Client Name: DVT
Project Name: KinderFinder

Software Architecture

Team Name: MAU Technologies

Uteshlen Nadesan	28163304
Michael Johnston	12053300
Po-Han Chiu	11063612

https://github.com/MrBean355/KinderFinder.git

Version: 0.1

23 May 2014

Contents

1 Architecture requirements

1.1 Architectural scope

The project will consist of five main components of which one is still a proof of concept (POC).

- Web Administration
- Mobile Application
- Web API
- Basic Reporting
- Embedded Hardware and prototyping (POC)

1.2 Quality requirements

The quality requirements are the requirements around the quality attributes of the systems and the services it provides. This includes requirements like performance, scalability, security, auditabilty, usability, and testability requirements.

1.3 Integration and access channel requirements

There are no external systems that must be integrated with. However, it needs to integrate with the various hardware devices (such as RFID transmitters and receivers).
The system will be access in two ways; the mobile app, which is used by the parents that are monitoring their children and through a web portal, which is how administrators will access the system for maintenance.

1.4 Architectural constraints

The only constraint for this project is the tools used to build the database for the system, those are the tools the client prefers and specified in the requirements specification.

2 Architectural patterns or styles

The main programming language to be used is C#, since we are required to use the .NET Framework.
The client browsers and app will be using HTTP messages to communicate with the web API.
The client requested that we make use of the Entity Framework, which is an ORM (object-relational mapper), as it will greatly improve maintainability.

3 Architectural tactics or strategies

As for Architectural tactics and strategies, we will are proposing to use a multi threaded approach in order to achieve scalability. Caching will be done on the maps that need to be downloaded to the mobile application, this will allow users to keep the maps of areas they feel they will frequently visit.

4 Use of reference architectures and frameworks

We will be using an object relational mapper in the form of the Entity Framework. This will be used to map to the SQL database making accessing database values easier through objects in the code. Reasons for choosing the Entity Framework is that in the type of database access we are dealing with, this object relational mapper provides reuse-ability so that if changes need to be made these changes are easier to implement in the future.

5 Access and integration channels

End product users will access the system through a Mobile application and the administrators of the system will access the system through a Web API with more privileges and higher level of access of the system compared to the average user. Due to the nature of the project and technologies that will be used, integration is fairly due to the fact that the programming languages for most components are all compatible.

6 Technologies

The technologies that will be used for this project are the following:

- Microsoft Visio Studio
- ASP .Net MVC 5
- HTML 5
- JavaScript
- JQuery
- ASP .Net WebApi
- C#
- Microsoft SQL Server 2012
- Xamarin

The above mentioned technologies are the technologies that are to be used for this project, but may still be subject to change at a later stage of the software development due to the unforeseen situations.