Utveckling av Androidapplikation med C# och .NET

Bokningssystem

LUNDS UNIVERSITET
Campus Helsingborg

LTH Ingenjörshögskolan vid Campus Helsingborg
Institutionen för datavetenskap

Examensarbete:
Ammar Al-Suhairy

LTH Ingenjörshögskolan vid Campus Helsingborg
Lunds universitet
Box 882
251 08 Helsingborg

LTH School of Engineering
Lund University
Box 882
SE-251 08 Helsingborg
Sweden

Tryckt i Sverige
Lunds universitet
Lund 2016

Sammanfattning

Detta examensarbete har utförts åt Omegapoint AB. Syftet med examensarbetet var att utveckla ett digitalt bokningssystem med C# och .NET. Med hjälp av bokningssystemet skulle företagets anställda kunna boka, ändra och avboka deras konferensrum. I syfte att minska risken för tidsdröjningen, och för att uppnå ett resultat som passar med uppdragsgivaren önskemål, bestämdes det att utvecklingen och samarbetet med kund skulle ske agilt. Analys ingick som en del i examensarbetet och utvärderade bland annat utvecklingsmiljöer, databas, databashanterare och operativsystem.

Baserat på analysresultaten valdes utvecklingsmiljön Xamain i Visual Studio. Valet av databasen blev relationsdatabas med MySQL som databashanterare och PHP som frågaspråk. I början av examensarbetet bestämdes att skulle utvecklas en mobilapplikation med bokningskalender till Android & iOS, men på grund av begränsad resurs blev utvecklingen bara till Android.

I produkten som utvecklades kan användaren skapa event, ändra, radera samt bjuda in andra medarbetare till eventet snabbt och enkelt. Detta sker genom att avändaren väljer direkt den tiden han vill boka eller klickar vart som helst i kalendern och därefter ändra datum och tid i tidsfälten. I rapporten presenteras en mer detaljerade beskrivning av ovanstående nämnda punkter

Nyckelord: Android, C#, MySQL, Xamarin

Abstract

This exam work has been carried by Omegapoint AB. The goal with this exam work was to develop a digital booking system with C# and .NET. With the help of the booking system the company employees would be able to book, change and cancel the conference room booking. In order to reduce the risk of time delay and to reach the results that would fit the client's goal. The decision was made that the development and cooperation with the customer would happen in an Agile method, also with the partial analysis that was made. The analysis evaluated, among other things, development environment, data base, data control and operation system.

Based on the results of the analysis were chosen Xamain development environment in Visual Studio. The choice of the database became the relational database with MySQL as datacontrol and PHP as a questioning language. In the beginning of the exam work it was decided to develop a mobile application with booking calendar for android and iOS, but because of limited resources the development happened only for android.

The product that was developed can be used by the user to book, change, and delete, as well as invite other colleagues to a specific event fast and easily. This happens through the user choosing directly the time in which he will book or clicks anywhere in the calendar and then change the time and date in that particular time. In the report more detailed description of the above named points will come.

Keywords: Android, C#, MySQL, Xamarin

Förord

Detta arbete har genomförts för Omegapoint AB. Examensarbetet ingår som en avslutande moment för utbildningen i Lund Tekniska Högskola.

Jag vill tacka Fredrik Lundbeck för möjligheten att utföra arbetet på Omegapoint. Ett tack till Sebastian Lundh min handledare och ett tack till alla på Omegapoint.

Jag vill även tacka Christian Nyberg min handledare och min examinator Christin Lindholm.

Malmö, 2016-06-08
Ammar Al-Suhairy

Ordlista

AOSP	Android open source-project. Delen av Android operativsystem som ansvarar för utveckling av Android
JSP	Java Server Pages. Frågaspråk, utveclkad av Sun Microsystems, som används för att hämta svar från en webserver.
ASP	Active Server Pages. Frågaspråk utvecklad av Microsoft som används för att hämta svar från en webbserver.
GPL	General Public License. En öppen källkod licens som används av MySQL.
CIL	Common Intermediate Language. Ett assemblyspråk.
DBMS	Database management systems. Se avsnitt 2.4.
SQL	Structured Query Language. Se avsnitt 3.4.2.
ARM	Advanced RISC Machine. En typ av RISC (Reduced Instruction Set Cumputing) arkitekturer för processor.
RDM	Raima Database Manager. Ett databasledningssystem som implementerar nätverksmodellen.
SDK	Software development Kit. En uppsättning mjukvara utvecklingsverktyg som gör det möjligt att skapa applikationer för en viss mjukvara.
Malware	Program som installeras i enheten utan användarens samtycke, dessa program kan vara spyware, virus, masker eller trojaner

Innehållsförteckning

1 Inledning 1
1.1 Bakgrund 1
1.2 Syfte 1
1.3 Målformulering 1
1.4 Problemformulering 1
1.5 Avgränsningar 2

2 Teknisk Bakgrund 3
2.1 Mobila plattformar 3
2.1.1 Android 3
2.1.2 iOS 4
2.1.3 Windows Phone 4
2.2 Plattformar och utvecklingsmiljöer 4
2.2.1 Android studio 4
2.2.2 Xcode 4
2.2.3 Visual Studio 4
2.3 Cross Plattform 5
2.3.1 Cordova 5
2.3.2 Xamarin 6
2.4 Databas 6
2.4.1 Databastyper 6
2.4.2 SQL 7
2.5 PHP 8
2.6 **Syncfusion 9**

3 Metod 11
3.1 Förstudie 11
3.2 Kravsamling 11
3.2.1 Intressentanalys 11
3.2.2 Intervju 11
3.2.3 Observation 12
3.2.4 Enkät 12
3.2.5 Kravspecifikation 12
3.3 Utveckling av prototyper 13
3.4 Projektmodell 13
3.5 Dokumenteringsmetod 14

4 Analys 15
4.1 Liknande produkter 15
4.2 Android vs iOS 15
4.2.1 Gränssnitt 16

4.2.2 Säkerhet ... 16
4.2.3 Utveckling och publicering av applikationer ... 17
4.2.4 Val av enhet ... 17
4.2.5 Val av plattform ... 17
4.3 Val av utvecklingsmiljö ... 18
4.3.1 Xamarin i Visual Studio ... 18
4.3.2 Xamarins sätt att dela kod ... 19
4.3.3 Fördelar med Xamarin ... 20
4.4 Val av databas ... 21
4.4.1 Objektorienteraddatabas vs Relationsdatabas ... 21
4.4.2 Databashanterare ... 23
4.5 Prototyp ... 24
4.6 Databasens struktur ... 25
4.7 Användartester ... 25
4.8 Källkritik ... 26
5 Resultat ... 28
5.1 Inloggningsskärm ... 28
5.2 Lösenord återställningsskärm ... 29
5.3 Bokning ... 30
5.4 Tidsändring ... 31
5.5 Tidsradering ... 31
5.6 Testresultat ... 32
6 Slutsats ... 33
6.1 Svar på examenarbetets problemformuleringar ... 33
6.2 Övriga slutsatser ... 34
6.3 Framtida utvecklingsmöjligheter ... 35
7 Källförteckning ... 36
8 Bilagor ... 38
8.1 Databasens struktur ... 38
8.2 Prototyper ... 39
8.2.1 Prototyp1 ... 39
8.2.2 Prototyp2 ... 40

1 Inledning

1.1 Bakgrund

Företaget Omegapoint AB i Malmö hade behov av ett digitalt bokningssystem för deras konferensrum. I konferensrummet kan Omegapoints anställda ha olika slags event såsom personalmöte, projektredovisning, intervjuer etc.
Omegapoint är ett stort företag som finns i Stockholm, Göteborg, Malmö och Umeå. Medarbetarna i Omegapoint arbetar främst inom applikationssäkerhet. Enligt Great Place TO Work [28] i år är Omegapoint för tredje år i rad raknat som en av de bästa arbetsplatserna i Sverige.
Omegapoint vill ha ett digitalt bokningssystem, då alla bokningar sker manuellt i dagens läge, det tar tid och det finns risk för dubbelbokningar. Det digitala systemet ska göra en bokning enklare för deras medarbetare. Att boka ett rum samt bjuda in andra medarbetare till ett visst event, skall vara enkelt och tidsbesparande, man skall inte behöva gå runt och fråga om vilka medarbetarna kan delta i ett event eller inte. Alla medarbetare ska ha tillgång till systemet för att kunna boka och se andra bokningar, systemet ska ej vara beroende av en bokningsansvarig.

1.2 Syfte

Syftet med examensarbetet är att underlätta för Omegapoints medarbetare att bjuda in medarbetare till möten och att boka lokaler för dessa möten. En mobilapplikation som är skräddarsydd för Omegapoints medarbetare kommer att utvecklas för detta ändamål.

1.3 Målformulering

Det ursprungliga målet med examensarbetet var att designa och utveckla en applikation som skall fungera i Android och IOS. Utvecklingen skulle ske i en .NET miljö där programmeringsspråket ska vara C#. Med systemet skall man kunna boka och avboka konferensrummet samt kunna bjuda in andra medarbetare till ett event där även medarbetarna skall kunna ge ett svar för eventet.
Men det blev ändring i och med att utvecklingen för iOS inte gjordes under detta examensarbete på grund av begränsad resurs

1.4 Problemformulering

Här listas de problem som arbetet ska ge svar på:

- Vilka system finns på marknaden och vilka är deras fördelar och nackdelar för Omegapoints anställda?
- Vilka metoder kan användas för att analysera Omegapoints anställdas behov?
- Vilken plattform skall applikationen utvecklas på?

- Vilken databas kan vara lämplig för att spara information från bokningssystemet?

1.5 Avgränsningar

I detta avsnitt kommer avgränsningarna att presenteras

- Applikationens användbarhet skall testas på Android och IOS. Detta var ursprunglig avgränsning, sedan blev det bara Android.
- Applikationen avgränsas till ett språk (svenska eller engelska).

2 Teknisk Bakgrund

Detta kapitel beskriver de tekniker som använts för att analysera och utvärdera tekniker och för att utveckla prototypen.

2.1 Mobila plattformar

De senaste tio åren har de smarta telefonernas frammarsch förändrat sättet vi kommunicerar på. Om man har en smartphone så kan man enkelt söka efter information på webben, titta på video och hantera e-post i telefonen, precis som man gör på datorn, men det finns mer att göra än dessa enkla exempel. Man kan ladda ner applikationer och göra alla möjliga saker som kontrollera sina Facebook och Twitter-flöden, hantera sitt bankkonto, beställa mat och spela spel. Man kan planera händelser på telefonens kalender och se dem på sin dator eller planera de i sin dator och plocka upp dem på sin telefon. Man kan också titta på film eller lyssna på musik.

Av alla 2.4 miljarder smarta telefoner och läsplattor som såldes under 2015 använder mer än 54% operativsystemet Android [1]. De konkurrenter som utmanar på marknaden är bland annat iOS och Microsoft. Av de 2.4 miljarderna enheter var 283 miljoner med operativsystemet Windows och 293 miljoner enheter var med iOS [1]. Men ingen av dem kommer i närheten av Android, som är störst med 1,3 miljarder sålda enheter. Android sålde mer än dubbel så mycket som både Windows och iOS tillsammans [1].

2.1.1 Android

Android är ett operativsystem för mobila enheter som används främst i smarta telefoner och läsplattor. Android utvecklades ursprungligen av Android, Inc., som köptes av Google 2005. Operativsystemet lanserades 2007 [2]. Google släppte Android med öppen källkod under Apache Licence. Android Open Source Project (AOSP) som ansvarar för utveckling av Android [3].

Operativsystemet är baserat på en Linuxkärna. Utvecklingen av den ursprungliga Linux-kärnan kommer att fortsätta oberoende av Android [3].

Utveckling av Android-applikationer sker i första hand genom Android SDK som använder sig primärt av Java som programmeringsspråk. I Android finns det olika versioner och beskrivs med hjälp API-nivåer [4].

Java kan kopplas ihop med C/C++ och tillsammans med ett antal icke-standard drivrutiner som ger ett bättre stöd för Androidutveckling i C++ [5]. Det är även möjligt att utveckla med språket C#. Utvecklingen med C# måste ske med Xamarin-plattformen, vilket kommer att diskuteras lite senare i avsnitt 4.2.3.

