezeigt.

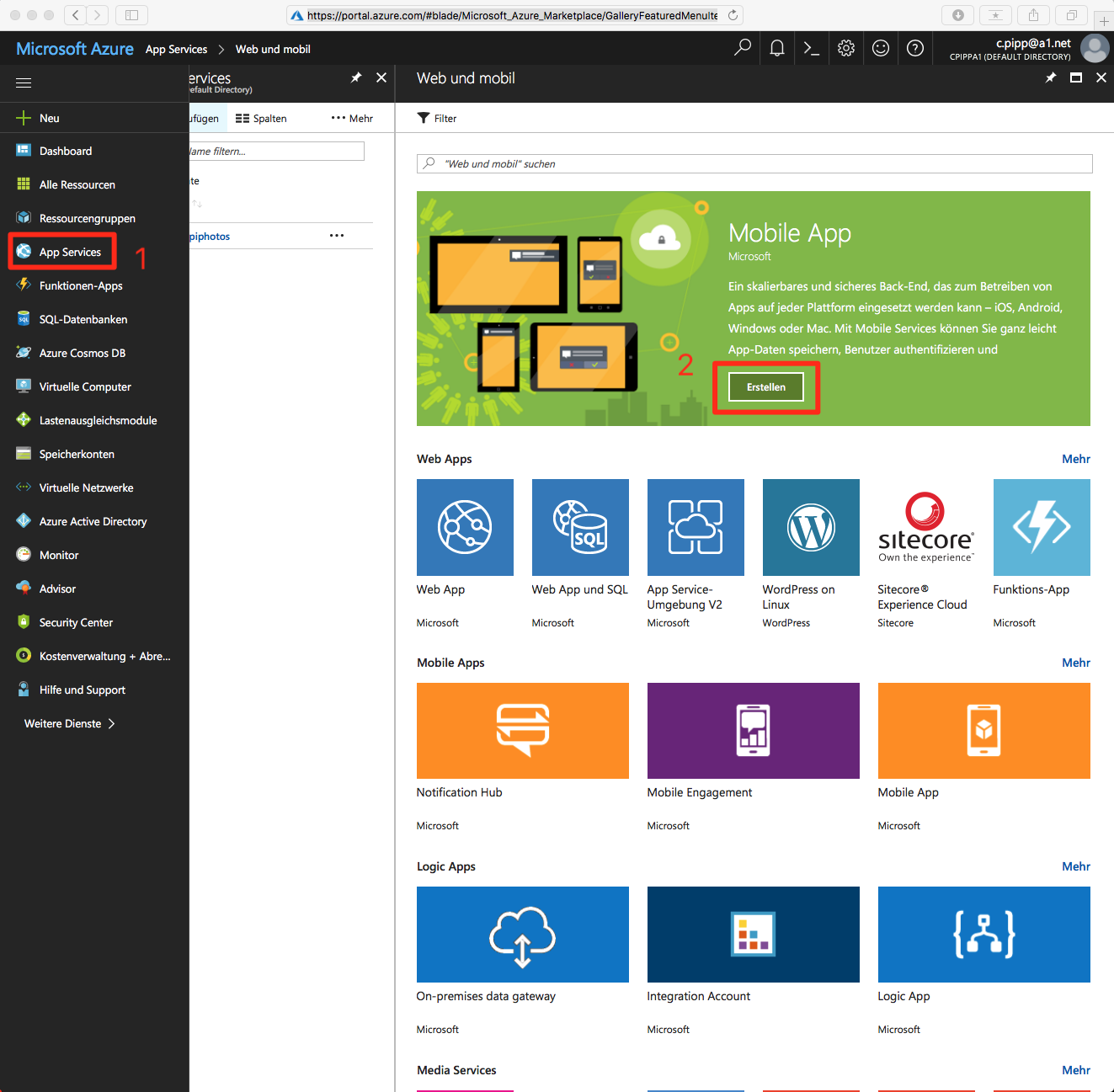


Abbildung : Einrichten eines App Services

Im nächsten Schritt wird man gefragt nach dem Namen des App Services gefragt und in welcher Ressourcengruppe das Service angelegt werden soll. Dabei ist es wichtig dass diese Ressourcengruppe gleich mit der Gruppe des Windowsservers sein muss sonst kommt es später zu Problemen, da sich Elemente unterschiedlicher Gruppen nicht austauschen können.

