Software Design Document for "Fun & Learn " Nursery

Doha Tariq, Rawan Hossam, Sara Nagy

August 1, 2018

1 Introduction

We are students and undergraduates from the university of Misr International University (MIU). In this course, we were asked to build a system for an institute, regardless the size, of our choosing. In said choice we picked a start-up nursery called "Fun & Learn" and this document serves as the software design document that aims to explain, clarify and enhance the knowledge of anyone seeking to understand how the system is build, how some of the choices were made and to clarify anything that isn't clear within the system itself.

1.1 Purpose of this document

The Purpose of this document is to encompass the complete software design requirements for our Nursery Management System (NMS) for a Nursery called "Fun & Learn". This Document targets any future developer and the professor supervising this course as our audience. This document also seeks to set and clarify the design parameters within and used in our Nursery Management System both functional and none functional and describes the design decisions, architectural design and the detailed design encompassed within the system. The audience intended by this document is any developer past, future or present that is going to take the responsibility of editing our system and edit or change it in anyway and the Professor supervising this course.

1.2 Scope

This document aims to assist future developers and designers understand our project and work. This document will include anything related to the design our system from the design patterns used to the architecture used to build this website. All for the hopes that it will serve as a guideline for what has been done and hopefully attempt to clarify anything within our system and explain some of the choices we've made in building this system. This system has the one and sole objective of making the Nursery day-to-day tasks easier and quicker.

This system has clear benefits over the pen and paper method of being quicker, versatile and overall more efficient than the previously used methods.

1.3 Overview

The system is supposed to provide a fast way of running the customer's nursery without the customer relying on pen-and-paper method of keeping data and storing information. The System shall be available to store all data necessary that is entered by Admins, Teachers and Users successfully and in an organized form, The System shall be designed to be child-friendly, it shall also perform reliably and accomplish the tasks assigned to the functionalities delivered,it must therefore be designed and implemented to ensure that the system always meets these requirements.

1.4 Reference Material

https://sourcemaking.com/design_patterns/strategy/php

1.5 Definitions and Acronyms

Term	Definition
Software Design Document (SDD)	Used as the primary medium for communicating software design information.
Design Entity	An element of a design that is structurally and functionally distinct from other elements.
encompass	surround and have or hold within.
Design Patterns	A software design pattern is a general, reusable solution to a commonly occurring problem within a given context in software design. It is not a finished design that can be transformed directly into source or machine code. It is a description or template for how to solve a problem that can be used in many different situations. Design patterns are formalized best practices that the programmer can use to solve common problems when designing an application or system.
enormous	Large in Number
significant	important
Normalize	homogenize, regularize, standardize
Manifest	become apparent, demonstrate
Encrypt	conceal data in
Hashed	A sequence of letters and numbers that is produced by the hashed item, the hash is completely unrecognizable from its original form

2 System Overview

Our system can and with automatic ease accomplish several of the Nursery's day-to-day tasks, all this without the need for the old pen-and-paper method and these task are as follows:

1. User side
   (a) We put the "Online Application" for children and their parents to apply
   (b) we put the "Teacher Registration" for teacher to apply for job openings
   (c) we put the "About us" page which provides information for visiting users
   (d) we put the "Contact us" page which provides information regarding the locations of the branches of "Fun & Learn" Nursery
   (e) we put the "Events" page which provides event related information to visiting users
   (f) we put the schedules in the "schedule" of the children for Visiting users to see despite the fact that our Nursery sometimes provides tailored scheduled for specific students so we put the standardized schedules on that page leaving the admin being able to tailor a schedule for specific students
2. Admin Side
   (a) we put the "Child Acceptance" & "Child Edit" on the admin side allowing admin to view, edit, update or delete the records
   (b) we put the "Teacher Acceptance" & "Teacher Edit" on the admin side allowing admin to view, edit, update or delete the records
   (c) we put the "Schedule" and within it "Courses" page on the admin side to allow the admin to custom tailor schedules if needed
   (d) we added the "Event" page on the admin side to allow the admin to add new events that can later on be displayed on the user side
   (e) we also put the "Control Panel" that gives the admin full CRUD control to an enormous amount of tables including all look up tables that in turn feedback to the user side such as:
      - "Status" Table
      - "Branch" Table
      - "Address" Table
      - "Marital Status" Table
      - "Qualification" Table
      - "Month" Table

