Software Architecture Documentation

For

Discovery Center - Museum Experience Survey

Prepared by: Andrew Mueller, Edward Dinki, Jonathon Shippling, Robert Vrooman

RIT Senior Project - Team MESSE

October 28, 2014

Table of Contents

1. High Level Architecture

Figure 1

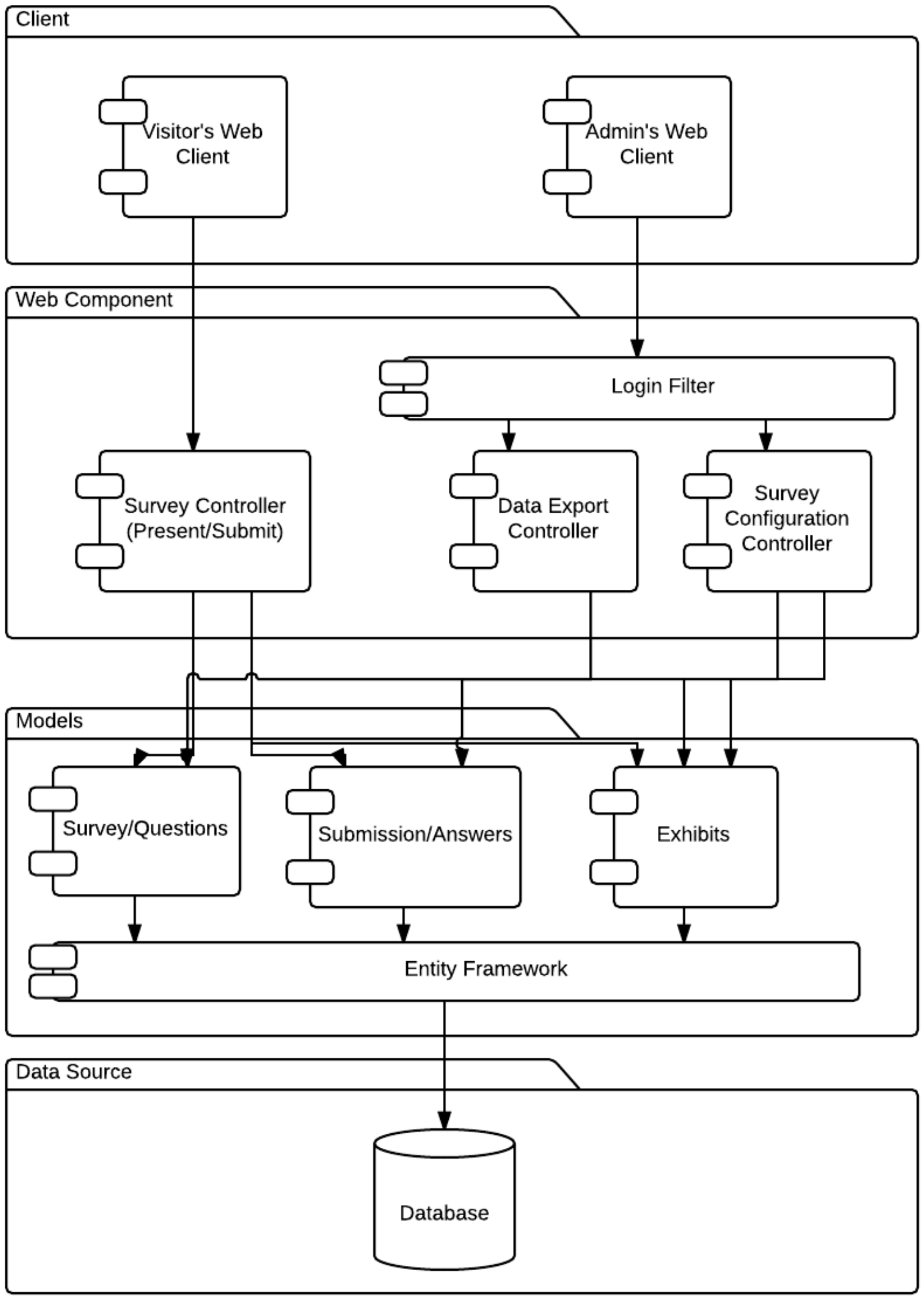


Figure 1 depicts logical components and dependencies within the application. This highlights the two different sides of the system experienced by the two different users, Visitors and Admins. Both have different sets of controllers determining what they can do and how they view data. Login Filter is used to show that admins using the system must pass through some sort of authentication before using any of the admin features. Also of note is that while there are these two different sides of the system, both share the same underlying models and database.