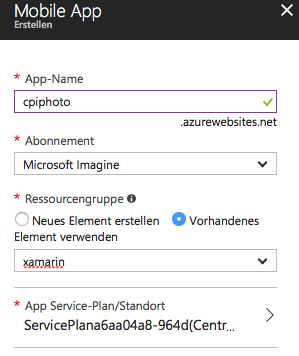


Abbildung : Benennen des App Services

Gleich anschließend wird das gewünschte Service erstellt, dies kann auch einige Minuten dauern. Nach Abschluss sollte man das Service in der Übersicht wieder finden:

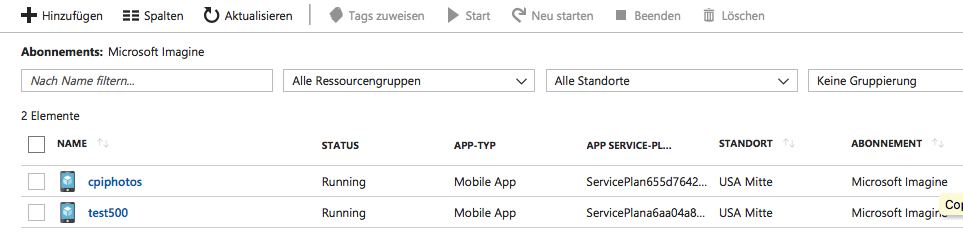


Abbildung : Übersicht App Services

Danach geht es weiter mit der Anlage einer MSSQL Datenbank. Dazu muss ein Microsoft Server zur Verfügung stehen auf welchen die Datenbank laufen soll. Zur Anlage dieser Datenbank verwendet man am besten den Punkt Schnellstart im App Service Menü, der unter dem Punkt Bereitstellung zu finden ist. Da wir unser Applikation für iOS und Android lauffähig machen wollen wählen wir auch dementsprechend Xamarin.Forms aus

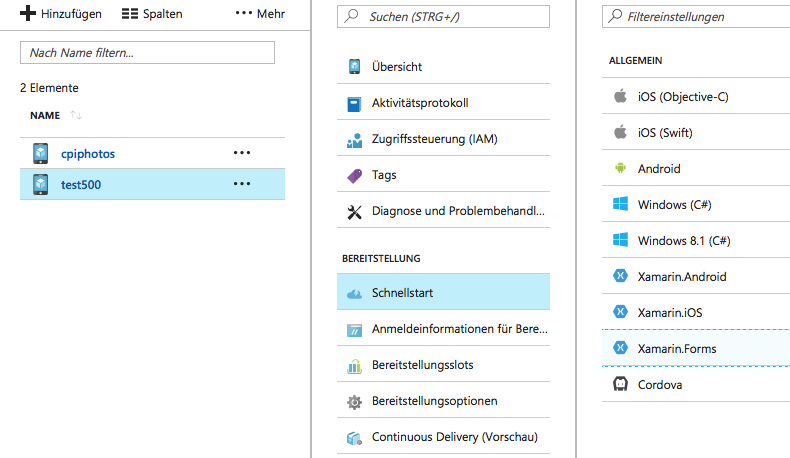


Abbildung : Einrichten Datenbank Azure

Anschließend kann man die Datenbank festlegen, welche Backendsprache man verwenden will und sich damit ein Beispielprojekt generieren lassen (an Hand einer einfachen Todo Liste). Man kann dann allerdings auch noch später die gewünschten Datenbanktabellen anlegen. Dies macht man zu Beginn am besten über das Azure Portal, da auch gleichzeitig alle notwendigen serverseiteigen Skripts angelegt werden. Dies ist notwendig da in der Applikation nicht direkt mit dem MSSQL Server kommuniziert wird sondern das Backend des Server aufgerufen wird und alle Abfragen handelt. Als Backend Sprache wird *Node.js* benutzt.