- "Salary" Table
- "Nationality" Table
- "Payment" Table
- "Options" Table
- "Bill" Table

3 System Architecture

3.1 Architectural Design

Our systems architectural design is MVC; we used MVC because it gives a clear division between the data our System represents and how the user interacts with it. Moreover, MVC is more flexible and allows us (developers) to apply the changes that the customer comes up with during the phase of developing the system with minimum costs and its more structural, everything is separated in its own file. Using MVC makes it easier to maintain the code since the Models contain the parts of the code that directly access the database so any changes applied to the database will be directly applied there and the Views are only for displaying data to the user which means the design of our System and how the data will be represented, while the Controllers connect both Models and Views together. controllers also control what kind of data is displayed where on the UI. Furthermore, as previously mentioned it makes it easier to maintain the System which increases the efficiency of our System and makes it more suitable for any further future implementations.

3.2 Decomposition Description

A design pattern is a general repeatable solution to a commonly occurring problem in software design. Design pattern have many uses such as speeding up the development process by providing tested, proven development paradigms. Effective software design requires considering issues that may not become visible until later on the implementations We used several design pattern to accomplish some basic tasks such as encryption, salary calculation & multiple views. In order to achieve the tasks we used several design pattern which are: Strategy design pattern, Decorative design pattern, & MVC, respectively.

3.3 Strategy

In strategy design pattern we define a family of algorithms, encapsulate each one, and make then interchangeable. Strategy lets the algorithm vary independently from the clients that use it. Clients can then couple themselves to an interface, and not have to experience the upheaval associate with change.
Capture the abstraction in an interface and treat it as a base class, bury the implementation details in derived classes

3.4 Decorative

Decorative design pattern is implemented to represent some sort of layers of things on top of each other in order to come up with an entity of all the combined components.
The intent of decorative pattern is as follows: Attach additional responsibilities to an object dynamically. Decorators provide a flexible alternative to sub classing for extended functionality. It also is a client-specific embellishment of a core object by recursively wrapping it with the components chosen by the user, & it is a lot like wrapping a gift, putting it in a box, and wrapping the box.

3.5 MVC

MVC is a design pattern used to separate applications concern. It in turn divides the responsibly it has among its smaller components. While one component handles the back-&- forth operation between itself and the database, it is known as model. The other one handles how we direct the user through the system, what the user can and cannot see, and what the user can and cannot access; this is known as a controller. The last one is known as view which handles what shows up on screen for any user that accesses the page.

3.6 Model

A model represents an object that contains the functionality and the SQL statement needed to access, retrieve and store its data in the database.

3.7 View

A view represents the visualization of the data that the model contains

3.8 Controller

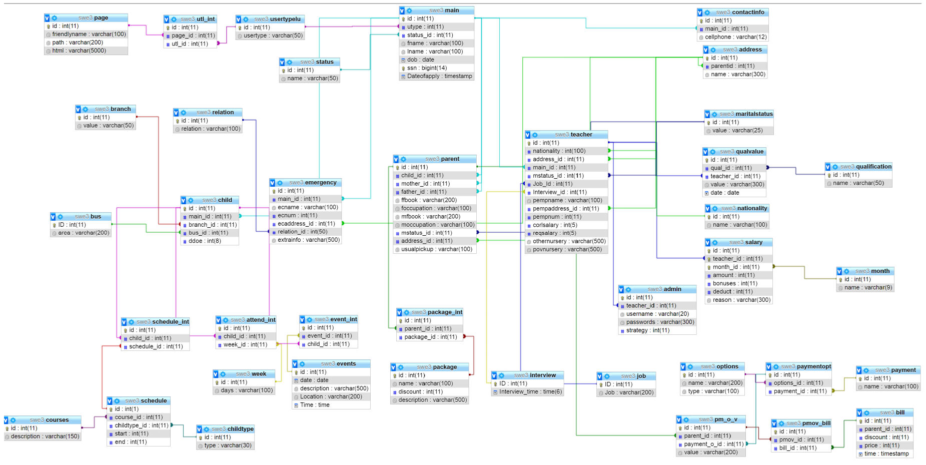
The controller acts on both model and view. It controls the data flow into the model objects and update the view whenever data change. It keeps the view and the model separate.