Google play store är den största applikation store med applikationer som kan installeras på Androidenheter. Applikationerna från Google play store uppfyller Googles kompatibilitetskrav och tillstånd för Google mobiltjänst för programvaran [4].

2.1.2 iOS
iOS är ett mobilt operativsystem som skapats och utvecklas av Apple och används bara i deras mobila enheter iPhone, iPad och iPod. iOS är baserat på Darwin och körs på ARM-baserade processorer [6]. Darwin är en del av OS X som Apple har släppt som öppen källkod. Den delen fungerar som ett operativsystem i sig men saknar de grafiska funktioner och andra delar.
Applikationer till iOS utvecklas i Objective C eller Apples egna språk Swift, men det är även möjligt att använda C, C++ och C#. Utvecklingsmiljön Xcode innehåller fullständiga verktyg för att skriva, kompilera och debugga iOS-produkter. Detta beror på att Xcode har stöd av Apples ramverk Cocoa Touch. Där finns det färdiga klasser och funktioner för iOS [7][8].
Applikationer för iOS är tillgängliga via Apple App Store. För att publicera appar på App Store krävs medlemskap i Apple Developer Program, samt ett godkännande från Apple för varje app.

2.1.3 Windows Phone
Windows Phone är ett mobilt operativsystem som är skapad och utvecklas av Microsoft. Det lanserades i oktober 2010 med Windows Phone7 och senaste operativsystem för mobila enheter heter Windows Phone 10 [7]. Den senaste versionen (10.0) släpptes i jan 2015. Utveckling av applikationer för Windows Phone görs i Microsoft Visual Studio och skrivs i C/C++ samt C# [8].

2.2 Plattformar och utvecklingsmiljöer

2.2.1 Android studio
Androids mobilapplikationer utvecklas främst med Android Studio, som är utvecklat av Google och den är tillgänglig för Windows, iOS och Linux. Android Studio har fullständiga verktyg för att skriva, kompilera och debugga Android-produkter. Detta har sin grund i att den är baserad på IDEA IntelliJ som utvecklas av JetBrains och har ersatt det tidigare utvecklingsverktyget ADT Bundle (Android Developer Tools Bundle) som använder Eclipse IDE. Android rekommenderas framför ADT Bundle och detta beror på att det är mer stabil [10].

2.2.2 Xcode
Xcode är en integrerad utvecklingsmiljö (IDE) som innehåller en uppsättning verktyg för programvaruutveckling som utvecklats av Apple för att utveckla programvara för OS X, iOS, WatchOS och tvOS.
Xcode stöder källkod för programmeringsspråken C, C ++, Objective-C, Java, Python, och Swift, med en mängd olika programmeringsmodeller [8][10].

2.2.3 Visual Studio
Visual Studio är en integrerad utvecklingsmiljö(IDE) som är utvecklad av Microsoft. Den har fullständiga verktyg för att skriva, kompilera och debugga Microsoft datorprogram samt mobilapplikationer för WP, Andriod och iOS.
Visual Studio stöder källkod för många programmeringsspråk i varierande grad såsom C, C++, C#, Visual Basic, XHTML, Java Script och CSS. Genom att använda Visual

Studio kan man komma åt andra verktyg som ägs av Microsoft eller av företag som Microsoft samarbetar med [12].

2.3 Cross Plattform

Cross-plattformar stöder utvecklingen för olika operativsystem samtidigt via ett gemensamt bibliotek. Koden för programmens logik kan vara gemensam för de olika operativsystemen. Men koden för användargränssnittet(UI) måste skrivas för vart och ett av operativsystemen. I rapporten kommer de två mest kända cross-plattformarna att presenteras, vilka är Cordova och Xamarin. Tabell 2.3.1 visar en jämförelse mellan Cordova och Xamarin.

Tabell 2.3.1: Fakta kring utvecklingsmiljöerna som studerades

	Cordova	Xamarin
Ägs av	Adobe System	Microsoft
Utvecklingsspråk	JS,HTML,CSS	C#
Plattformstöd	Android, iOS, WP , BlackBerry ,Symbian ,mf.	Android, iOS, WP
Hybrid/Native	Hybrid	Native
Utvecklingsmiljö	Valfri texteditor / IDE	Xamarin Studio och Visual Studio
Möjlighet att utveckla för iOS utan Mac	Nej	Nej
Stöd för Pushmeddelanden	Ja,via plugin	ja
Kostnad	1 app gratis, annas $10/månaden	Nu graits, innan 1 app gratis(max 64k),annar $299/år/utvecklare

2.3.1 Cordova

Cordova är ett utvecklingsverktyg som skapades av Nitobi och köptes upp av Adobe System 2011, dessförinnan kallades det för PhoneGap. Cordova släpptes senare som en öppen källkod version av programvaran som kallas Apache Cordova. Cordova tillåter utvecklaren att utveckla applikationer för mobilenheter genom att använda CSS3, HTML5 och JavaScript i stället för att använda plattformsspecifika API:er som de i Android, iOS eller Windows Phone.

Cordova sammankopplar funktioner i JavaScript med mobilernas hårdvarusystem. Detta ger möjlighet att hantera och använda mobilens hårdvarukomponenter, som till exempel kamera och GPS. Kopplingen mellan Cordova och enhetens hårdvara är unik för varje operativsystem, vilket medför att endast en prototyp behöver utvecklas för att fungera till flera plattformar [14].

2.3.2 Xamarin

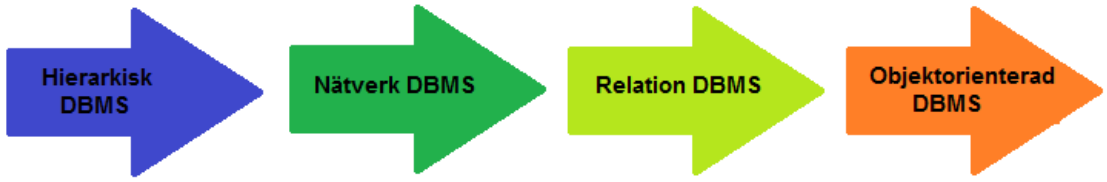
Xamarin är ett utvecklingsverktyg som ägs av Microsoft. Med Xamarin är det möjligt att utveckla appar genom att skriva kod i JavaScript, Visual Basic, C++ och C#. Xamarins användare kan utveckla appar för Android, iOS och Windows med C# i ett och samma projekt genom att använda en gemensam C#/.Net kodbas. Användaren kan använda två olika utvecklingsmiljöer. Det är möjligt att utveckla appar i Visual Studio eller Xamarin Studio och både två är likvärdiga [12].

2.4 Databas

En databas är en systematiserad samling data. Eftersom data i en databas är organiserad, underlättas datahantering. Database Management System (DBMS) är en samling av program som gör det möjligt för användare att få tillgång till en databas, manipulera data, samt kontrollera tillgången till databasens bibliotek. DBMS är inget nytt begrepp och introducerades redan på 1960-talet. Charles Bachman lanserade Integrated Data Store(IDS) och det sägs att han var den första databashanteraren i historien. Med tiden har tekniken utvecklats och anatalet funktioner i databashanterare har ökat mycket [16].

2.4.1 Databastyper

För att se hur DBMS har utvecklats med tiden, se figur 2.4.1 som visar utvecklingen av DBMS kategorier. Det finns fyra huvudsakliga typer av databashanterare:



Figur 2.4.1 Visar utvecklingen av DBMS

2.4.1.1 Hierarkisk DBMS

Denna typ av DBMS utnyttjar föräldra-barnrelationen att lagra data. Denna typ av DBMS används sällan idag. Den hierarkiska strukturen är som ett träd med anteckningar som representerar poster och grenar som representerar fält. Windows-registret som används i Windows XP är ett exempel på den hierarkiska databasen, där inställningarna lagras som trädstrukturer med [17].

2.4.1.2 Nätverks DBMS

Denna typ av DBMS stödjer många-till-många relationer vilket oftast resulterar i komplexa databasstrukturer. RDM server är ett exempel på ett databasledningssystem som implementerar nätverksmodellen [17].

2.4.1.3 Relation DBMS

Denna typ av DBMS används för att hitta databasrelationer i form av tabeller så kallad relation DBMS. En skillnad från nätverks DBMS är att RDBMS inte stöder många-till-

många relation. RDBMS har oftast förväg definierade datatyper som de kan stödja. Relation DBMS är den mest populära DBMS typen i marknaden. Exempel på RDBMS är MySQL, Oracle och Microsoft SQL Server [16].

2.4.1.4 *objektorienterad RDBMS*

Denna typ stöder lagring av nya datatyper, som lagras i form av objekt. Objekten som ska lagras i databasen har attribut till exempel kön eller ålder och metoder som definierar vad som kan göras med dessa data. PostgreSQL är ett exempel på objektorienterade RDBMS [16].

2.4.2 SQL

Structured Query Language (SQL) är standardspråket för att hantera relationsdatabaser. SQL kan effektivt användas för att sätta, söka, uppdatera och ta bort databasposter. Men det betyder inte att SQL inte kan göra saker utöver det, i själva verket bidrar SQL till optimering och underhåll av databaser. Relationsdatabaser som kan hantera är t.ex MySQL, Oracle och MS SQL [19].

2.4.2.1 MySQL

MySQL är ett relationsbaserat databashanteringssystem och den är världens mest populära databas [18]. MySQL är öppen källkod och används under GNU General Public License (GPL). MySQL har utvecklats av det svenska företaget MySQL AB. Idag ägs det av Oracle. Det är gratis att använda MySQL, men det finns en företagsversion som har utökad funktionalitet och support och denna version är inte gratis. Enligt Oracle används MySQL i nio av de tio mest trafikbelastade webbplatserna eftersom den är plattformsoberoende och har stöd för alla standardfunktioner. För att se hur man skriver en enkel SQL fråga med MySQL [16][18], se figur 2.4.2.1

```
SELECT age
FROM person
ORDER BY age ASC
LIMIT 1 OFFSET 2
```

Figur 2.4.2.1 Visar hur man kan skriva en SQL fråga med MySQL

2.4.2.2 Microsoft SQL Server

Microsoft SQL Server är ett managementsystem för relationsdatabaser som utvecklas av Microsoft. Microsoft SQL är en databasserver med huvudfunktion att lagra och hämta data som begärs av andra program. Detta kan köras antingen på samma dator eller på en annan dator i ett nätverk (inklusive Internet) [15].

Microsoft har olika versioner av Microsoft SQL Server som riktar sig till olika målgrupper och för arbetsbelastningar som varierar från små enkla maskinapplikationer till stora internetapplikationer med många samtidiga användare [18]. För att se hur man skriver en enkel SQL fråga med SQL Server, se figur 2.4.2.2

```
SELECT TOP 3 WITH TIES *
FROM person
ORDER BY age ASC
```

Figur 2.4.2.2 Visar hur man kan skriva en SQL fråga med Microsoft SQL Server.

2.5 PHP

Hypertext Preprocessor är ett populärt och gratis skriptspråk som är utformat främst för webbutveckling med dynamiskt innehåll. Ursprungligen skapades det av Rasmus Lerdorf 1994 och ägs av PHP-gruppen. PHP är öppen källkod och publicerad under PHP-licens. Det finns också en kommandotolk ”CLI-applikation” för PHP som gör att skript kan användas för fristående program direkt, utan att vara kopplad till webbserver. PHP kan ofta jämföras med Java Server Pages (JSP) från Sun Microsystems, eller Active Server Pages (ASP) från Microsoft [20].

