Table of Contents

1.1 Description.................................................................................................................................4
1.2 Development Tools.......................................................................................................................5
1.3 Goals................................................................................................................................................5
1.4 Milestones.........................................................................................................................................6
1.5 Online activity managers...............................................................................................................6
1.6 Server Side Scripting.......................................................................................................................7
1.7 Client side scripting........................................................................................................................8
1.8 Programming languages................................................................................................................9
2.1 Relational Model............................................................................................................................11
2.1 Entity sets descriptions..................................................................................................................12
2.2 Conceptual Database Design ......................................................................................................13
   Name: Account.................................................................................................................................13
   Name: Users.......................................................................................................................................15
   Name: Chores.................................................................................................................................16
   Name: Rewards...............................................................................................................................19
   Name: rlog........................................................................................................................................20
   Name: clog........................................................................................................................................22
2.3 MYSQL CODE...............................................................................................................................23
2.4 Created Tables...............................................................................................................................27
3.1 Website design...............................................................................................................................29
3.2 Website Goals...............................................................................................................................29
3.3 Website page view.........................................................................................................................31
4.1 Important Code........................................................................................................................................37

4.2 Left.php..................................................................................................................................................37

4.3 Transaction.php.....................................................................................................................................39

4.4 Register.php.........................................................................................................................................40

4.5 Top.php..................................................................................................................................................41

4.6 Logout.php............................................................................................................................................42

4.7 edit.php................................................................................................................................................43

4.8 create.php..............................................................................................................................................44

4.9 Transaction1.php...................................................................................................................................45

4.10 Wall.php...............................................................................................................................................46

5.1 Update and feedback...............................................................................................................................47

5.2 Make views use month............................................................................................................................47

5.3 Let the user download a text file with full records...............................................................................48

5.4 Make a Household report.........................................................................................................................49

6.1 Difficulties with project...........................................................................................................................50

6.2 Triumphs of the Project..........................................................................................................................50

6.3 Project significance.................................................................................................................................51

6.4 What I have learned...............................................................................................................................52
1.1 Description

The goal of my project is to develop and create a website that functions as a chore log for a given household. I decided on this project after a good experience in cs342 database class. I have always liked web development and found that consumer applications and websites were something that interested me greatly. The chore website allows people to add users to a household and remove them at leisure. The master user can also create chores and rewards with specific point values that the users will be able to choose from and complete.

I will be using a database to store all relevant information such as account info, chore info, and reward info. The website itself is designed from a list of HTML templates that I found online. I chose this specific one because I enjoyed the layout and flow of the structures, I have modified it greatly to change it to the aspects such as the different sizes and styles of the web page as well as adding numerous other pages. For this project, the database chose will be MYSQL stored on the www.cs.csubak.edu server. I originally planned to make it on the Delphi server but due to constant connectivity issues, moved it to the main website. The languages I will be using are CSS, HTML, HTML5, PHP, and Javascript.
1.2 Development Tools

- Filezilla for quick ftp loading and testing.
- Notepad++ for multiple web page editing.
- Putty to connect to CSUB's server.
- PHP language for all server side scripting.
- HTML(5) to build the website interface for the users.
- Javascript for certain addons and scripts.
- MySQL as the database software.
- CSS to style said HTML pages.

1.3 Goals

1) Create Web pages for the user to navigate from.
2) Create a database that stores information for the website.
3) Use PHP to make calls the select database to store and retrieve information.
4) Style the website so that it has useful design and can be attractive.
5) Make sure login info is validated and follows guidelines.
6) Allow the Account creator complete control over the list of events and users.
7) Expand on the knowledge and effort invested into the cs 342 project.
8) Create and maintain all of this project as a solo endeavor.
1.4 Milestones

✔ Deciding on what project to do.

✔ Originally trying to develop in android studio for mobile.

✔ Changing the project to web-based design.

✔ Figuring out what tables and their relations that need to be created.

✔ Creating the database, load test data, and make sure test queries work.

✔ Designing the website and making all main PHP pages.

✔ Integrate and test login/register information.

✔ Create Rest of queries for the display of data.

✔ Create last set of queries for deleting and creating data.

✔ Add finishing touches to polish website design.

✔ Document the project.

1.5 Online activity managers.

Online websites mirror things that are obtainable in normal every day life. A balance book can keep track of finances, A fitness routine that logs the hours
complete, or even like the previous project we built, a finance tracker for household transactions. These websites usually support themselves through ad based revenue or by offering extra features for an upgrade fee, usually while offering initially free services. What ever subject you can think about listing and writing about in the physical world can be done through online computing. As society trends to an even more technological slant, people will look to replace pen and paper devices with information solely stored in the digital medium.