3.9 Design Rationale

The reason why we chose MVC as our main architectural design is as mentioned before; it has a clear division between everything. But just like any other design it has its own pros and cons For example the pros of MVC was it classifying everything in different places; database (sql) code is all together in the Model so its faster to modify or maintain, the UI or how the data is displayed for the user to manipulate is all in the View, again faster and easier to modify upon the users constant requests. Controllers are a bit trickier than the Models and the View because they help the Models and the views speak to each other in a way that is normalized but is also very hard to maintain due to the immense requests that has to go through it. It does save time to maintain but its high in cost when it comes to implementing. It also makes it harder for another developer to understand if they havent been there since the beginning. But overall MVC was the best architectural design choice.

4 Data Design

4.1 Data Description



According to the user stories we have gathered, the core tables we deal with for user data storage are:

1. Main
2. Parent
3. Teacher
4. Emergency
5. child
6. Contacinfo
7. qualvalue

The "Main" table stores all of the most basic information such as First name, Last Name, Date of birth and social security number, while the other tables store their own specific Data that cant be put in "Main". **REFER TO THE FIGURE IF NEEDED** The other core tables that only assists in the effective and normalized storage of the tables are:

1. Relation
2. Branch

3. Status
4. Address
5. Marital Status
6. Nationality
7. qualifications
8. Month
9. Options
10. Payment
11. Paymentopt
12. Pm_o_v

Each of the above tables despite being none user data storage tables, plays a significant role in database normalization & Homogeneity which also means that they help in achieving the first of our nonfunctional requirement which is Reliability.
Our second nonfunctional requirement was Maintainability which was achieved by the tables "options","payment","paymentopt","pm_o_v" and even though this functional requirement can not be attributed to these specific tables; they still represent a strong pillar to where maintainability manifests.
Some new nonfunctional requirements presented themselves while the software was under development which was the nonfunctional requirement known as Security. Security mainly manifested itself in the table admin which implemented the "Strategy" Design pattern and is used for holding the encrypted passwords of any admin.

4.2 Data Dictionary

1. Main
   - id (int)
   - utype (int)
   - status_id (int)
   - fname (varchar)
   - lname (varchar)
   - dob (date)
   - ssn (int)
   - dateofapply (timestamp)
2. Teacher

- id (int)
- nationality (int)
- address_id (int)
- main_id (int)
- mstatus_id (int)
- job_id (int)
- interview_id (int)
- pempname (varchar)
- corlsalary (int)
- reqsalary (int)
- othernursery (varchar)
- povnursery (varchar)

3. qualvalue
    - id (int)
    - qual_id (int)
    - teacher_id (int)
    - date (date)
4. parent
    - id (int)
    - child_id (int)
    - mother_id (int)
    - father_id (int)
    - main_id (int)
    - ffbook (varchar)
    - foccupation (varchar)
    - mfbook (varchar)
    - moccupation (varchar)
    - mstatus_id (int)
    - address_id (int)
    - usualpickup (varchar)
5. child
    - id (int)
    - main_id (int)

- branch_id (int)
- bus_id (int)
- ddoe (int)