PHP passar de flesta databaser och har som huvudsaklig uppgift att fungera som ett filter. Källkoden tolkas av en interpretator kallad Zend Engine, där slutresultat skrivs ut i form av en textström. Resultatet påverkas av indata som programmet får vid körning, oftast i form av instruktioner och text. Ofta används kommunikation med databaser för att presentera lagrad information. Mottagaren är oftast en webbläsare [20]. Figur 2.5.1 visar hur ett enkelt exempel på hur PHP kod ser ut.

```
<?php
require"init.php";
$mail=mysqli_real_escape_string($con, $_POST["username"]);
$pass= mysqli_real_escape_string($con,$_POST["pass"]);
$sql_query= "SELECT name,mail FROM user_info WHERE mail = '$mail' AND password = '$pass'";
$result=mysqli_query($con,$sql_query);
$row=mysqli_fetch_row($result);
$count=mysqli_num_rows($result);


$login_info=array();
if($count=="1")
{
$login= array("Id"=>$row[0],"Name"=>$row[1],"Count"=>$count);
array_push($login_info,$login);
echo json_encode($login_info);
}

else
{
echo"Login Failed...Try Again";
}
```

Figur 2.5.1 Ett enkelt exempel på hur man skriver PHP kod

2.6 Syncfusion

Syncfusion är ett företag som levererar ett brett utbud av webb-, mobil- och skrivbordskontroller (såsom diagram, schema, kalender och kareta) tillsammans med en serviceinriktad strategi under hela applikationens livscykel. Syncfusion har etablerat sig som en partner över hela världen för användning i affärskritiska applikationer. Grundades 2001 och har sitt huvudkontor i Research Triangle Park, North Carolina [21]. I detta examensarbete valdes att implementera en kalender från Sysfunction. Detta beror på att Sysfunctions verktyg ger tillgång till många färdiga funktioner. Om Kalendern skulle utformas och utvecklas manuellt, skulle utvecklingsprocessen spränga tidsramarna för ett examensarbete.

3 Metod

I detta kapitel beskrivs de metoder och arbetssätt som använts under examensarbetet. Examensarbetet bestod av tre delar, en förstudie, en kravinsamling och en utvecklingsdel vilka beskrivas i detta kapitel.

3.1 Förstudie

Eftersom examensarbeteren tidigare inte har arbetat med C# och Visual Studio, så var inlärningsprocessen en stor del av arbetet. Den delen skedde främst via Youtube, där information inhämtades från föreläsningar av olika förläsare [26][27]. Föreläsningarna kompletterades även med böcker inom ämnet som [23][29].

3.2 Kravsamling

Under denna del har olika metoder använts för att inhämta den domänkunskap som behövs för att kunna skriva kraven och sedan utveckla prototypen.

3.2.1 Intressentanalys

På grund av att examensarbetet handlade om Omegapoints behov och Omegapoint är beställaren, så är intressenterna Omgepoints anställda. Detta var både positivt och negativt, positivt för att man alltid har någon intressent tillgänglig som man kunde fråga eller intervjua. Men det var också negativt, eftersom de många intressenterna hade många olika önskemål som krockade med varandra. Detta problem lede till svårigheter när kravspecifikationen skulle tas fram i början av examensarbetet. Problemet löstes genom att sortera de önskemål som passade bäst med varandra och behöll systemets syfte, under ett mötte med chefen förklarades för- och nackdelarna för de olika grupperna.

3.2.2 Intervju

En ostrukturerad intervju med chefen och handledaren gjordes i början av examensarbetet för att förstå deras behov och verifiera målet med arbetet. Projektbeskrivningen som gjordes i början av examensarbetet baserasdes på denna intervju. Fler intervjuer gjordes i samband med informationsamlingen och vissa av dessa intervjuer var strukturerade och semi-strukturerade [30]. Strukturerade intervjuer valdes eftersom intervjuerna var strukturerad med sina delar, med bestämda frågor och bedömningar. Bedömningarna har bestämda checklistor eller bedömningsskalor. Semi-strukturerade intervjuer valdes för att man kan tillåta sig att följd ställa frågor utifrån vad kondidaten svarar. Dessutom båda typer valdes för att minska risken för ledande frågor och få frågorna besvarade. Antal intressenterna som blev intervjuade var olika och detta hade sin grund i antal tillgängliga intressenter vid tillfället. Under intervjuerna ställdes frågor om hur applikationen skulle fungera, vad den skall användas till och hur de vill att en funktion skulle fungera i detalj. Intressenternas önskemål antecknades och användes vid kravidentifiering.

3.2.3 Observation
I början av examensarbetet studerades befintliga bokningssystem. Två av de analyserade applikationerna var Tiny Calendar och PocktLife. De valdes på grund av högt betyg i Google play store och Apple app store. Funktionaliteten som de två applikationer stödjer undersöktes, samt hur olika problem löstes. Tre användare fick använda de två applikationer och deras användning blev observerades. I avsnitt 4.1 förekommer mer detaljerade beskrivning av undersökningen.

3.2.4 Enkät
En annan metod som använts under informationsamlingen var enkätundersökning. Enkäten användes först och främst för att kontrollera vilken plattform som används mest i företaget, så att utvecklingen börjas för denna plattform först.
Intressenterna fick svara på enkla flervals- och svarfältsfrågor såsom:

- Vilket operativsystem använder de?
- Hur ofta brukar de boka rummet?
- Vilken information vill de veta om den bokade tiden?
- Önskar de sig en speciell funktion?

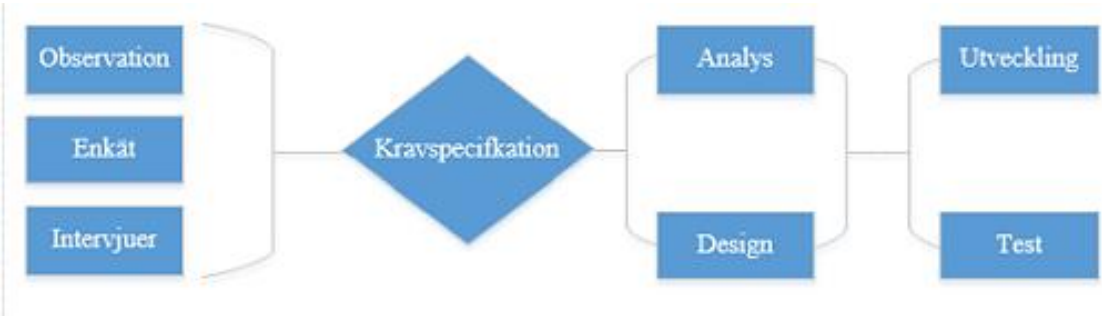
Enkäten delades i papperform till 17 användare under en rast i företaget. Bland de som fick enkäten var chefen och handledaren, alla som fick enkäten svarade på frågorna. För mer detaljer gående enkäten och dens resultat se avsnitt 4.2.5.

3.2.5 Kravspecifikation
Resultaten av de ovannämnda avsnitt i kapitel 3 gav en klar bild av vilka funktioner som mobilappalikationen skulle innehålla. Detta lede till utvecklingen av kravspecifikationen. Den innehöll bara högnivåkrav för systemets beteende såsom:

1- Användaren ska kunna logga in med email och lösenord.
2- Användaren ska vara inloggad tills man själv loggar ut.
3- Användaren bör kunna klicka på tabellen för att välja en tid.
4- Användaren ska kunna boka när som helst (24/7).
5- Användaren ska kunna ändra den valde tiden och datum.
6- Användaren ska kunna bjuda in andra medarbetare via applikationen.
7- Bokningens ägare ska synas på bokningsfält.
8- Bokningens titel ska synas på bokningsfält.
9- Användaren ska kunna ändra en bokad tid.
10- Användaren ska kunna radera en bokad tid.
11- Inga redundanta tider.
12- Applikationen ska vara på svenska.

3.3 Utveckling av prototyper

I början av den här delen användes resultaten av förstudiedelen och kravsamling (se figur 3.3.1) för att designa två olika digitala prototyper (se bilaga 8.2) som var bara design (innehöll inga funktioner). Under ett möte med chefen och handledaren i Omegapoint, blev en av de två prototyperna valda för utveckling. Under mötet diskuterades också de möjliga scenarierna för applikationens funktionalitet. De gav också sina åsikter för vilka som skulle passa bäst med deras önskemål. Efter att prototypen blev primärt bestämd, började själva utvecklingen.



Figur 3.3.1 Visar arbetsflödet under hela arbetet

3.4 Projektmodell

Under examensarbetets gång användes metoden Kanban för att strukturera arbetet. Kanban är en agil projektmetodik [22], som kan innebära ett visuellt processledningssystem som talar om vad som ska produceras, när ska produceras och hur mycket ska produceras.

Med Kanban behöver man inte åta sig en viss mängd arbete per iteration. Istället drar man ett begränsat antal önskemål från back log till doing-sektionen så fort det finns utrymme för mer arbete.
Önskemålen utvecklas och levereras sedan kontinuerligt. Eftersom Kanban inte föreskriver några tidsbegränsade iterationer kan leveransdatum bestämmas mer flexibelt än t.ex. med projektmodellen Scrum. Det viktigaste är att arbetsflödet visualiseras, detta genom att kolla på tre följande händelser: vad som sker nu, vad som har skett och vad som kommer ske härnäst [22].

Valet av projektmodellen baserades på:

- Kanban föreskriver inga tidsbestämda iterationer, vilket passar bra då tidplanen inte kan garanteras.
- Med Kanban behöver man inte åtaga sig en viss arbetsvolym per iteration och behöver endast begränsa Work in progress. Detta passades bra då arbetets svårigheter varierade, vilket kunde ske under arbetets gång.
- Kanban förskriver inga roller. Vilket passades bra, eftersom examensarbetet gjordes av en student så allt skulle göras av en person

Komponenterna i projektmodellerna som användes i projektmodellen:

- Kanban beskriver inga dagliga möten, detta har använts minst en gång i veckan. Under dessa möten diskuterades med handledaren vad som gjordes och vad skulle göras härnäst.
- Ingen backlog tavla gjordes. Detta beror på att arbetet utfördes av bara en person.

3.5 Dokumenteringsmetod

Metoden som användes för att hantera dokumentation under arbetsgången har varit Google Drive. Efter varje redigering så har driven uppdaterats med versions och datum.

4 Analys

I detta kapitel kommer valen att motiveras när det gäller operativsystemet, utvecklingsmiljön, databasen, designen och funktionalitet.

4.1 Liknande produkter

Som det nämndes i avsnitt 3.2.3 blev två liknande applikationer observerade. Inga av dessa två är bokningssystem, båda två är planeringskalendrar där användaren får planera sina dagar. Men de valdes för de löser nästan samma problem. Skillnaden är att de planerar för en användare och applikationen som skall utvecklas planerar för ett rum som ska användas av flera användare. Båda Tiny Calendar och PocketLife synkroniserar information med Google kalender och iOS kalender som man har i sin enhet. I båda kan finnas redundanta tider. De är gratis men innehåller en hel del reklam. Med PocketLife kan man bjuda in folk och det kan man också göra med Tiny Calendar men funktionen kostar 75kr. Med Tiny Calendar kan användaren bara ändra tiden, medan med PockelLife man kan ändra datumet och tiden. Användarna fick använda båda applikationerna och deras användning blev observerad. Denna ledde till att användarna inte ville känna sig inlösta och ville ha frihet att ändra allt. Man ska kunna klicka var som helst i kalendern och inte bara i mitten. Eftersom den ska vara ett bokningssystem, så måste användaren inte kunna välja en redan bokade tid. Alla tider sparas i en databas istället för användarens kalender.

4.2 Android vs iOS

Både Android och iOS används som OS för mobila enheter. De används mest i mobiltelefoner och surfplattor. Med hjälp av avsnitt 2.1 och 3.1 gjordes en analys för att undersöka vilket operativsystem skulle väljas först. Tabell 4.2.1 visas några av de skillnaderna.

Tabell 4.2.1 visar några skillnader mellan Android och iOS

	Android	**iOS**
Utvecklas av	Google	Apple
Källkod modell	Öppen källkod	Stängd (med öppen källkod komponenter.)
Öppen källkod	Kärnan, UI, och några av utvecklingsmiljön (såsom Android studio)	Inte öppen källkod men baserad på öppen källkod ”Darwin OS”.
passerad på	Linux	OS X och UNIX
Utvecklingsspråk	Java, C/C++	Objective-C, C/C++ och Swift
Mest kända utvecklingsmiljön	Android Studio	Xcode
CPU Arkitektur	ARM, MIPS, Power, x86	ARM
Sensate version	Marshmallow (2015-10-05)	9.3.2(2015-05-23)

4.2.1 Gränssnitt

Android och iOS använder touchskärm som har mycket gemensamt såsom: svepa, klicka och zooma. Båda operativsystemen startas från en startskärm. iOS startskärm innehåller endast rader av appikoner medan Android tillåter användning av widgets, som uppvisar automatisk uppdatering av information såsom väder. Båda användargränssnitten har ett fält där användarna kan fästa sina mest använda program.