The Client layer serves to represent Visitor's and Admins connecting to the system via a web browser. These two users are split into two clients to more clearly show the different actions they will take in the system, but note that they do not necessarily have to be on different devices or browsers. The goal of the system, being a web application, is that it can run on a variety of hardware with multiple browsers. However, the Visitors will most often, if not exclusively, be connecting via an android tablet, and an admin will most often be connecting via a windows 7 desktop.

The Web Component layer shows some of .Net MVC's controller objects and their interactions. See Section 5 Controller Descriptions for more details on these objects. The component labeled Login Filter is present to represent that the controller's underneath it will require the user to pass some sort of authentication and authorization for the actions. Details of how logging in and authenticating will not be covered here.

The Models layer shows model objects used to represent surveys, questions, submissions, answers, and exhibits. These are standard objects with simple fields, and getters and setters. The Entity Framework component here aims to show our use of the Entity framework to interact with the database. The Entity Framework allows us to interact with model objects, and have them automatically persisted to the database.

The Data Source layer simply shows that we will be able to persist all model objects to a database. For more information on the database of choice, entity framework, and .NET MVC, please see Section 2 Technology Choices.

Figure 2

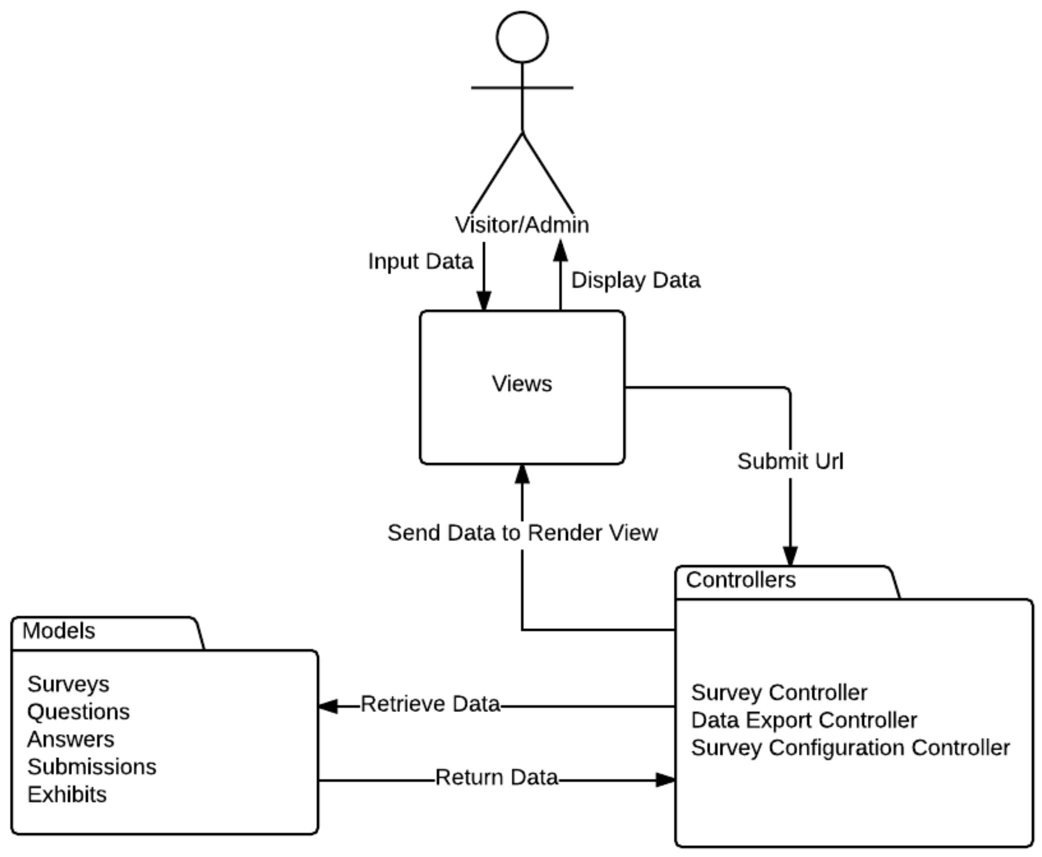
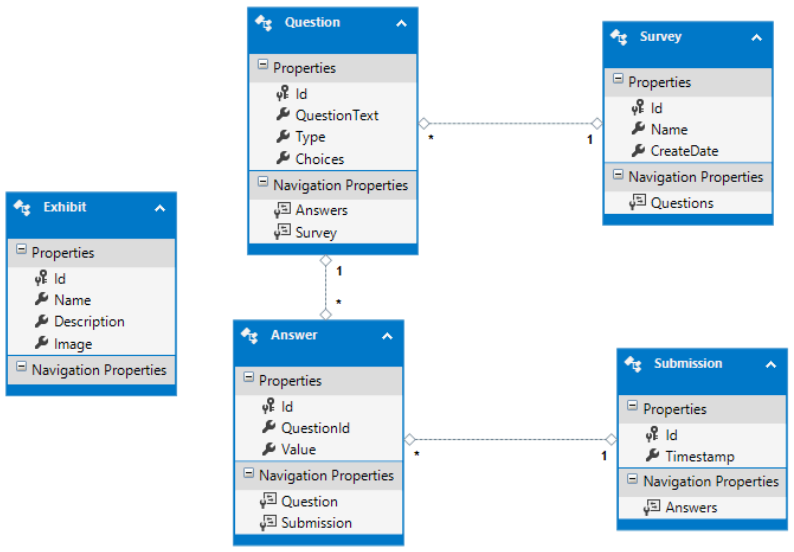