6. Contactinfo
   - id (int)
   - main_id (int)
   - cellphone (int)
7. Emergency
   - id (int)
   - main_id (int)
   - ecname (int)
   - main_id (int)
   - cellphone (int)
   - main_id (int)
   - cellphone (int)
8. Relation
   - id (int)
   - relation (varchar)
9. Branch
   - id (int)
   - Value (varchar)
10. status
    - id (int)
    - name (varchar)
11. Address
    - id (int)
    - parentid (int)
    - name (varchar)
12. Marital Status
    - id (int)
    - Value (varchar)

13. options
    - id (int)
    - name (varchar)
    - type (varchar)
14. payment
    - id (int)
    - name (varchar)
15. paymentopt
    - id (int)
    - payment_id
    - options_id (varchar)
16. paymentopt
    - id (int)
    - parent_id
    - payment_o_id (varchar)
    - value (varchar)
17. Bill
    - id (int)
    - parent_id
    - discount (int)
    - price (int)
    - time (CURRENT_TIMESTAMP)

All the above tables helped in some very important things in our project. The EAV model of the 4 tables named as payment,paymentopt,options,pm_o_v assisted in an important nonfunctional requirement known as maintainability, while tables like Branch,Status,relation, qualification, status, address, marital status are referred to as Reference tables. These tables are also known as Look-up tables which help in maintaining the database by normalizing the database with its contents.

5 Component Design

First off, we used a triple layer encryption and hashing algorithm to make sure that passwords are extremely secure, the pseudo code is as follows:
user input: password
processing: password is taken then salt is added onto password from both ends like so: salt1+password+salt2
output: saltedpass
2nd layer goes as follows: input:saltedpass
processing:

1. Take command line arguments for string to be encoded and a single alpha character as a cipher key.
2. Validate cipher key is alpha, convert to an integer and offset relative to 0 irrespective of case (e.g. C = 2, d = 3)
3. Loop through each character in saltedpass, change value by the value of the cipher only when alpha.

output: encryptedpass 3rd level we use the Secure Hash Algorithm 1 pre-made function to hash the password as follows:
input: encryptedpass
processing: sha1 takes encryptedpass as a parameter like so: sha1(encryptedpass)
output: The output is the password hashed and encrypted

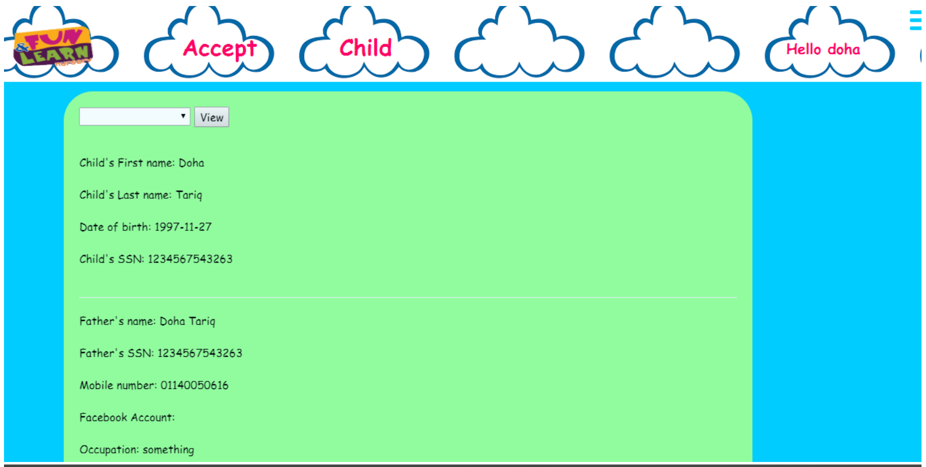
6 Human Interface Design

6.1 Overview of User Interface

We intended to make our GUI easy to deal with per our customer's request so we made the screen connected but not too connected that it becomes confusing for the user how to get to a specific page. On the admin side we've a navigation menu that leads to all the pages. Navigation Bar → Child Acceptance → Home page
Navigation Bar → Edit Child → Home page
Navigation Bar → Control Panel → Home page
Navigation Bar → Teacher Acceptance → Home page
Navigation Bar → Teacher Edit → Home page
Navigation Bar → Schedule → Home page
Navigation Bar → Events → Home page
Navigation Bar → Logout → Home page (In this step the admin will be signed out but sent to the homepage still)
On the user side navigation is equally easy to handle as on the admin side and while the homepage contains basic things like "Contact us"&"About us" information there are things that were given their own pages Navigation Bar → Online Application → Home page