Ett statusfält löper över toppen på både iOS och Android, som erbjuder information, tid, WiFi och batteritid. På Androids statusfält visar de nya mottagna e-post, missade samtal, meddelanden och påminnelser.

4.2.2 Säkerhet

Androidapplikationer är isolerade från resten av systemets resurser om användaren inte ger ett program tillgång till andra funktioner. Detta ökar systemets säkerhet, men kan bli lidande när en utvecklare begär onödiga behörigheter till dens applikationer. Den mest utbredda skadliga koden på Android är där textmeddelanden skickas till betalnummer utan användarens vetskap och att skicka personuppgifter till obehöriga tredje parter. Eftersom det är det mest populära operativsystemet för smartphones, är det mer sannolikt att vara i fokus för attacker.

Utvecklare av malware (Program som installeras i enheten utan användarens samtycke, dessa program kan vara spyware, virus, masker eller trojaner) är mindre benägna att skriva kod för iOS, på grund av Apples granskning av alla program och verifiering av utvecklarens identitet. Men om en iOS-enhet är jailbreakad (Apples enheter får

fullständig tillgång till filsystemet, detta i sin tur öppnar alla iOS portar) och appar installeras utanför Apples butik, kan enheten vara sårbar för attacker och skadlig kod.

4.2.3 Utveckling och publicering av applikationer

Androidapplikationer programmeras med C, C ++ och Java. Som det nämndes i tabellen ovan så är Android en öppen källkodsplattform dvs, vem som helst kan ladda ner Androids källkod och Android SDK och kompilera koden själv. Vem som helst kan skapa och distribuera Android-appar gratis. Användarna är fria att ladda ner appar utanför den officiella Google Play-butiken. Det finns dock en engångs registreringsavgift som ligger på 25$ för utvecklaren som vill publicera sina program (kostnadsfria/betalprogram) på den officiella Google Play Butiken. Appar som publiceras på Google Play genomgår en granskning av Google. Android SDK är tillgänglig för alla plattformar (Mac, PC och Linux).
iOS-appar programmeras med Objective-C. Utvecklaren måste betala 99$ varje år för att få tillgång till iOS SDK och rätten till att publicera applikationer i Apples App Store. IOS SDK är endast tillgänglig för Mac-plattformen.

Vissa utvecklingsplattformar såsom Xamarin och Cordova erbjuder ett sätt att koda en gång och omvandla plattformen till "nativa" kod som fungerar på Android, iOS och Windows plattformar. Dessa plattformar kommer att diskuteras i avsnitt 4.3.

4.2.4 Val av enhet

Ett brett utbud av Androidenheter finns på många olika prisklasser, storlekar och maskinvarufunktioner.

iOS är endast tillgänglig på Apple-enheter: iPhone som en telefon, iPad som en tablett, och iPod Touch som en MP3-spelare. Dessa enheter är i genomsnitt dyrare än motsvarande hårdvara som använder Android.

4.2.5 Val av plattform

Med hjälp av analysen som gjordes i avsnitten (4.1.1-4 .4.4) och med hjälp av enkätens resultat (se 3.2.4). I enkäten svarade 17 anställda på några frågor och resultatet visas i tabell 4.2.5.1.

Tabell 4.2.5.1 Visar enkätens resultat

Frågan	Android	iOS	Windows
Vilket operativsystem använder du?	10 medarbetare	6 medarbetare	1 medarbetare
Brukar du boka rummet?	Ja	Ja	Ja
Hur ofta brukar du boka rummet?	3-7ggr/vecka*	3-7ggr/veck*	<3
Hur lång tid brukar du boka?	30-60min*	30-60min*	30-60min*
Vilka informationer vill du se om den bokade tiden?	Namn & syftet	Namn & syftet	Namn & syftet

(*) per person

Valet blev att utveckla applikationen för Android först och detta beror på:

- Det är det mest använda operativsystemet på företaget.
- Utvecklingen för Android kan göras på en Windows-dator vilket var det enda som fanns tillgängligt för detta examensarbete.
- Utvecklingen för iOS måste ske med en Mac dator som ej fanns tillgång till.
- Tidigare erfarenhet för Android-utvecklingen.
- Androids öppna källkod hjälper att man kan hitta svar på de flesta frågor som man kommer på.

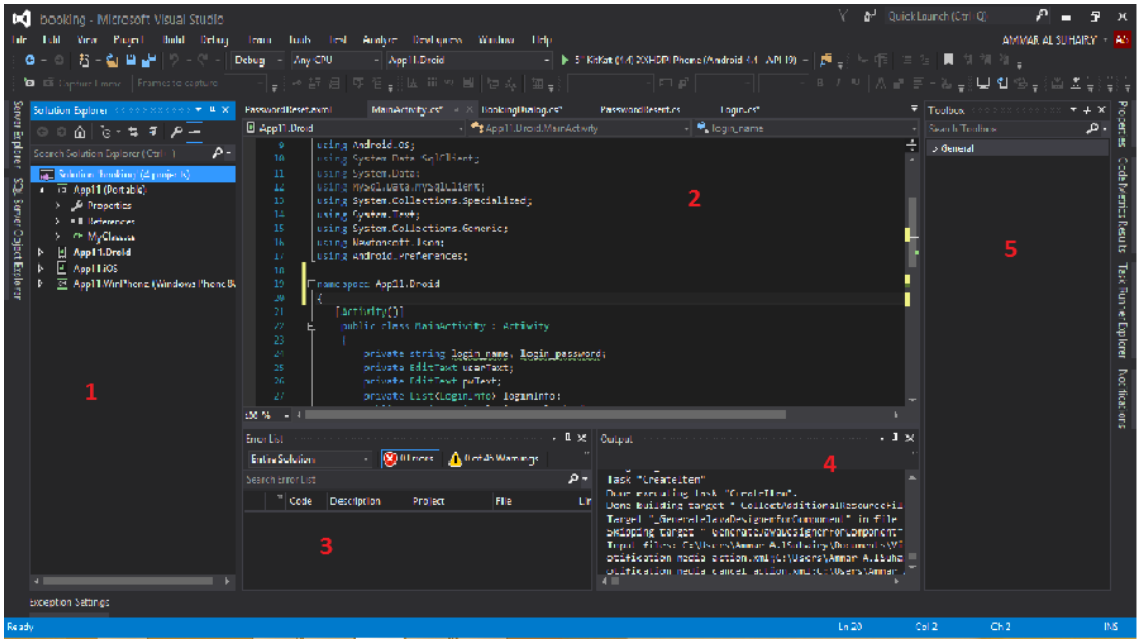
4.3 Val av utvecklingsmiljö

Som det nämndes i målformuleringen (avsnitt 1.3) ska applikationen fungera på Android och iOS. Dessutom skulle applikationen utvecklas med C# och .NET för båda operativsystemen. Detta gjorde att valet av Android studio samt Xcode omöjligt. Detta beror på att Android studio stödjer utvecklingen med Java, C och C++, medan Xcode stödjer utvecklingen med Objective C, Objective C++, C och C++. Hade examensarbetet inte varit begränsad för just C#, då hade applikationen kunnat utvecklas med Android Studio och Xcode med hjälp av programmeringsspråk C++ [12][14].

Med hjälp av dessa källor [12, 13, 23 och 24] hittades en hel del resultat som handlar om Xamarin med Xamarin studio och Visual studio, och det visade sig att båda två handlar om en och samma sak. Vilket är utveckling av nativa applikationer för Android, iOS och Windows med C# som programmeringsspråk.

4.3.1 Xamarin i Visual Studio

Som det nämndes i avsnitt 2.3.2, Xamarin är en utvecklingsplattform för mobilapplikationer. Med Xamarin kan man utveckla nativa applikationer för Android, iOS och Windows genom att använda en gemensam C# kodbas (se figur 4.3.2.1). Där återanvändning av koden kan uppnå till mer än 75 % mellan de olika plattformarna. Xamarin stödjer 100 % av API täckningen, detta innebär att alla API-funktioner som kan vara tillgängliga via nativa utvecklingen, kan vara tillgängliga via Xamarin och C# [22]. Denna bror på att Xamarin skapar en nativ bindning/mappning mellan den delade C# biblioteket och det nativa biblioteket, vilket leder till att varje plattfrom kan använda sina egna API-funktioner oberoende av andra plattformar [22]. Figur 4.3.2.1 visar hur Visual studio ser ut.



Figur 4.3.1.1 visar hur Visual studio ser ut. Nr 1 visar det delade biblioteket, där alla tre pattformarna ligger under ett projekt. Nr 2 visar kodeditorn. Nr 3 visar den delen som visar fel och varningsmeddelanden, Nr 4 visar information vid exekvering. Nr 5 visar toolboxen som kan användas för just den plattformen.

4.3.2 Xamarins sätt att dela kod

Det finns två olika sätt för att dela koden mellan de tre olika operativsystemen. Först genom att Dela projekt, man kan använda typen ” Shared Asset Project ” för att organisera källkoden genom att använda #if-direktiv till kompilatorn som krävs för att hantera plattformsspecifika krav [14].
Fördelarna med detta är:

- Möjligt att dela koden över de delade projekten.
- Kod som är utvecklad för en viss plattform kan återanvändas för andra plattformar.

Nackdelarna:

- Man måste uppdatera alla de berörda funktioner, om man uppdaterar en funktion till exempel genom att byta namn på den så måste all kod som använder denna funktion också uppdateras.
- Den ger inte någon assembly som output eftersom hela projektet bara kommer att ha en DLL-fil.

Det andra sättet är genom att använda ”Portable Class Libraries”(PCL) för de plattformar som applikationen skall stödja.
Fördelarna med detta sätt är:

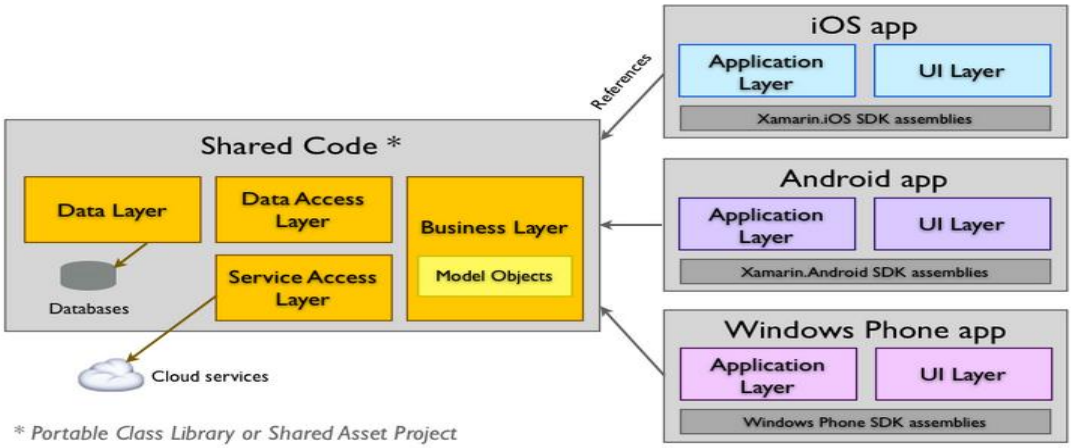
- Det är också möjligt att dela koden över flera projekt.

- Om man uppdaterar en funktion så blir alla delar av koden som använder den funktion uppdaterades.

.