Figure 2 shows our use of Model View Controller (MVC) in the application and the basic flow of data. ASP .NET MVC provides this basic structure for us, and this diagram shows where our own specific implemented code would go. Additionally a simple and general flow of data is seen here. User inputs data, which gets submitted to a URL. Then the backend system's controller's execute some logic based on that and fetch appropriate data. Then the data is sent back to a view that renders itself for a user.

MVC is essentially splitting up the system into three logical pieces. This has a few benefits, including the fact that the modularity of the system allows different implementations to be swapped in much easier than if it was a high coupled system. This is why MVC stresses encapsulation and separation of concerns. Also by going with MVC we can make use of ASP .NET MVC instead of just plain ASP .NET using webpages. This provides a lot of scaffolding (auto generated code) that gets generated and put in place for us to work off us, allowing us to get the application up and running much faster.

Figure 3



The Entity-Relationship model for our database is depicted above in Figure 3. This design focused on simplicity of storing data as we did not need complex relationships. A Survey is a list of Questions. Each Question may have many different Answers by the separate users. After a Survey is completed by a user, the Answers are Timestamped and grouped together in a submission.

Question 'Type' refers to how the question is displayed; this may be multiple choice, open answer, or any other type available. The Question 'Choices' refers to a delimited string of answers that would be available for the question if the question is not open answer. The system will interpret the Question 'Type' and then decide how to parse the Question 'Choices'.

As noted in the Diagram, exhibits do not have relationships. The system will determine when to query the table, so it does not have to be directly linked to the others.

2. Technology Choices

ASP. NET MVC

ASP.NET MVC stands for Active Server Pages using the .NET framework and the Model View Controller Pattern. The Model View Controller pattern is something we as a team have used before and have familiarity with the design. ASP.NET MVC by default lends itself to use the Entity Framework and Microsoft SQL server as these are both part of the .NET framework.

Empty ASP.NET applications come with a scaffolded Authentication package that allows roles, such as Admin or User, to be set up quickly.

Entity Framework
When designing how to store Survey information, we came up with Objects such as Exhibits, Surveys, and Questions. Using these objects we decided that a Relational database was necessary to use an Object-Relational mapping type of approach to store the data. Entity Framework solves this mapping by the Use of "Entities" (table per object) that can be serialized and sent to the database.

The Entity framework includes integral security, such as providing inherent SQL injection security by design, as direct SQL does not need to be called to query the database or save objects to the database. LINQ (Language-Integrated Query) or DBContext.SaveChanges() is used instead. To actually execute an SQL command that would alter the database requires that the DBContext call DBContext.ExecuteSqlCommand.

Microsoft SQL Server
MS SQL Server is more than just a database, it is a Relational Database Management System. This means that tools for manipulation and viewing are available to an admin.

Viewing	- ability to view Table Data in a meaningful manner via SQL queries
Manipulating	- ability to change the data in a pre existing row in the database
Testing	- same as manipulating, but inserting bogus values to see how the survey system reacts.

MS SQL Server is built to work with .NET and this decreases the time spent on integration work. In comparison to MySQL, SQL Server has better Transaction and Replication support.

Drawbacks:
MySQL is free for all size database; the express version of MS SQL Server, which is the free version, only supports up to 10 Gigabytes of data. Due to the nature of the information being collected, a database size of 10 Gb will not be achieved. Data that is beyond two years old shall be deleted from the system.