Navigation Bar → Teacher Application → Home page
Navigation Bar → Schedule → Home page
Navigation Bar → Events → Home page

6.2 Screen Images



Child's First name: Doha

Child's Last name: Tariq

Date of birth: 1997-11-27

Child's SSN: 1234567543263

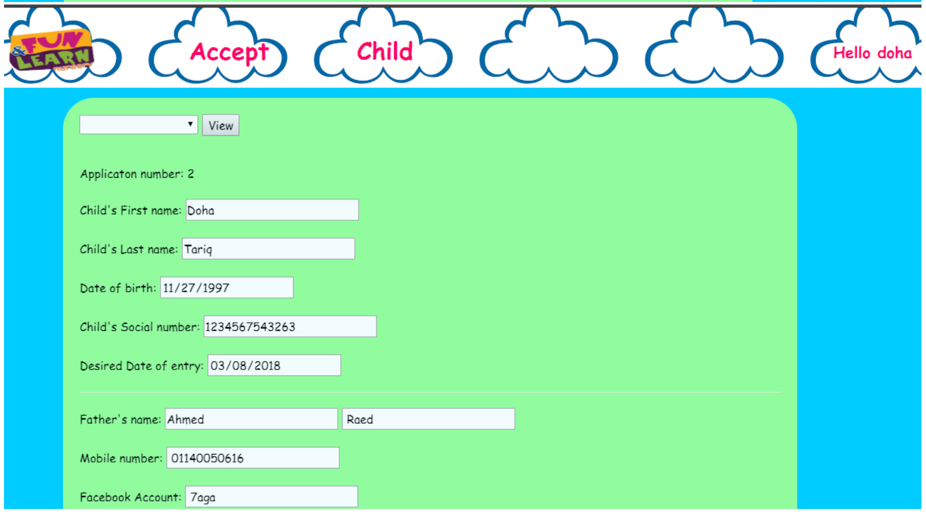
Father's name: Doha Tariq

Father's SSN: 1234567543263

Mobile number: 01140050616

Facebook Account:

Occupation: something



Applicaton number: 2

Child's First name: Doha

Child's Last name: Tariq

Date of birth: 11/27/1997

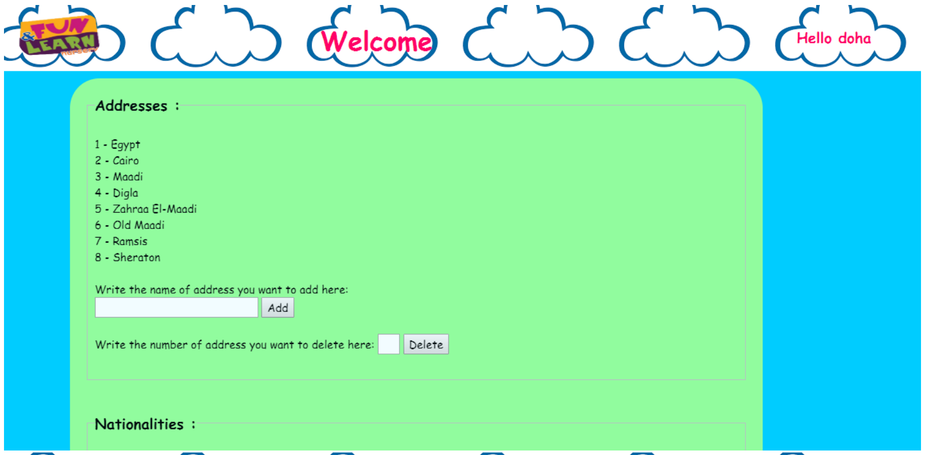
Child's Social number: 1234567543263

Desired Date of entry: 03/08/2018

Father's name: Ahmed Raed

Mobile number: 01140050616

Facebook Account: 7aga



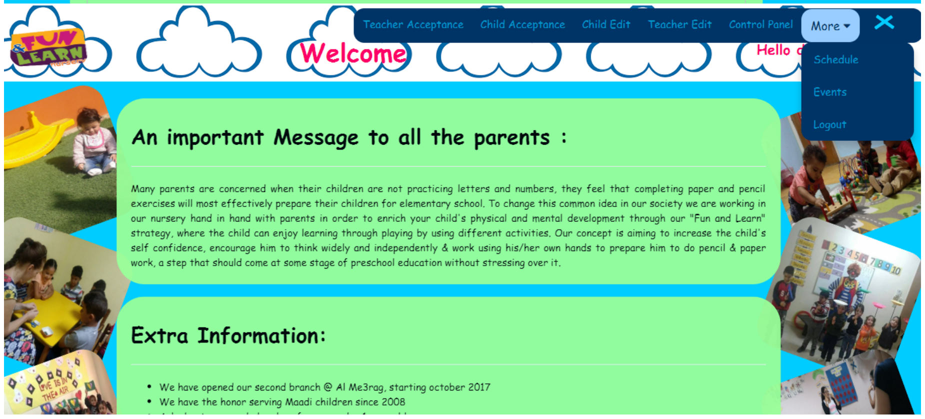
Addresses :

1 - Egypt
2 - Cairo
3 - Maadi
4 - Digla
5 - Zahraa El-Maadi
6 - Old Maadi
7 - Ramsis
8 - Sheraton

Write the name of address you want to add here:
Add

Write the number of address you want to delete here:
Delete

Nationalities :



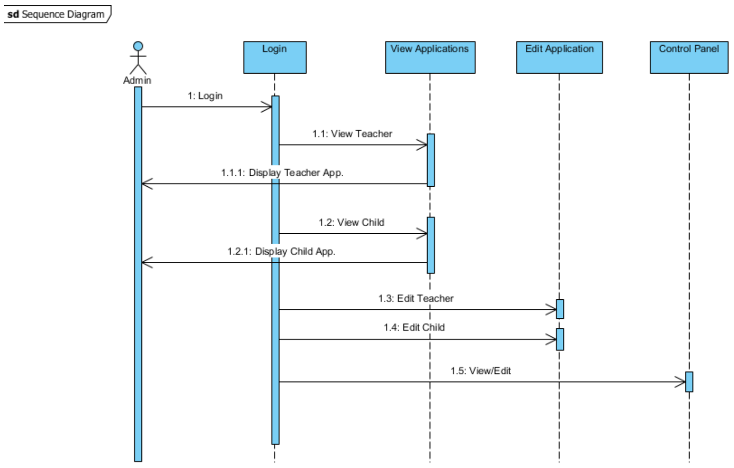
An important Message to all the parents :

Many parents are concerned when their children are not practicing letters and numbers, they feel that completing paper and pencil exercises will most effectively prepare their children for elementary school. To change this common idea in our society we are working in our nursery hand in hand with parents in order to enrich your child's physical and mental development through our "Fun and Learn" strategy, where the child can enjoy learning through playing by using different activities. Our concept is aiming to increase the child's self confidence, encourage him to think widely and independently & work using his/her own hands to prepare him to do pencil & paper work, a step that should come at some stage of preschool education without stressing over it.

Extra Information:

- We have opened our second branch @ Al Me3rag, starting october 2017
- We have the honor serving Maadi children since 2008

6.3 Screen Objects and Actions



7 Requirements Matrix