1.6 Server Side Scripting.

Server-side scripting is a technique used in web development which involves employing scripts on a web server which produce a response customized for each user's (client's) request to the website. The alternative is for the web server itself to deliver a static web page. Scripts can be written in any of a number of server-side scripting languages that are available. Server-side scripting is distinguished from client-side scripting where embedded scripts, such as JavaScript, are run client-side in a web browser, but both techniques are often used together.

Server-side scripting is often used to provide a customized interface for the user. These scripts may assemble client characteristics for use in customizing the response based on those characteristics, the user's requirements, access rights, etc. Server-side scripting also enables the website owner to hide the source code that generates the interface, whereas with client-side scripting, the user has access to all the code
received by the client. Examples of Server-side-scripting are PHP, Perl, and Java.

1.7 Client side scripting

Client-side scripting generally refers to the class of computer programs on the web that are executed client-side, by the user's web browser, instead of server-side (on the web server). This type of computer programming is an important part of the Dynamic HTML (DHTML) concept, enabling web pages to be scripted; that is, to have different and changing content depending on user input, environmental conditions (such as the time of day), or other variables.

Client-side scripts are often embedded within an HTML or XHTML document (hence known as an "embedded script"), but they may also be contained in a separate file, to which the document (or documents) that use it make reference (hence known as an "external script"). Upon request, the necessary files are sent to the user's computer by the web server (or servers) on which they reside. The user's web browser executes the script, then displays the document, including any visible output from the script. Client-side scripts may also contain instructions for the browser to follow in response to certain user actions, (e.g., clicking a button). Often, these instructions can be followed without further communication with the server.
1.8 Programming languages

HTML - is written in the form of HTML elements consisting of tags enclosed in angle brackets (like <HTML>). HTML tags most commonly come in pairs like <h1> and </h1>, although some tags represent empty elements and so are unpaired, for example <img>. The first tag in a pair is the start tag, and the second tag is the end tag (they are also called opening tags and closing tags). HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts written in languages such as JavaScript which affect the behavior of HTML web pages.

PHP- is a server-side scripting language designed for web development but also used as a general-purpose programming language. PHP code can be simply mixed with HTML code, or it can be used in combination with various template engines and web frameworks. PHP code is usually processed by a PHP interpreter, which is usually implemented as a web server's native module or a Common Gateway Interface (CGI) executable. After the PHP code is interpreted and executed, the web server sends resulting output to its client, usually in form of a part of the generated web page; for example, PHP code can generate a web page's HTML code, an image, or some other data.

CSS- is a style sheet language used for describing the look and formatting of a
document written in a markup language. While most often used to change the style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any kind of XML document, including plain XML, SVG and XUL. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications.

JavaScript is a dynamic computer programming language. It is most commonly used as part of Web browsers, whose implementations allow client-side scripts to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed. It is also used in server-side network programming with run time environments such as Node.js, game development and the creation of desktop and mobile applications. With the rise of the single-page Web app and JavaScript-heavy sites, it is increasingly being used as a compile target for source-to-source compilers from both dynamic languages and static languages.
2.1 Relational Model
2.1 Entity sets descriptions

Account: This will hold the user data for logging into the database. This will be the parent relation to the rest of relations. It holds the relevant login information as well as household name and code which is the pin number to create and edit in the website.

Users: This holds all users for the database, it is linked to an account. It also has a one to many relationship with both log entity's Names and point values are stored here.

Rewards: Holds all reward descriptions and point values. It has a child relationship with Account much liker User and it is part of a junction table rlog.

Chores: Holds all Chore descriptions, point values and the time the chores are to be performed by. It, like Rewards, has a child relation with Account and shares the junction table rlog with Users.

rlog: A Junction table based of Users and Rewards. This table stores a user with a specific reward as well as time stamping when it happens.

clog: A Junction table based of Users and chores. This table stores a user with a specific chore as well as time stamping when it happens.
2.2 Conceptual Database Design

Entity 1:

**Name: Account**

Description: The main log-in entity.

Attributes:

- **Name: AccountID**
  - Description: Main Primary key for Database
  - Variable type: int.
  - Value-Range: 1-99999
  - Default value: 1
  - Null value: No.
  - Unique: Yes.
  - Single or multiple: Single.
  - Simple or composite: Simple.