Nackdelarna:

- Direktiv till kompilatorn kan inte används.
- PCL typen är bara tillgänglig via Visual Basic



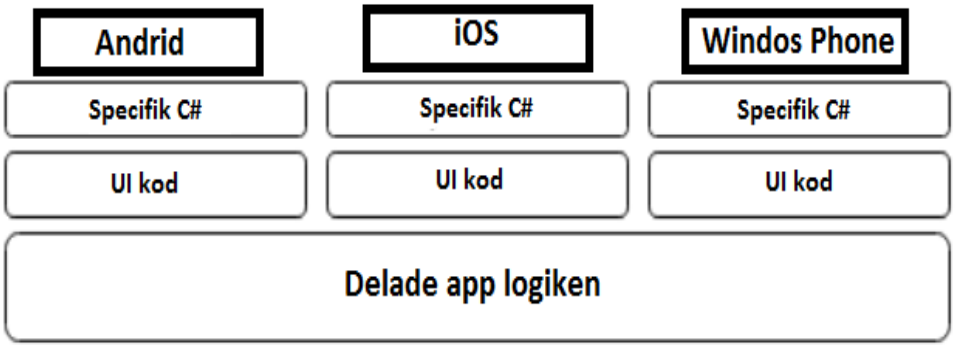
Figur 4.3.2.1 visar hur man kan använda strategier för att dela kod i flera plattformar [14].

Det sättet som Xamarin producerar nativa applikationer skiljer sig mellan plattformarna. Som t.ex. på iOS ahead-of-time (AOT) kompilator från Mona Framwork kompilerar Xamarin.iOS applikation och producerar en nativ ARM binär som sedan kan köras på iOS-plattformen. De .NET Framework som ingår i applikationen, men oanvända skalas bort för att minska filstorleken [13].

På Android Xamarin.Android program kompileras till Comman Intermediate Language (CIL) och oanvända klasser från .NET Framework skalas bort också. När Programmet startar Just-In-Time (JIN) kompilering används för att producera en nativ assambly [13].

4.3.3 Fördelar med Xamarin

- Xamarin tillåter att skriva kod för applikationer som fungerar på flera plattformar i ett enda språk. Vilket sparar mycket pengar och tid genom att utvecklarna använder sin kunskap i C# för att utveckla mobilapplikationer istället för bara web och Windowsapplikationer.
- Nativa gränssnitt. Det är inte hela koden man skriver med Xamarin som kan delas mellan plattformarna, vilket ger en stor fördel (Se figur 4.3.3.1). Detta beror på att den delen som inte kommer att delas, är den delen som kommer att garantera att applikationerna har det unika och nativa beteende som krävs för plattformen [13].



Figur 4.3.3.1 delade koden mellan plattformarna

- Alltid up-to-date med de senaste nativa API:ar. Xamarin ger en API som speglar de nativa API:ar och de spelglade API:ar är så bra gjorda så att man kan be om hjälp från iOS eller Android utvecklare i Java eller Obejtiv C och lätt omvandla koden till C#.
- Xamarin har nyligen (sedan 2016-03-31) blivit öppen källkod under MIT-licens, gratis och tillgängligt i Visual Studio Community Edition, som är gratis för enskilda utvecklare, open source-projekt, akademisk forskning, utbildning och små professionella team. Visual Studio Community Edition är inte tillgänglig för utvecklarna som använder Mac, men Xamarin löste det genom att kan göra den nyutvecklade Xamarin Studio Community Edition gratis [24].

  Detta kommer leda till att användning av Xamarin kommer att öka mycket med tanke på det tidigare priset på $299/år för att använda den i Visual Studio Community Edition

4.4 Val av databas

I detta stycke diskuteras valet av databasmodell samt verktyget som har använts under examensarbetet.

4.4.1 Objektorienteraddatabas vs Relationsdatabas

- Objektorienterad databas är sätt att lagra data på, med objektet i centrum. Skillnaden mellan objektorienterad databas och relationsdatabas är att den sistnämnda databasen har data i fokus istället för objekt.
- Relationsdatabassystem är baserade på tvådimensionella tabeller där varje objekt visas som en rad. Förhållanden mellan datan som uttrycks genom att jämföra de värden som är lagrade i dessa tabeller. Ett språk som SQL tillåter tabeller som skall kombineras i farten för att uttrycka relationer mellan data. Objektmodellen bygger på den täta integrationen av kod och data, flexibla datatyper, hierarkiska relationer mellan datatyper och referenser.
- Relationsdatabassystem är bra för att hantera stora mängder av data, medan objektorienterade programspråk är bra på att uttrycka komplexa relationer

mellan objekt. RDBMS är bra för datahämtning men ger lite stöd för datamanipulation. Objektorienterade programmeringsspråk är utmärkta på datamanipulation men ger lite eller inget stöd för databeständighet och hämtning.

- Objektorienterad databas saknar standard och av denna anledning finner man många brister i detta system jämfört med relationsdatabas.
- Prestanda i objektorienterad databas när stora mängder objekt ska läsas är lägre än relationsdatabasens prestanda.
- Det är smidigare för programmeraren att använda sig av en objektorienterad databas eftersom man kan använda ett gemensamt programmeringsspråk som Smalltalk,C++ och Java. Vilket programmeraren inte kan göra vid användning av RDBMS, då måste man använda ett frågespråk såsom SQL.
- Tabell 4.4.1.1 visar flera skillnader mellan de två modellerna.

Tabell 4.4.1.1 Mer skillnader mellan Relations- och Objektorienteradsdatabaser

	Objektorienterade databaser	**Relationsdatabaser**
Verktyg	- Saknar mognad	- Schemadesignverktyg, kodgeneratorer, användargränssnittsverktyg
Interaktion mellan applikation och databas	- Persistent programmering innebär att det inte finns någon separation mellan databasen och applikationen	- Mur mellan applikationsprogrammet och databassystemet
Objektidentifierare	- Objektidentifierare - Komplex datastruktur	- Primärnyckel - Information om objektet spritt över olika tabeller
Design av databasen	- Relationer genom relationsegenskaper eller referensattribut - Verkliga relationer - Arv stöds i modellen	- Relationer genom attribut med matchande värden - Join-operationer - Arv finns ej i modellen
Prestanda	- Bättre prestanda vid applikationer som hanterar komplexa objekt - Frågeoptimering väldigt komplext	- Mindre effektivt - Bättre prestanda när stora mängder objekt ska hämtas och undersökas sekventiellt
Lättanvändbarhet	- Inlärningskurvan lägre	- Lättare att lära sig - Lätt att förstå

Efter att man fördjupat sig i och studerat fördelarna samt nackdelana mellan de två modellerna så visade det sig att Relationsdatabas modellen passar bäst för det här arbetet. Detta beror på att applikationen inte ska innehålla något komplext, tidigare erfarenhet av MySQL, behovet av att identifiera data med primärnyckel så att datan kan spridas över olika tabeller.

4.4.2 Databashanterare

Vid bedömningen av vilket verktyg som ska man använda för datahantering så är de mest populära valen är MySQL och SQL Server. Båda två är RDBMS, båda är effektiva på att hålla data organiserade, båda är enkla att använda via ett användargränssnitt och båda två lagrar data i form av tabeller.

Det gjordes en jämförelse mellan de två för att se skillnaderna mellan dem. MySQL är mer inriktad mot datahantering så att dessa data kan visas, uppdateras och sparas igen. MySQL är svagare när det gäller att sätta in och ta bort data, detta bror på att MySQL saknar en del av avancerade funktioner som SQL Server har. Flera skillnader i tabell 4.4.2.

Tabell 4.4.2.1 skillnader mellan MySQL och SQL server

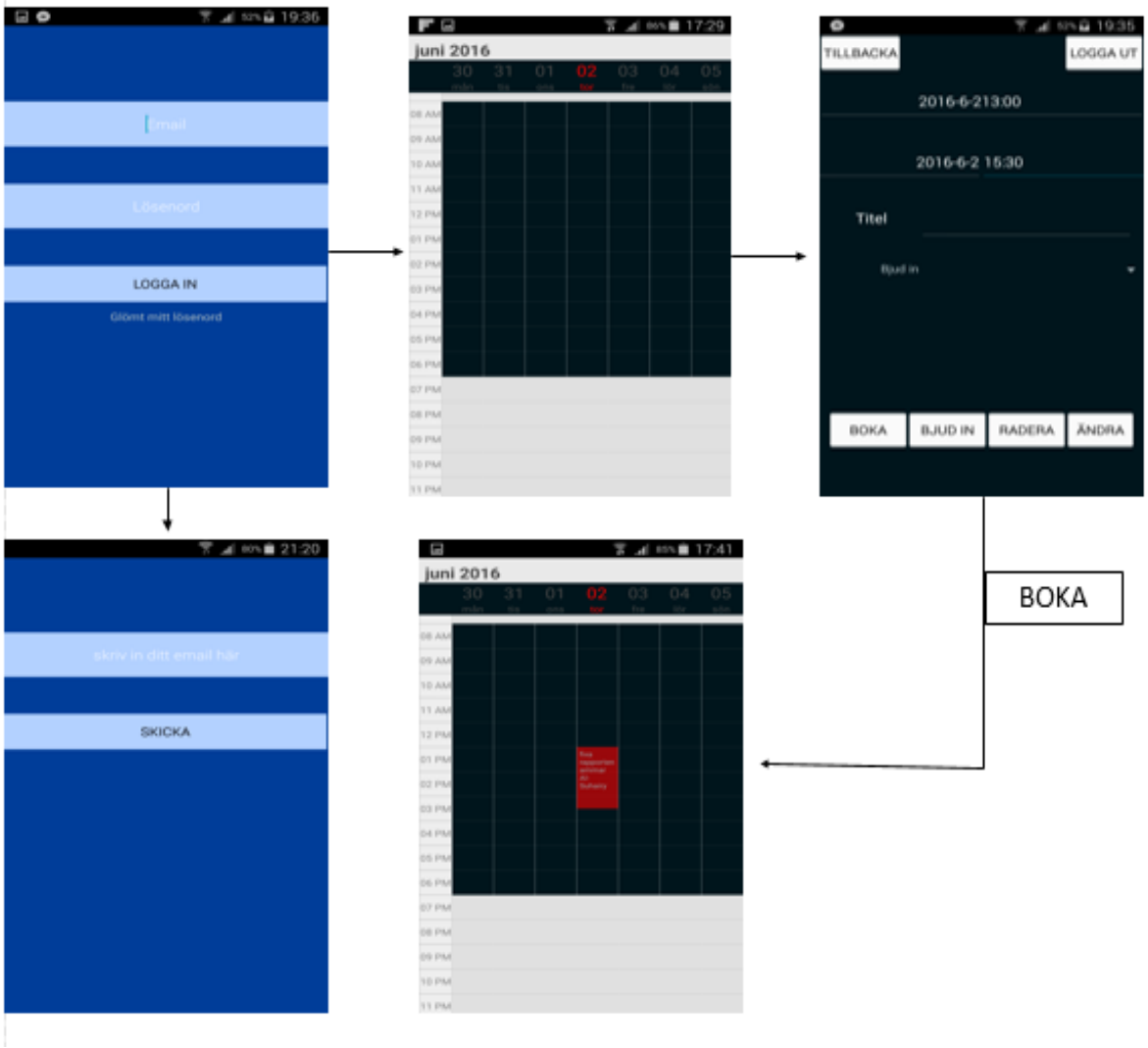
	MySQL	**SQL Server**
Operativsystem	Windows, Linux, OS X	Windows
Språkstöd	Java, C, C++, C#, ObjC	Java, C#, VB, Paython, .NET
Kostnaden	Gratis	$3717
Gränssnitt	SQL	SQL, GUI
Licens	Öppen källkod	Stängd källkod

Valet blev att använda MySQL som databashanterare och detta beror först och främst på tidigare erfarenhet av MySQL då man arbetade med MySQL under en kurs som relaterade till utbildningen. Utvecklingen sker med C# som programmeringsspråk, vilket MySQL stödjer. MySQL är öppen källkod, vilket hjälper att hitta svar på dem flesta problem som kommer att uppstå under arbetet, dessutom gratis.

4.5 Prototyp

Som det nämndes i avsnitt 3.3 att under ett möte med chefen och handledaren i Omegapoint skulle väljas mellan prototyp1 och prototyp2 (se bilaga 8.2). Valet blev prototyp2 och detta beror på att den ger mer frihet till användaren. Då användaren kan ändra tiden medan med prototyp1 kan användaren inte ändra tiden, då användaren klickar direkt på den tiden som han vill boka. Under mötet bestämdes att applikationen ska vara på svenska och att användarna skulle kunna ändra datumet istället för bara tiden så att användarna har mer frihet, istället för att svepa till det datumet som användaren vill boka rummet i.