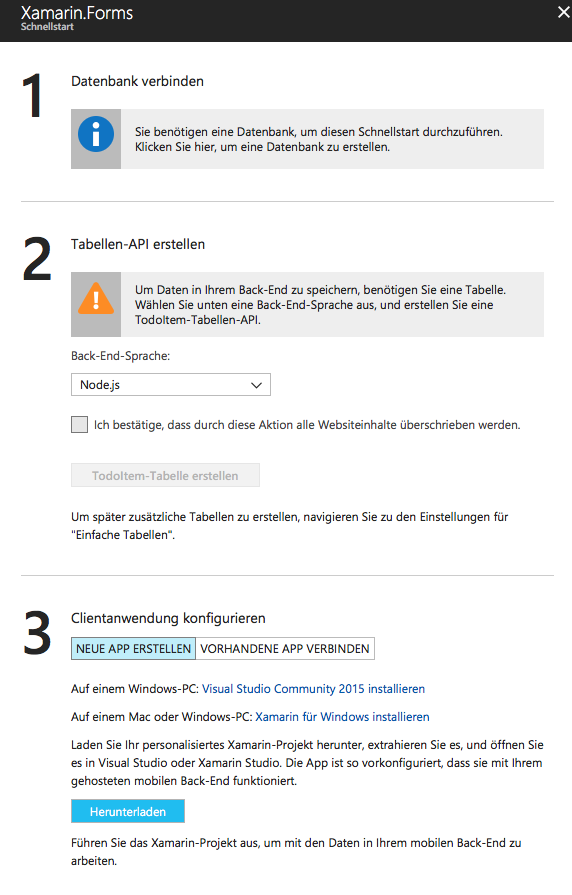


Abbildung : Grundkonfiguration App Services

Nachdem die Datenbank verknüpft ist, ist am Server (bis auf die Datenbank Tabellen) einmal alles soweit eingerichtet. Man könnte sich testweise auch eine TodoItem Tabelle erstellen und dann damit eine Testprojekt generieren. Als nächsten Schritt werden nun Datenbanktabellen, die für die Applikation benötigt wird angelegt.

## Erstellen von Datenbanktabellen

Wir wollen das Anlegen der Datenbanktabellen im Azure Portal machen, da damit auch alle notwendigen Backendskripts generiert werden. Diese kann man im Punkt *Mobil -> Einfache Tabellen* anlegen [2]. Je nachdem ob man die TodoItem Tabellen generieren hat lassen oder nicht gibt es dort schon Einträge.

Literaturverzeichnis

[1] Charels Petzold, *Creating Mobile Apps with Xamarin.Forms*, 1. Auflage. Microsoftpress.

[2] „Getting Started with Azure Mobile Apps’ Easy Tables |“, *Xamarin Blog*, 28-Jan-2016. .

[3] Martin Sauer, *Grundkurs Mobile Kommunikationssysteme*, 5. Aufl. Wiesbaden: Springer Vieweg, 2013.

Abbildungsverzeichnis

Abbildung 1: Architektur Xamarin.Forms (Quelle: [1]) 4

Abbildung 2: Einrichten eines App Services 5

Abbildung 3: Benennen des App Services 6

Tabellenverzeichnis

**No table of figures entries found.**  
In your document, select the words to include in the table of contents, and then on the Home tab, under Styles, click a heading style. Repeat for each heading that you want to include, and then insert the table of contents in your document. To manually create a table of contents, on the Document Elements tab, under Table of Contents, point to a style and then click the down arrow button. Click one of the styles under Manual Table of Contents, and then type the entries manually.