- **Name: Code**
  - Description: pin number to unlock edit controls
  - Variable type: int.
  - Value-Range: 1-9999
  - Default value: 0
  - Null value: No.
  - Unique: Yes.
  - Single or multiple: Single.
  - Simple or composite: Simple.

- **Name: Login**
  - Description: login name
  - Variable type: TINYTEXT
Value-Range: 255 (2^8 - 1)
Default value: 0
Null value: No.
Unique: Yes.
Single or multiple: Single.
Simple or composite: Simple.

**Name: HashPass**
Description: password
Variable type: VARCHAR(30)
Value-Range: 0-30 characters
Default value: 0
Null value: No.
Unique: no
Single or multiple: Single.
Simple or composite: Simple.

**Name: Household**
Description: Household name to be displayed
Variable type: VARCHAR(20)
Value-Range: 0-20 characters
Default value: 0
Null value: No.
Unique: no
Single or multiple: Single.
Simple or composite: Simple.
**Name: Users**

Description: List of all users and which accounts they belong too

**Attributes:**

- **Name: UserID**
  - Description: Primary key of table Users
  - Variable type: int
  - Value-Range: 1-99999
  - Default value: 1
  - Null value: No.
  - Unique: Yes.
  - Single or multiple: Single.
  - Simple or composite: Simple.

- **Name: AccID**
  - Description: Foreign key to Account table
  - Variable type: int
  - Value-Range: 1-99999
  - Default value: 1
  - Null value: No.
  - Unique: No.
  - Single or multiple: Single.
  - Simple or composite: Simple.

- **Name: Points**
  - Description: Point value for individual Users
  - Variable type: int
  - Value-Range: 1-999999999
  - Default value: 1
  - Null value: No.
  - Unique: No.
Single or multiple: Single.
Simple or composite: Simple.

**Name:** Name
Description: Name of users
Variable type: Varchar(20)
Value-Range: 20 characters
Default value: 1
Null value: No.
Unique: no
Single or multiple: Single.
Simple or composite: Simple.

Entity 3:

**Name: Chores**

Description: List of all Chores created.

**Attributes:**

**Name:** ActivityID
Description: Primary key of table Chores
Variable type: int
Value-Range: 1-99999999
Default value: 1
Null value: No.
Unique: Yes.
Single or multiple: Single.
Simple or composite: Simple.

**Name:** CaccountID
Description: Foreign key to Account table
Variable type: int
Value-Range: 1-99999
Default value: 1
Null value: No.
Unique:no
Single or multiple: Single.
Simple or composite: Simple.

Name: Description
Description: Chore description
Variable type: TinyText
Value-Range: 255 ($2^8 - 1$)
Default value: null
Null value: yes
Unique:no
Single or multiple: Single.
Simple or composite: Simple.

Name: ChoreTime
Description: When chores are expect
Variable type: DATETIME
Value-Range: date + time
Default value: null
Null value: yes
Unique: no
Single or multiple: Single.
Simple or composite: Simple.

Name: PointValue
Description: Chore points value
Variable type: INTEGER(20)
Value-Range: 1-999999999
Default value: 0
Null value: no
Unique: no
Single or multiple: Single.
Simple or composite: Simple.

**Name: Frequency**
Description: Set values to once, daily, weekly, or monthly.
Variable type: VARCHAR (was enum)
Value-Range: varchar(10)
Default value: 0
Null value: yes
Unique: no
Single or multiple: Single.
Simple or composite: Simple.

**Name: Completed**
Description: binary option to see if chore is completed or not
Variable type: Boolean
Value-Range: 0-1
Default value: 0
Null value: no
Unique: no
Single or multiple: Single.
Simple or composite: Simple.

Entity 4:
**Name: Rewards**

Description: Rewards created by user

Attributes:

**Name: RewardID**
Description: Primary key of table Rewards
Variable type: int
Value-Range: 1-99999999
Default value: 1
Null value: No.
Unique: Yes.
Single or multiple: Single.
Simple or composite: Simple.

**Name: RaccountID**
Description: Foreign key to Account table
Variable type: int
Value-Range: 1-999999
Default value: 1
Null value: No.
Unique: no
Single or multiple: Single.
Simple or composite: Simple.

**Name: Reward**
Description: Reward description
Variable type: TinyText
Value-Range: 255 (2^8 - 1)
Default value: 0
Null value: yes
Unique: no
Single or multiple: Single.
Simple or composite: Simple.