Figur 4.5.1 visar flödesdigrammet för den prototypen som utvecklades. Pilarna visar nästkommande sida. Beskrivning för varje sida och hur man kommer till nästa sida kommer på nästa kapitel.



Figur 4.5.1 Flödesdiagrammet för prototypen som skulle utveckla

4.6 Databasens struktur

Databasen som används är uppbyggd med MySQL och PHP. Databasen ligger på ammar.one och består av tre klasser, klassen Profiles innehåller användarnas information, där det finns en rad för varje användare. Klassen bokedTimes innehåller de bokade event, en rad för varje event. Klassen InvitedMembers innehåller de inbjudna anställda, en rad för varje event. För att se vad varje klass består av se appendix 8.1.

4.7 Användartester

I detta avsnitt kommer tester att presenteras. Testerna utfördes först när applikationen blev körbar. Resultatet användas för att kontrollera att funktionerna ger rätt svar, samt för att förbättra prototypen. Användarens upplevelse står i centrum och målet är att användaren känner sig bekväm med applikationen och att den inte bjuder på några överraskningar.

Inloggning. Applikationen är installerad på en Androidmobil. Användaren har en existerande inloggning. Användaren startar applikationen genom att klicka på applikationens ikon. Först möts användaren av en splashsskärm med företagets logo och därefter visas inloggningsskärmen.

Återställ lösenordet. Användaren har ett existerande emailkonto. Användaren klickar på ”Glömt Mitt lösenord”. Användaren kommer att presenteras med en sida där användaren får skriva sitt email, om mailet redan finns i databasen då får han sitt lösenord via mail.

Bokning. Användaren loggar in. Efter en lyckad inloggning möts användaren med en veckokalender. Användaren klickar på en tid och då presenteras användaren med bokningsskärmen. Användaren kan kontrollera att han kan ändra datum, tid, skriva en titel, klickar på bjud in knappen och bjuda in andra medarbetare. Användaren kommer att presenteras av att välja från vilket email han vill skicka inbjudandet från om användaren har valt att bjuda in andra medarbetare. Annars blir tiden bokad om tiden är tillgänglig och veckokalenderskärmen visas.

Tidsändring. Användaren är inloggad. Användaren har en bokning. Användaren försöker klicka på en tid som är bokad av en annan användare då får användaren ett felmeddelande som ser ut så här ”*Den här tiden är inte tillåten*”. Användaren klickar på en ej bokad tid, han kommer fram till bokningsskärmen. Användaren försöken ändra en ej bokad tid då får användaren detta felmeddelande ”*Du kan inte ändra den här tiden*”. Om användaren klickar på en tid som tillhör honom. Han kommer till bokningsskärmen. Då kan användaren ändra datumet, tiden, titeln, bjuda in flera medarbetare. Väljer man en tid som inte är tillgänglig då får användaren detta felmeddelande ” Den här tiden är inte tillgänglig”. Annars får man en alert som ser ut så här ” Är du säker på att du vill ändra din bokning från” den gamla bokningen ” till ”den nya bokningen”. Svarar användaren med nej så det händer ingenting. Svarar användaren med ja så ändras bokningen och man går åter till huvudskärmen som visar alla bokningar. Användarens

upplevelse står i centrum och målet är att användaren känner sig bekväm och inga konstigheter.

4.8 Källkritik

I detta avsnitt presenteras källorna som har använts under arbetet. Med hjälp av [25] kunde följande frågor ställas för varje källa som är hämtad från Internet

- Vem som står bakom källan?
- Vad är syftet med källan?
- Vem är källan inriktat till?
- Kan källan vara trovärdigt?

Referens [1] är en artikel på www.dagensanalys.se som är skriven av Bartosz Podniesinski. Han har många tekniska artiklar och många citat hittades till hans artiklar. Källan kan anses trovärdlig.

De källor som användes till mobila plattformar (Android, iOS och Windows), utvecklingsmiljöer (Android Studio, Xcode, Visual Studio, Cordova och Xamarin) och databas hanterare (MySQL och SQL server) [källanr. 2-15,18, 19 och 21] är upphovsmännens egna hemsidor. De används för att hänvisa användarna, fånga nya användare, ge tillräckliga informationer om produkterna. Dessa källor kan anses trovärdiga eftersom de har ett stort egenintresse av att lämna korrekt teknisk information till utvecklare.

Källan som användes till databasen [16] är skriven av Thomas Padron-McCarthy. Han är en lärare på Örebro universitet och författare till boken (Databasteknikbok studentlitteratur, ISBN 91-44-04449-6). Hemsidan är sammanfattning av hans bok. På grund av författarens yrke och att hans lärobok används i kurser vid universitet kan denna källa anses trovärdig.

Källan Database System Concepts Sixth edition 2010 ISBN 0-07-352332-1 till McGraw-Hill [17]. Boken kan anses trovärdig eftersom boken utgiven av ett ansett förlag och används som kurslitteratur.

Källan Kanban och Scrum få det bästa av två världar [22] Henrik Kniberg & Mattias Skarin. Boken kan anses trovärdig eftersom boken är utgiven av ett ansett förlagoch används som kurslitteratur.

Xamarin Mobile Application Development: Cross-Platform C# and Xamarin.Forms Fundamentals [23]. Källan användas mycket under inlärningsprocessen för att den beskriver Xamarin och hur man använder den på ett bra detaljerad sätt. Källan kan anses trovärdig eftersom boken är mycket omfattande och används som kurslitteratur. Boken rekommenderas dessutom av Xamarin.

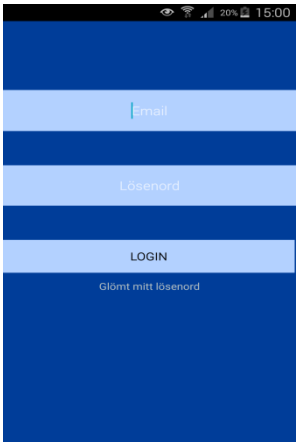
Youtube kanaler [26, 27] som användes under arbetet. De kan räknas som trovärdiga och detta beror på att den ena tillhör själva Xamarin, där deras medarbetare beskriver Xamrin, förklarar fördelarna och visar hur man kan använda verktyget. Den andra knallen tillhör en programmerare som förklarar med exempel hur man kan skriva olika koddelar med Xamarin. Det som gör att denna källa trovärdig är att han kör koden direkt och med mina tester stämde överrens med hans lösningar.

5 Resultat

I detta kapitel kommer resultatet av detta examensarbete att presenteras. Arbetet ledde till att en Androidapplikation utvecklades. Applikationen är ett bokingsystem, där Omegapoint anställda kan boka och avboka konferensrummet.

5.1 Inloggningsskärm

Efter att applikationen har startas, kommer användaren till inloggningsskärmen se figur 5.1.1. Där kan man skriva in sitt email, lösenord och klicka på Login knappen. Under Login knappen finns ”Glömt mitt lösenord” om man klickar på denna text kommer man att presenteras av lösenord återställningsskärm. Om man klickar på login och inloggningen lyckades då kommer man presenteras för veckokalendarsskärm. Annars får man ett felmeddelande som ser ut såhär ”Var god och kontrollera ditt email och lösenord”.



Figur 5.1.1 Inloggningskärm

Inloggningen sparas tills man själv loggar ut. Så att man inte behöver loggar in varje gång man vill använda applikationen. Denna görs med hjälp av **IsharedPreferences** som sparar variablensvärde se figur 5.1.2. Vid inlogging sparas variablen hasLoggedIn som true och vid utloggning sparas variablen som false. Varje gång applikationen startas, kontrolleras värdet på hasLoggedIn och om den är true så kommer användaren direkt till veckokalendersskärmen, annars kommer användaren till inloggningsskärmen.

```
void client_UploadValuesCompleted(object sender, UploadValuesCompletedEventArgs e)
{
   RunOnUiThread(() =>
    {
    string json = Encoding.UTF8.GetString(e.Result);    // The result from database
    if(json.Equals("Login Failed...Try Again"))         // If user cound be found
    {
        Toast.MakeText(this, "Var god och kontrolle ditt email och lösenord", ToastLength.Short).Show(); // message to show
    }
    else
    {
        loginInfo = JsonConvert.DeserializeObject<List<LoginInfo>>(json); // Deserialize the JSON to a .NET object.
        int newID = 0;
        string id = loginInfo[0].Count;   // How many persons could be found in the database
        loginName = loginInfo[0].Id;      // Ursername for this user who want to sign in.
        loginId = loginInfo[0].Name;      // UserId.
        int.TryParse(id, out newID);      // convert the id form string to int.
        if (newID == 1)                   // checking if we could find just one user.
        {
                ISharedPreferences prefs = PreferenceManager.GetDefaultSharedPreferences(this);   //Gets a SharedPreferences instance
                                                                                                  // that preferences managed by this will use.
                ISharedPreferencesEditor editor = prefs.Edit();  // Editor to save the values
                editor.PutBoolean("hasLoggedIn", true);   // hasloggedIn will be true.
                editor.PutString("userId", loginId);      // Save the userEmail
                editor.PutString("userName", loginName);  // Save the username
                editor.Apply();
                Intent intent = new Intent(this, typeof( Login ));   // start the activity
                this.StartActivity(intent);
            }
           else
           {
                Toast.MakeText(this, "Var god och kontrollera ditt email och lösenord", ToastLength.Short).Show();
           }
```

Figur 5.1.2 Visar en del av koden som användes för inloggningen.

5.2 Lösenord återställningsskärm

Ifall man inte kommer ihåg sitt lösenord och detta kan lätt hända på grund av att användaren inte behöver logga in varje gång användaren vill använda applikationen. I så fall kan man klicka på texten ”glömt mitt lösenord” och då kommer man att presenteras för denna skärm se figur 5.2.1. Där det står skriv in ditt email, kan man ange sitt email. Ifall man anger ett existerande email, då får man detta meddelande *”du kommer att få ett mail med ditt lösenord inom kort* ” och lösenordet skickas med ett mail till detta existerande email. Annars får man detta fel meddelande*” Var god och kontrollera att du anger ett korrekt email! Prova igen!”.*
Återställningen sker via PHP skriptet. Se figur 5.2.2

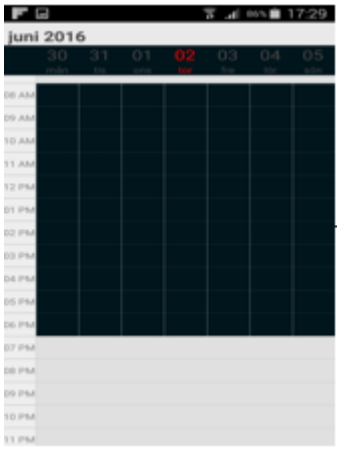
Figur 5.2.1 återställningsskärm

```php
<?php
require"init.php";
$mail=mysqli_real_escape_string($con, $_POST["userEmail"]);
$sql_query= "SELECT password FROM user_info WHERE mail = '$mail' ";
$result=mysqli_query($con,$sql_query);

$count=mysqli_num_rows($result);

if($count!=0)
{
        $pass  =  $rows['password'];
        $to=$mail;
        $subject= "Ditt lösenord";
        $body= $pass;
        $additionalheaders = "From: <[redacted]>rn";
        $rows=mysqli_fetch_array($result);

        mail($to,$subject,$body,$additionalheaders);
}
        else{
        echo "Your Email is not corecct...";
        }

?>
```

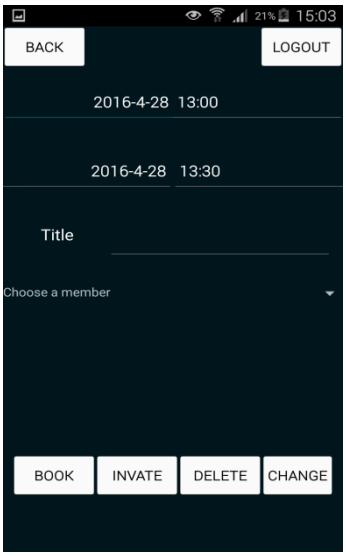
Figur 5.2.2 Visar PHP koden för lösenordsårterställningen

5.3 Bokning

Vid en lyckad inloggning, kommer användaren att mötas av en veckokalender med dagens datum se figur 5.3.1. De svartfärgade tider motsvarar arbetstiderna. Där kan man svepa till höger om han vill komma till föregående veckor, svepa till vänster om han vill komma till nästkommande veckor och svepa neråt om han vill komma till tidigare tider. Användaren kan se de redan bokade tiderna. Tiderna som är redan bokade visas med röd färg, namn på den som äger bokningen och syftet med bokningen. Syftet är dock inte obligatoriskt. Ifall man klickar på en tid som är redan bokad av **en annan person**, då kommer man mötas av detta fel meddelande ”*Den här tiden är inte tillgänglig*”. Annars kommer man mötas av skärmen nedan (Figur 5.3.2).