MySQL is not our first choice because MS SQL Server was designed to work with the .NET environment.

3. Quality Attributes

3.1 Usability

Child Survey: The primary users of the children's surveys are going to be kids of all ages, so the children's survey should be as easy to use as possible. This includes limiting the number of buttons on the screen to reduce confusion, and asking questions that aren't too difficult. The children are assumed to be at a 4th grade reading level.

Adult Survey:The adult survey is targeted toward the general adult population. Normally applications for the general public target the users with moderate computer knowledge, but for this survey it is important to make it easy enough for even those with little computer knowledge to fill out the survey. To make the adult survey easier to use, we will use a wizard for taking the survey as well as using large buttons and text for visibility. UI conventions will be used where appropriate (ex. navigation on top or right side, icons will be commonly recognized).

Museum Admin Pages: The museum admin pages used to modify the surveys and view data will target the staff at the museum who have minimal computer experience. Taking this into account, page complexity should be minimized and the user should be guided through the application (ex. a wizard).

How to assure Usability: Gather the times that it takes for typical users of the child and adult surveys. The **adult survey should take under 5 minutes** while the **kid's survey should take under 2 minutes.** The UI walkthroughs will give the museum staff a preview of what the application will look like. This will allow them to give feedback so that the final product is what they want.

3.2 Security

This web application won't involve highly sensitive data like credit card information, but email addresses and personal information will still be involved and should be secured from unauthorized personnel.

How to assure Security: The web application requests should protect against basic vulnerabilities such as Cross Site Request Forgery attacks, buffer overflows, integer overflows, SQL injections, and any well-known vulnerabilities that allow spoofing of data. The application will also be hosted on a private wireless network making use of WPA, which will handle encrypting data when sending it over Wi-Fi. This should prevent any man-in-the-middle attacks since the application will be hosted locally.

3.3 Performance

Performance in the museum application environment is important for user satisfaction; However this is not a critical attribute.

How to assure Performance: ASP.net's mvc package supports concurrent user requests. These requests should be non-blocking. **Requests should take no longer than 5 seconds to**

complete and should take no longer than 1 second to show progress. However, this performance depends on factors including network quality and computer specifications which are out of our control.

3.4 Modifiability

For the most part, the surveys will be static, but the survey will change over time. Making the surveys modifiable means that the surveys will be easier to build for testing purposes as well as increase the longevity of the application's use in the museum. Without modifiability, the surveys may become obsolete. Modifiability along with usability will allow the museum admins to build and modify their own surveys and change them as they desire to reassure that they don't become obsolete or irrelevant.

<u>How to assure Modifiability:</u> An admin page will be designated toward survey creation and modification.

3.5 Appearance

The children's survey in particular will require a likeable interface for children to actually interact with the device and take the survey. Without an attractive appearance, children may lose interest or even think that they are breaking the rules by interacting with the survey device.

For the adult and admin pages, the page shouldn't look playful and hectic, but rather should be easy on the eyes while also being readable. Making it too boring may also deter adults.

<u>How to assure quality Appearance:</u> The color scheme and font will be bright and playful to attract kids. UI elements such as buttons will also be bigger so that they stand out more for kids. For the adult and admin pages, the color scheme will be chosen avoiding really bright colors. The font should also be larger so that it can be read by those with worse eyesight (within reason). Widgets will have styling to make them aesthetically pleasing.

4. Wireframes

See wireframe document.

5. Controller Descriptions

Controller: Survey Controller

Description:

Responsible for feeding Survey objects to the proper views, iterating through Surveys Question by Question. As well as creating Answers for those Questions as the user completes each one in a Submission object. Finally the survey controller submits the Submission to be saved to the database once the user finishes the survey.

Actions/Inputs:

Action	Inputs
Get list of surveys	-
Load survey	Survey ID
Get Next/Previous question	-
Submit Answer	Answer
Submit Survey	-

Controller: Survey Configuration Controller

Description:
Survey Configuration Controller is responsible for loading surveys into the proper views for configuring. Saves the changed/created surveys to the database.

Actions/Inputs:

Action	Inputs
Get list of surveys	-
Load survey	Survey ID
Create New Survey	-
Save Changes	Survey/Questions

Controller: Data Export Controller

Description:
For CSV Export, it will load the appropriate survey(s) from the database and start the conversion to a CSV, and send it to the client. For viewing the data on a webpage, it will load the appropriate survey and give the questions to the appropriate views.

Actions/Inputs:

Action	**Inputs**
Get list of surveys	-
Load Survey	Survey ID
Export Survey to CSV	Survey ID