**Name: RewardValue**
Description: Reward points value
Variable type: INTEGER(10)
Value-Range: 1-999999999
Default value: 0
Null value: No
Unique: No
Single or multiple: Single.
Simple or composite: Simple.

**Name: rlog**
Description: log of rewards when claimed by a user

**Attributes:**

**Name: RlogID**
Description: Primary key of table rlog
Variable type: int
Value-Range: 1-999999999
Default value: 1
Null value: No.
Unique: Yes.
Single or multiple: Single.
Simple or composite: Simple.
Name: IDrewards
Description: Foreign key of table Rewards
Variable type: int
Value-Range: 1-9999999
Default value: 1
Null value: No.
Unique: Yes.
Single or multiple: Single.
Simple or composite: Simple.

Name: IDUSER
Description: Foreign key of table Users
Variable type: int
Value-Range: 1-9999999
Default value: 1
Null value: No.
Unique: Yes.
Single or multiple: Single.
Simple or composite: Simple.

Name: REWoccurrence
Description: Takes the time when a user claims a reward.
Variable type: TimeSTAMP
Value-Range: date time
Default value: 1
Null value: No.
Unique: Yes.
Single or multiple: Single.
Simple or composite: Simple.
**Name: clog**

Description: log of chores when claimed by a user

Attributes:

**Name: ClogID**
Description: Primary key of table Clog
Variable type: int
Value-Range: 1-9999999
Default value: 1
Null value: No.
Unique: Yes.
Single or multiple: Single.
Simple or composite: Simple.

**Name: IDactivity**
Description: Foreign key of table Chores
Variable type: int
Value-Range: 1-9999999
Default value: 1
Null value: No.
Unique: Yes.
Single or multiple: Single.
Simple or composite: Simple.

**Name: IDUSER**
Description: Foreign key of table Users
Variable type: int
Value-Range: 1-9999999
Default value: 1
Null value: No.
Unique: Yes.
Single or multiple: Single.
Simple or composite: Simple.

**Name:** CHoccurrence

Description: Takes the time when a user claims a chore.
Variable type: TimeSTAMP
Value-Range: date time
Default value: 1
Null value: No.
Unique: Yes.
Single or multiple: Single.
Simple or composite: Simple.

### 2.3 MYSQL CODE

CREATE TABLE account
(
    accountid INTEGER(5) NOT NULL auto_increment,
    code      INTEGER(4) NOT NULL,
    login     TINYTEXT,
    hashpass  VARCHAR(30),
    household VARCHAR(20),
    CONSTRAINT pk_account PRIMARY KEY (accountid)
CREATE TABLE users
(
    userid INTEGER(5) NOT NULL auto_increment,
    accid INTEGER(5),
    points INTEGER(10),
    name VARCHAR(20),
    CONSTRAINT fk_users_accid FOREIGN KEY (accid) REFERENCES account(accountid)
    ON UPDATE CASCADE ON DELETE CASCADE,
    CONSTRAINT pk_user PRIMARY KEY (userid)
);
CREATE TABLE chores
(
    activityid INT NOT NULL auto_increment,
    caccountid INT,
    completed BOOLEAN DEFAULT 0 NOT NULL,
    CONSTRAINT fk_chores_caccountid FOREIGN KEY (caccountid) REFERENCES accountid (account) ON UPDATE CASCADE ON DELETE CASCADE,
    CONSTRAINT pk_chores PRIMARY KEY (activityid)
);
choccurence TIMESTAMP,

CONSTRAINT fk_clog_iduser FOREIGN KEY (iduser) REFERENCES users (userid) ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk_clog_idactivity FOREIGN KEY (idactivity) REFERENCES chores (activityid) ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT pk_two_keys PRIMARY KEY (clogid);

CREATE TABLE rlog

(
    rlogid       INTEGER(8) NOT NULL auto_increment,
    idrewards    INTEGER(8),
    iduser       INTEGER(5),
    rewoccurence TIMESTAMP,

    CONSTRAINT fk_rlog_iduser FOREIGN KEY (iduser) REFERENCES users (userid) ON UPDATE CASCADE ON DELETE CASCADE,

    CONSTRAINT fk_rlog_idrewards FOREIGN KEY (idrewards) REFERENCES rewards (rewardid) ON UPDATE CASCADE ON DELETE CASCADE,

)
CONSTRAINT pk_rlog_keys PRIMARY KEY (rlogid)