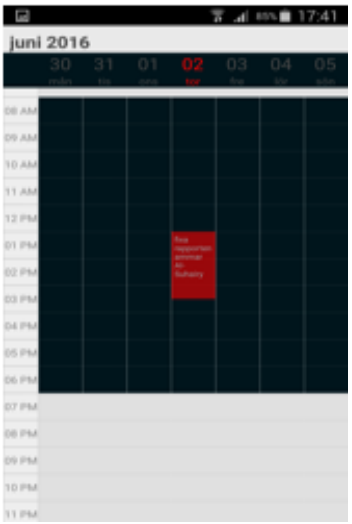


Figur 5.3.1 veckokalendersskärm



Figur 5.3.2 bokmingsskärm

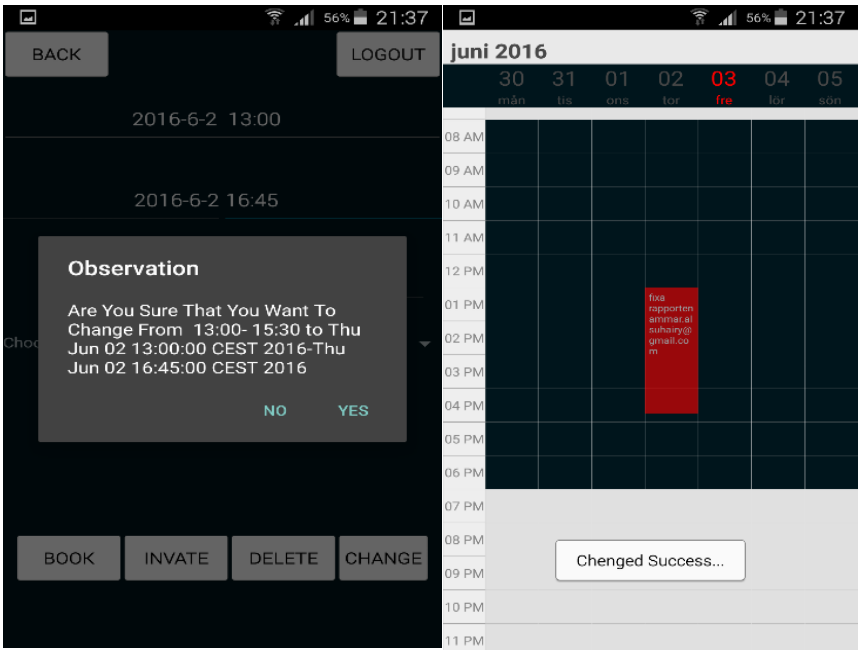
Efter att användaren har klickat på en tid, så kommer han till denna skärm, se figur 5.3.2. Här kan man ändra datumet och tiden genom att klicka på datum- och tidfälten, skriva en titel på fältet där det står titel. Användaren kan bjuda in andra medarbetarna till den tiden som han vill boka, genom att klicka på bjud in knappen därefter får han välja vilka som han vill bjuda genom att klicka på välj namn som visas efter att man har klickat bjud in knappen. Därefter klickar man på boka knappen om tiden som vill bokas är inte tillgänglig då visas detta fel meddelande ”*Den här tiden är inte tillgänglig*”. Annars blir tiden bokad och användaren hänvisas till att skicka ett mail till dem som blev inbjudna. Ifall användaren inte vill tillägga något till mailet så får han bara klicka på skicka. Därefter visas veckokalendarsskärmen med den bokade tiden. Se figur 5.3.3



Figur 5.3.3 Visar veckokalendersskärmen efter en bokning

5.4 Tidsändring

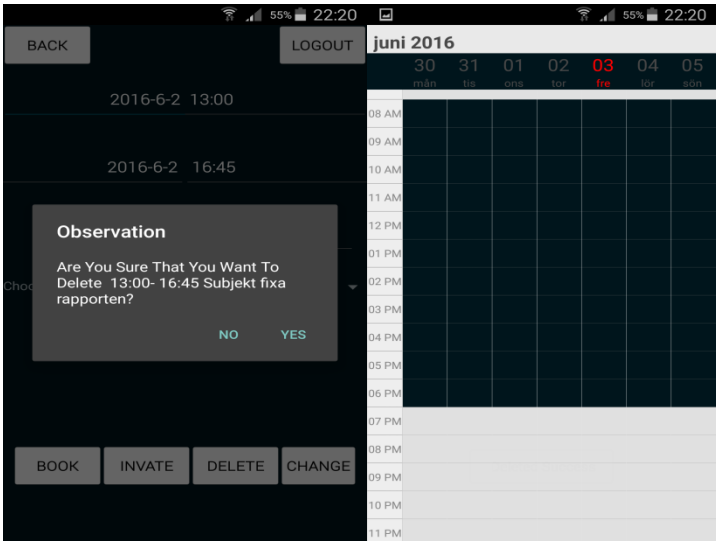
Om man vill ändra en tid, då får man klicka på den tiden som man vill ändra. Äger man inte den tiden? Då får man ett fel meddelande som ser ut såhär ”Du äger inte den här tiden”. Annars kommer man till figur 5.3.2. Då kan man ändra datumet, tiden, titeln, bjuda in flera medarbetare. Väljer man en tid som inte är tillgänglig då får man ett felmeddelande ” Den här tiden är inte tillgänglig”. Annars får man en alert som ser ut så här ” Är du säker på att du vill ändra din bokning från ”den gamla bokningen ” till ”den nya bokningen”. Svarar man med nej så det händer ingenting. Svarar man med ja så ändras bokningen (se figur 5.4.1) och man går åter till huvudskärmen som visar alla bokningar (veckokalenderskärmen).



Figur 5.4.1 visar observationsmeddellande och klalendern efter ändring

5.5 Tidsradering

Om man vill radera en tid, då får man också klicka på den tiden som man vill radera. Exakt på samma sätt med tidsändring så får man fel meddelande om man inte äger den tiden annars kommer man till bokningsskärm. Man klickar på *Radera knappen* och därefter får man en alert som ser ut såhär ” *Är du säker på att du vill radera den tiden*?” svarar man med nej, då sker det ingenting, svarar man med ja så raderas tiden (se figur 5.5.1) och ett mail skickas till dem som var inbjudna. Där det står att eventet är inställt.



Figur 5.5.1 visar observationsmeddelande och klalendern efter radering

5.6 Testresultat

I detta avsnitt presenteras resultaten av användartester (avsnitt 4.7).

Inloggningen
Vissa användare var osäkra om de hade klickat på inloggningsknappen eller inte. För att förbättre lösningen valdes att lägga en progress bar som blir synligt när man klickar på knappen och fortsätter att vara synlig tills man kommer till nästa skärm.

Bokning
Inga problem hittades

Tidsändring
Vid tidsändring skickas inga nya email till de som var inbjudna till eventet. Problemet löstes med en funktion som hämtar alla email som sparades i bokningen via bokningsId och skicka ett email igen.

Tidsradering
Inga problem hittades

6 Slutsats

I detta kapitel presenteras de slutsatser som examensarbetet ledde till.

6.1 Svar på examensarbetets problemformuleringar

Med hjälp av analysen kunde följande problem besvaras. Svaren användes för att hitta vilka funktioner som användarna ville ha i applikationen, vilken databas samt databashanterare passar för detta examensarbete. Dessutom hitta den lämpliga plattform för utvecklingen.

Vilka system finns på marknaden och hur med hjälp av analysresultaten hitta de funktioner som användarna är nöjda eller inte nöjda med?

Applikationerna som analyserades var Tiny Calendar och PocketLife. Användarna var inte nöjda med att vara inlösta med att välja, ändra, radera tiderna och dessutom de gillade inte att de själv skriver eller söker efter medarbetarnas mailadressar bland alla kontakter som finns i deras mailkonto. Vilket användaren måste göra med båda applikationerna.
Denna ledde till att användarna fick mer frihet med att kunna välja, ändra, radera en tid.
Fördelarna med Tiny Calendar och PocketLife är:

- De synkar information med Google och iOS kalenderna.
- De stödjer bokning, avbokning och ändring.
- De skickar notiser för att påminna användaren innan eventet.

Nackdelarna:

- De kan finnas redundanta tider.
- Användaren kan inte klicka var som helst i kalendern.
- De innehåller en del reklam

För mer detaljerade svar se avsnitt 4.1.

Vilka metoder kan användas för att analysera Omegapoints anställdas behov?

Metoderna som användades för att analysera användarnas behovs var:

- Intressantanalys
  Denna metod användes för att hitta de mest intressanta intressenter, vilka var Omegapoints anställda. För mer detaljerade svar se avsnitt 3.2.1.
- Observationer
  Denna metod användes för att studera befintliga bokningssystem samt användarnas beteende vid testning av applikationen. För mer detaljerade svar se avsnitt 3.2.3.
- Intervjuer
  Intervjuer med anställda var den mest använda metoden under detta examensarbete. Den användes för att hitta användarnas behov, vilka funktioner som skulle passa bäst och vilken prototyp skulle utvecklas. För mer detaljerade svar se avsnitt 3.2.2.

- Enkät
  Denna metod användes för kontrollera vilket operativsystem som används mest av Omegapoints anställda. För mer detaljerade svar se avsnitt 3.2.4 & 4.2.5.

Med hjälp av alla dessa metoder tillsammans samlades information in och användes till kravspecifikation för applikationens beteende. Vilket var en grund till utvecklingen.

Vilken plattform applikationen skall utvecklas på?

Valet blev Xamarin med Visual Studio som utvecklingsmiljö. Detta beror på resultatet av undersökningen och analysen visade sig att Xamarin stödjer utvecklingen för mobilapplikationer med C# på ett bra sätt och tillgång till Visual Studio då den finns i en gratis version. För mer detaljerade svar se avsnitt 4.2.

Vilken databas kan vara lämplig för att spara information från bokningssystemet?

Valet blev relationsdatabas med MySQL som databashanterare och PHP som frågaspråk. Denna bror på att relationsdatabas har data i fokus och är bra för att hantera stora mängder av data. Relationsdatabas är bra för datahämtning.
Relationsdatabassystem är baserade på tvådimensionella tabeller och ett språk som SQL tillåter tabeller som skall kombineras i farten för att uttrycka relationer mellan data. För detta användes MySQL och detta beror på att MySQL är öppen källkod, gratis, tidigare erfarenhet för den. För mer detaljerade svar se avsnitt 4.4.

6.2 Övriga slutsatser

Syftet med detta examensarbete har varit att utveckla en Android & iOS mobilapplikation för Omegapoints bokningssystem med tillhörande backend med databas. Detta mål kunde nås delvis, eftersom som det nämndes i avsnitt 4.2 att utvecklingen för iOS inte kan ske utan Mac, vilket inte fanns tillgång till. Tack vare det agila arbetssättet, då har man kunnat ändra och passa sig efter ändringen. Om en sekventiell modell hade använts, så hade det varit svårt att ändra planeringen på grund av ändringsprocessen i vattenfallmodellen. En annan sak är att utvecklingen hade varit långsammare och produkten inte hade liknat den producerade produkt.