2.4 Created Tables

```
mysql> desc Account;
+ Field    | Type         | Null | Key   | Default | Extra          |
+-----------|--------------|------|-------|---------|----------------|
| AccountID | int(5)       | NO   | PRI   | NULL    | auto_increment |
| Code      | int(4)       | NO   |       | NULL    |                |
| login     | tinytext     | YES  |       | NULL    |                |
| HashPass  | varchar(30)  | YES  |       | NULL    |                |
| Household | varchar(20)  | YES  |       | NULL    |                |
```
```sql
mysql> desc Users;
+-------+--------+-------+--------+-------+---------------------+
| Field | Type   | Null | Key   | Default | Extra          |
+-------+--------+-------+--------+-------+---------------------+
| UserID | int(5) | NO    | PRI    | NULL   | auto_increment     |
| AccID  | int(5) | YES   | MUL    | NULL   |                   |
| Points | int(10)| YES   |        | NULL   |                   |
| Name   | varchar(20) | YES |        | NULL   |                   |
+-------+--------+-------+--------+-------+---------------------+
4 rows in set (0.00 sec)

mysql> desc Chores;
+-------+--------+-------+--------+-------+---------------------+
| Field | Type   | Null | Key   | Default | Extra          |
+-------+--------+-------+--------+-------+---------------------+
| ActivityID | int(8) | NO    | PRI    | NULL   | auto_increment     |
| CaccountID | int(5) | YES   | MUL    | NULL   |                   |
| Description | tinytext | YES |        | NULL   |                   |
| ChoreTime | datetime | YES |        | NULL   |                   |
| PointValue | int(20) | YES   |        | NULL   |                   |
| Frequency | tinytext | YES |        | NULL   |                   |
| Completed | tinyint(1) | NO |        | 0      |                   |
+-------+--------+-------+--------+-------+---------------------+
7 rows in set (0.01 sec)

mysql> desc Rewards;
+-------+--------+-------+--------+-------+---------------------+
| Field | Type   | Null | Key   | Default | Extra          |
+-------+--------+-------+--------+-------+---------------------+
| RewardID | int(8) | NO    | PRI    | NULL   | auto_increment     |
| RaccountID | int(5) | YES   | MUL    | NULL   |                   |
| Reward | tinytext | YES |        | NULL   |                   |
| RewardValue | int(10) | YES |        | NULL   |                   |
+-------+--------+-------+--------+-------+---------------------+
4 rows in set (0.00 sec)
```
3.1 Website design

I used a template website and modified to my needs. The first thing I did was strip it of everything except its bear CSS and Div structures. I wanted the box method for web design and the template had the perfect combination of what I was looking for. I added roughly 12 additional PHP pages from the original 5 HTML pages the template used, leading to a total of 17 different PHP pages with 2-3 being similar in theme but completely different in function. The design challenges mostly involved manipulating the CSS style sheets to get exactly what I wanted.

3.2 Website Goals

1. Create login page that is not visible to logged in user.

2. Create Registration page that returns errors on failure.
3. Display database rewards and chores to different pages.

4. using GET to select a user and keep them as the logged in user.

5. Allow users to complete and purchase chores and rewards.

6. Allow master user to create/delete rewards and chores.

7. Upgrade master user account to be able to delete users.

8. Make sure user knows who is logged in.
3.3 Website page view

Index.php:

This is the main page users must first go to registration page.
Register.php

Users can register their account here, it will redirect after they successfully create an account.
Archives.php:

After logging in users will be redirect to the chore page:
example.php:

Clicking on the rewards page will go to the current rewards for the account.
wall.php:

Until a user selects one of the household helpers on the left bar, the user wall will not be available. This page is able to be navigated to once the user is selected as shown.

Notice the household name is now replaced with the user name indicating they are the current user.
Create.php:

This page requires the pin code created at user registration to unlock. Here the user can delete other users and delete/create chores/rewards.
4.1 Important Code

Connect.php

```php
<?php

// sets login information for mysql database
define("USERNAME", "scotera");
define("PASSWORD", "trindle");
define("DATABASE", "scotera");
define("HOST", "localhost");

// connect to the mysql server
$link = mysql_connect(HOST, USERNAME, PASSWORD, DATABASE);
if (!@$link)
{
    die('Could not connect: ' . mysql_error());
}

?>
```

set the database here and create a PHP page that will be called in every page at the beginning to insure my DB is connected.

4.2 Left.php:
This piece of code was used to create the drop down menus that propagate the complete and delete chores/rewards buttons. The session variable accountid is set when the user logs in from the front page. It then uses the query to propagate the option list from the drop down menu. The concatenated menu options are the tables rows names and point values. The option value for the form is the matching ActivityID, this will be used to determine what values need to be updated or deleted, which leads too...
<?php session_regenerate_id(true); 

$reward=$_POST['rewID'];
$choir=$_POST['choID'];
$ID=$_SESSION['accountid'];
$temp = $_SESSION['username'];
$query="SELECT * FROM Users WHERE UserID = $temp";
$result=mysql_query($link,$query);
if (mysqli_num_rows($result) > 0) {
    while($row = mysql_fetch_assoc($result)){
        $pnts=$row['Points'];
    }
}
$count;
$query="SELECT * FROM Rewards where RewardID = $reward";
$result=mysql_query($link,$query);
if ($result->num_rows > 0) {
    while($row = mysql_fetch_assoc($result)){
        $rpnts=$row['RewardValue'];
    }
}
$query="SELECT * FROM Chores where ActivityID = $choir";
$result=mysql_query($link,$query);
if ($result->num_rows > 0) {
    while($row = mysql_fetch_assoc($result)){
        $cpnnts=$row['PointValue'];
    }
}
if(isset($rpnts) || isset($cpnnts)){
    if(isset($cpnnts)){
        $curpntn=$pnts + $cpnnts;
        $sql = "UPDATE Users SET Points = $curpntn WHERE UserID=$temp";
        $link->query($sql);
    } else {
        $sql = "INSERT INTO log ( IDactivity, IDUSER) VALUES ('$choir', '$temp')";
        $link->query($sql);
    }
}
if(isset($rpnts) && ($rpnts<=$pnts)){
    $spentpnts = $pnts - $rpnts;
    $sql = "UPDATE Users SET Points = $spentpnts WHERE UserID=$temp";
    $link->query($sql);
}
$sql = "INSERT INTO log ( 1Drewars, IDUSER) VALUES ('$reward', '$temp')";
$link->query($sql);

header ("Location: http://www.cs.csuk.edu/~scotera/test/wall.php");
Both complete chore and claim reward drop down menus post to transaction.php. As you can see the post variables are turn into variables used to manipulate the queries. Here we have identical structure for both rewards and chores. Both must query and find the post from the drop down menu, if there is none then it is simple blank. If reward is chosen then the queries used will be find the users points then reward points and verify that the user has more points than point value of the reward, if not nothing will occur except a page refresh. The process applies to chore selection with the notable difference that you do not need to compare any point values just add and update point totals.

4.4 Register.php

```php
$sql = "SELECT login FROM Account WHERE login= $login";
$result = mysqli_query($link, $sql);
if ($result->num_rows == 0) {

    $sql = "INSERT INTO Account (Code, login, HashPass, Household VALUES ('Spin', '$login', '$pwd', '$household')";
    $link->query($sql);
    $aid;

    $sql="SELECT * FROM Account ORDER BY AccountID DESC LIMIT 1";
    $result=mysql_query($link,$sql);
    if ($result->num_rows > 0) {
        while($row = mysqli_fetch_assoc($result)){
            $aid = $row['AccountID'];
        }
    }

    $sql = "INSERT INTO Users (AccID,Points, Name) VALUES ('$aid', '$points', '$user')";
    $link->query($sql);
```
The code here inserts the values posted to the registration form and creates a new account, all code is MYSQL escaped to prevent hostile database instruction.

4.5 Top.php

```php
<?php
session_regenerate_id(true);

if (!$_SESSION['active']) {
    echo '<div class="container" style="color:#ff0000">Welcome, Guest!</div>
    <div class="container"><a href="index.php">Login</a> or <a href="Register.php">Register</a></div>
    $_SESSION['now'] = true;
}

else {
    if ($_SESSION['admin']) {
        echo '<div class="container" style="color:#ff0000">Hello, ' . $_SESSION["greeting"] . ')</div>
        <div class="container"><a href="logout.php">Log out</a></div>
    } else {
        echo '<div class="container" style="color:#ff0000">Hello, ' . $_SESSION["household"] . ' Household</div>
        <div class="container"><a href="logout.php">Log out</a></div>'
    }
}
?>
```