Produkten som har utvecklats följer Omegapoints behovs och önskemål. Vilka är att de skulle kunna skapa ett event, ändra, avboka och bjuda in andra medarbetare till eventet. Bokningen sker genom att användaren direkt klickar på den tiden som användaren vill boka eller klickar vart som helst i kalendern, därefter ändrar datumet och tiden i datums- & tids fält.
Den bokade tiden blir rödfärgad och användarens namn samt eventets titel visas. Detta gör att eventets ägare identifierat för de andra medarbetarna, så att de kan kontakta eventets ägare ifall de måste boka konferensrumet till något som är viktigare.

Med hjälp av användertesterna i avsnitt 4.7 utvärderades användbarheten av applikationen samt ömsesidig påverkan mellan applikationen och användarna. Denna

gjordes för att detektera och åtgärda fel och kontrollera att funktionerna gör vad som förväntas av dem.

6.3 Framtida utvecklingsmöjligheter

Arbetet lämnar stora möjligheter till vidareutveckling av Omegapoint och det som kommer att ske härnäst är att koppla applikationen till en Exchange server, så att bokningskalender och användarens kalender synkroniserar information till varandra. Detta leder till att de event som användaren skapar i applikationen transporteras till användarens egen kalender.

Det finns stora möjligheter att göra applikationen mer värdefull också såsom:

- Utveckla applikationen för andra plattformer såsom iOS och Windows, så att alla anställda kan använda applikationen.

- Utveckla applikationen så att den skickar push-notifikationer för att påminna användaren innan eventet äger rum på en vis tid.

- Utveckla applikationen så att den stödjer bokningen för flera rum.

- Översätta applikationen till engelska.

7 Källförteckning

1- Windows på mobile halkar ner till en tredjeplats (2016-04-05) http://www.dagensanalys.se/2016/04/windows-pa-mobilen-halkar-ner-till-en-tredjeplats/ [2015-05-13]
2- Andoid history http://www.androidcentral.com/android-history [2015-03-17]
3- Android Open Source Project http://source.android.com/ [2016-03-17]
4- Android studio https://developer.android.com/studio/index.html [2016-03-18]
5- Android NDK https://developer.android.com/ndk/guides/index.html [2016-03-21]
6- Vad är iOS http://www.apple.com/se/ios/what-is/ [2016-05-29]
7- Start Developing iOS Apps (Swift) https://developer.apple.com/library/ios/referencelibrary/GettingStarted/DevelopiOSAppsSwift/index.html?utm_source=statuscode&utm_medium=email [2016-05-29]
8- iOS Developer Library https://developer.apple.com/library/ios/documentation/General/Conceptual/DevPedia-CocoaCore/Cocoa.html [2016-05-29]
9- Microsoft Developer Tools https://www.microsoft.com/en-us/download/developer-tools.aspx [2016-05-03]
10- Android Developer Tools. https://developer.android.com/tools/help/adt.html [2016-05-24]
11- Apple Developer Tools https://developer.apple.com/xcode/ [2016-05-25]
12- Visual Studio Developer Tools [https://www.visualstudio.com/en-us/features/modern-web-tooling-vs.aspx [2016-04-15]
13- Xamarin https://www.xamarin.com/ [2016-04-15]
14- Xamarin Sharing Code Options https://developer.xamarin.com/guides/cross-platform/application_fundamentals/building_cross_platform_applications/sharing_code_options/ [2016-05-28]
15- Apache Cordova https://cordova.apache.org/ [2016-05-28]
16- Grunder om databaser https://www.databasteknik.se/webbkursen/[2016-05-22]
17- Database System Concepts Sixth edition 2010 ISBN 0-07-352332-1 McGraw-Hill
18- Oracle > MySQL http://www.oracle.com/us/products/mysql/overview/index.html [2016-06-05]

19- SQL server Developer Network
https://msdn.microsoft.com/en-us/library/hh272686(v=vs.103).aspx [2016-06-05]
20- https://sv.wikipedia.org/wiki/PHP [2016-06-06]
21- Essential Studio for Xamarin
https://www.syncfusion.com/company/about-us [2016-06-01]
22- Kanban och Scrum få det bästa av två världar
Henrik Kniberg & Mattias Skarin ISBN: 978-0-557-13832-6
23- Xamarin Mobile Application Development: Cross-Platform C# and Xamarin.Forms Fundamentals 2015th Edition. Dan Hermes ISBN-13: 978-1484202159
24- https://blog.xamarin.com/xamarin-for-all/
25- Källkritik på Internet
https://www.iis.se/lardig-mer/guider/kallkritik-pa-internet/
26- https://www.youtube.com/user/XamarinVideos
27- https://www.youtube.com/channel/UCFH-Tn6ggt9yLWJ4V9NizZA
28- http://www.mynewsdesk.com/se/pressreleases/omegapoint-en-av-sveriges-baesta-arbetsplatser-2016-1345427
29- Mobile Development with C#
2012th Edition. Greg Shackles. ISBN 978-1449320232
30- Software Requirements
Soren Lauesen ISBN 0-201-74557-4

8 Bilagor

8.1 Databasens struktur

Profiles klassen
har följande kolumner

- Namn: Är en string för användarens namn.
- Email: Är en string för användarensemail.
- Password: Är en string för användarenslösenord.

BokedTimes klassen
har följande kolumner

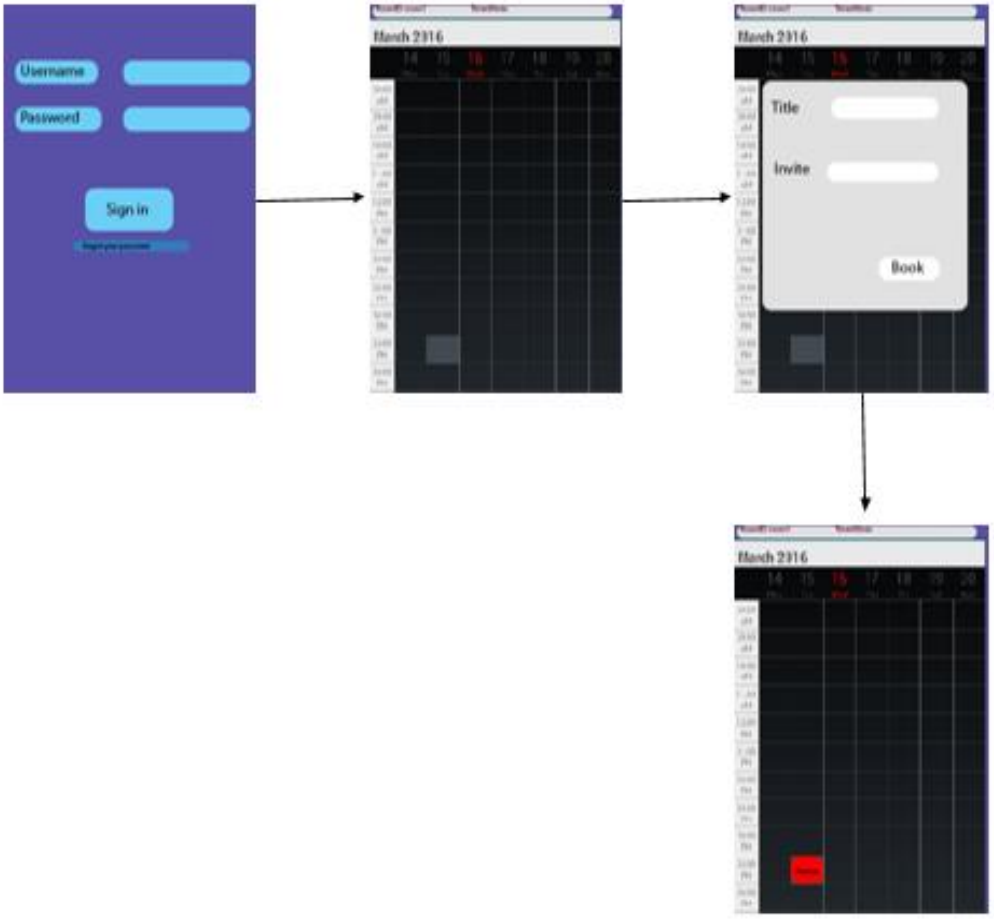
- BookingId: auto increment, ger unikt id för varje bokning. Används för att identifiera varje bokning.
- UsersEmail: String, innehåller email för den som har bokat.
- StartTime: String, innehåller start-datum och tid för bokningen.
- EndTime: String, innehåller end-datum och tid för bokningen.
- Title: String, bokningstitel.

InvitedMembers
har följande kolumner

- BookingId: Intager, unikt och ärvs från bokedTimes klassen.
- InvitedEmail: String, används för att spara användarnas email för de som blir inbjudna till ett vist event

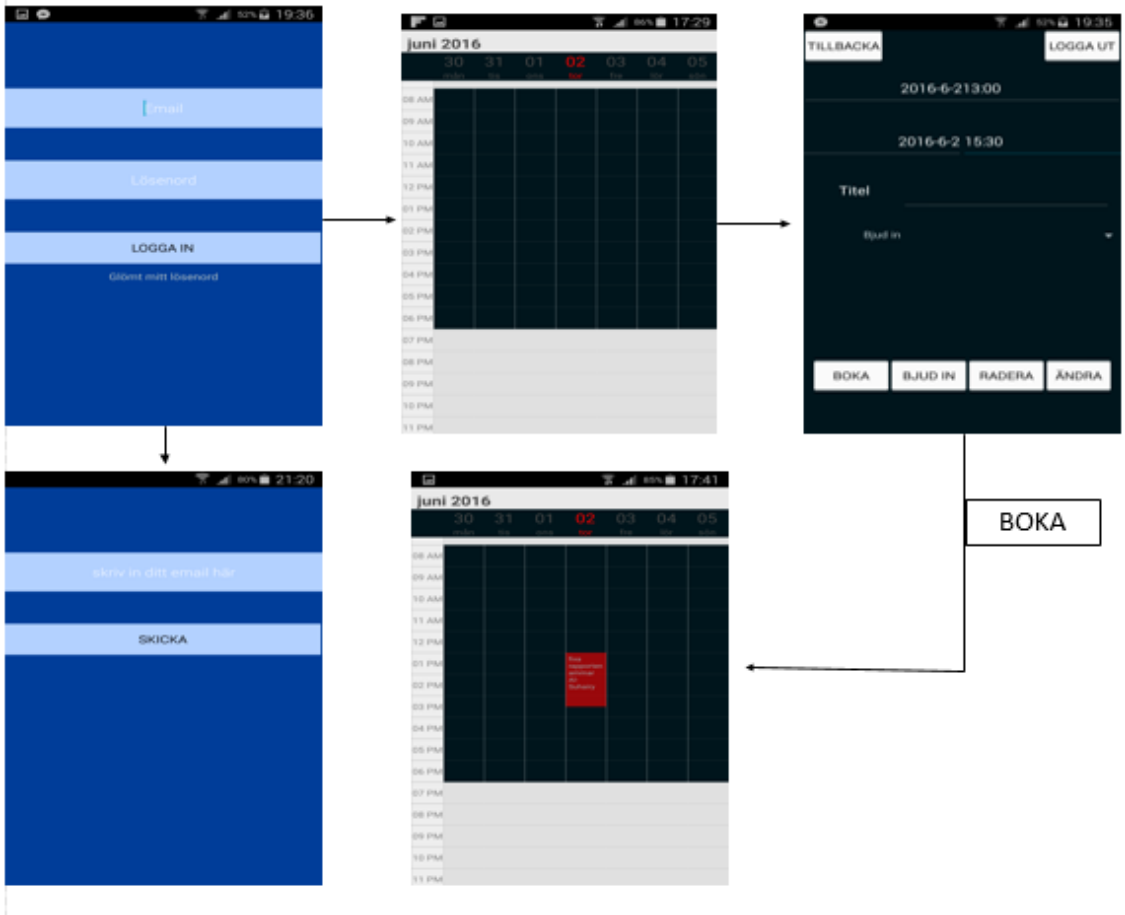
8.2 Prototyper

8.2.1 Prototyp1



Figur 8.2.1 visar en av de två prototyper som designades under arbetet.

8.2.2 Prototyp2



Figur 8.2.2 Visar den andra prototypen som designades under arbetet. Denna design valdes att implementeras.