This code is a simple script using an if then statement with the session variables to check what display message should be used. If not logged in show greeting while being logged in means show household unless a user has been selected.
4.6 Logout.php

```php
<?php
require_once('util.php');

// this is to clear the cookies and destroy the user session
$_SESSION=array();
if (ini_get("session.use_cookies")) {
    $params = session_get_cookie_params();
    setcookie(session_name(), '', time() - 42000,
               $params["path"], $params["domain"],
               $params["secure"], $params["httponly"]);
}

session_unset();
session_destroy();
// redirect to main page
header("Location: index.php");
?>
```

Once again a simple PHP page that is called when the user clicks logout. This page destroys all session variables and clears browser cookies.
4.7 edit.php

```php
<?php
if (!$_SESSION('active')) {
    echo 'Invalid session.

    <h3>Delete Household Users</h3>
    <div class="content">
        <p>Delete a user by clicking on their name in the sidebar. This will send a GET request to set the curuserid, defaulting to 0 if not specified.

        ```php
        $ID = $_SESSION['ACCOUNTID'];
        $userID = array();
        $count = 0;
        $query = "SELECT * FROM Users WHERE AccID = $ID ORDER BY Name";
        $result = mysqli_query($link, $query);
        if ($result->num_rows > 0) {
            while ($row = mysqli_fetch_assoc($result)) {
                $userID[] = $row['UserID'];
                $temp = $userID[$count];
                if ($_SESSION['greeting'] == $temp) {
                    echo '<li><a href="transaction1.php?delUID=\'.$temp.'">'. $row['Name'] .'\</a></li>';
                }
                $count = $count + 1;
            }
        }
        ```
    </div>
</div>
```
Create.php uses multiple forms that will POST to transaction1.php. transaction.php is called when records are to be updated or completed while transaction1.php is used to insert new chore/rewards/users into the database.
### 4.9 Transaction1.php

```php
$newUser=$_POST['newUser'];
$newPoints=$_POST['newPoints'];
$newCpoints=$_POST['newCpoints'];
$newReward=$_POST['newReward'];
$newDateTime=$_POST['newDateTime'];
$newChores=$_POST['newChores'];
$reward=$_POST['DelrewID'];
$chore=$_POST['DelchoID'];
$id = $_POST['code'];
$ptype = $_POST['integer'];
$temp=$_SESSION['temp'];

$query="SELECT Code FROM Account WHERE AccountID =$temp";
$result=mysql_query($link,$query);
if (mysql_num_rows($result) > 0) {
    while($row = mysql_fetch_assoc($result)) {
        $pin=$row['Code'];
    }
    if($id == $pin) {
        $_SESSION['pin']=true;
    }
}
if(isset($_GET['delUID'])){  
    $temp=$_GET['delUID'];
    $query="DELETE FROM Users Where UserID=$temp";
    $result=mysql_query($link,$query);
    if(isset($_POST['newUser'])){  
        $sql = "INSERT INTO Users (AccID,Points,Name) VALUES ('$temp','$newPoints', '$newUser')";
        $link->query($sql);
    }
    if(isset($_POST['newReward'])){  
        $sql = "INSERT INTO Rewards ( RaccountID,Reward,RewardValue) VALUES ('$temp','$newReward', '$newPoints')";
        $link->query($sql);
    }
    if(isset($_POST['newChores'])){  
        $sql = "INSERT INTO Chores ( CaccountID,Description,PointValue,ChoreTime) VALUES ('$temp', '$newChores', '$newCpoints', '$newDateTime')";
        $link->query($sql);
    }
    if(isset($_POST['DelrewID'])){  
        $query="DELETE FROM Rewards Where RewardID=$reward";
        $result=mysql_query($link,$query);
    }
    if(isset($_POST['DelchoID'])){  
        $query="DELETE FROM Chores Where ActivityID=$chore";
        $result=mysql_query($link,$query);
    }
```
Transaction1.php will set every POST variable from create.php and run through the list of options available. If the variable that the POST was assigned to is not empty, then the if-then statement inside will function.

4.10 Wall.php

This is how most rows are displayed. Example.php and Archive.php use the same process to create the records for their association of chores and rewards.

```php
// Code snippet
```

All main pages have queries set up to have the bottom bar list the latest chores and highest point value users. This is essentially the same style of code as Wall.php body content.
5.1 Update and feedback

After demoing my project on 3/17/2015 I was asked to make changes and corrections. I was tasked with adding a monthly view system for all records and creating a text output that the user can download with a history of all records. The work took some time but I was able to make the changes.

5.2 Make views use month.

```php
<form name="classList" action="" method="POST">
<select name="month" id="month" onchange="" size="1">
    <option selected disabled hidden value=""></option>
    <option value="01">January</option>
    <option value="02">February</option>
    <option value="03">March</option>
    <option value="04">April</option>
    <option value="05">May</option>
    <option value="06">June</option>
    <option value="07">July</option>
    <option value="08">August</option>
    <option value="09">September</option>
    <option value="10">October</option>
    <option value="11">November</option>
    <option value="12">December</option>
</select>
<input type="submit" value="Display Range"> </form>

$time = strtotime($row['CHoccurrence']);
$myFormatForView = date("m/d/y g:i A", $time);
$newdate=strstr($myFormatForView, '-16,');
if($_POST['month'] == $newdate{
    echo '<section>
    echo '<span>', $row['Description'], '</span>','
    echo '</section>
    
    echo '<section>
    echo '<span>', $myFormatForView, '</span>','
    echo '</section>
    
    echo '</section>

The code listed here on the left is a drop down box that the user can select which month of records to view, the default value in pages is the current month. The code on the right first changes the Sql time format to something more manageable. Next it creates a new variable using the time stamp that include only the numbers for the month. If the record matches the POST variable of month then it will be displayed. The code in another area also uses date() php function to grab the current month and
5.3 Let the user download a text file with full records.

```php
myfile = fopen("text/$house.txt", "x") or die("Unable to open file!");

fwrite($myfile, $spent);

$query="select * from Rewards where RaccountID = $ID";
$result=mysql_query($link,$query);
if ($result->num_rows > 0) {
    while($row = mysql_fetch_assoc($result)){
        $row["Reward"];
        $row["RewardValue"];
        $txt[] = "Reward $a costs a point value of $b\n";
        fwrite($myfile, $txt[$counti]);
        $counti=$counti+1;
    }
}

fclose($myfile);

chmod("text/$house.txt", 0744);
```

The next request was to enable a user to download the entire account history. This was done by opening a file and writing to it with multiple queries. The style remains the same for all 5 major queries, in which an array is formatted with one SQL rows worth of data then pushed onto an array. This array is then wrote back to the file. This continues on until fclose is called. The file also must have its permissions changed in order for the user to download it. The download is simply a link from the left toolbar menu. All histories are saved to a text folder with the individual names that
were used for the households.

5.4 Make a Household report

The code here makes php calls to different queries based on joins. The data is derived from the entire account and lets the user choose specifically what they want to view. It is the most comprehensive view of the account I have made.
I tried to make Asynchronous submits with jquery but was unable to get the results I wanted. Therefore I decided to go with server-side scripting.

### 6.1 Difficulties with project

I originally planned for this to be a self contained android application with fully functioning calendar. I may have bit off more then I can chew. On the positive side I did manage to work in android and produce basic levels of functionality, so now I have a frame of reference for my next project.

I had troubles with some java script and Ajax functions so I had to cut them out and use PHP with clever solutions. I am very interested now in trying asp due to the praise I’ve heard from classmates when working on their databases. I most forward look to having more experience in web page programming and will attempt to learn other languages like the aforementioned asp.

More than once I was stumped for hours due to multiple echo statements to run HTML in a purely PHP block, especially concerning session variables and escaped characters.

### 6.2 Triumphs of the Project

The database was built to make sure it follow the rules of normalization and
every effort was made to have data that made sense and was effective in its simplicity. I was able to convey all the ideas that I wanted for the program. Each time a new feature came up I found a way to incorporate it a timely fashion. I know that without the cs342 class and some of cs442 I would not have been able to make an effective database that was exactly suited to my needs and expectations.

The website itself was a good success, far better than last semester's attempt. I am very relieved that a project that seem incredibly scary just 3 months ago was now something I could sit down and plan for. The complete user control for how to set up chores, points, rewards and time was probably my favorite achievement. I had a clear goal in my head and tailor made the steps along the way to reach that goal.

The best thing about this project is how I can improve upon it in my spare time, I would like to take this to a commercial level. I feel that with the right type of look I can create this site to rival the best chore managers out there.

6.3 Project significance

As stated earlier in the report, people are moving more and more to digitally automating almost all list based tasks. I believe with some retooling and extra features I can legitimately make this a add based service and then expand up towards a subscription or premium model. The part in which I did develop in the android studio also expanded my skills and made hungry for more of a challenge. I will continue to make marketable apps to family who mostly likely would have the use for them.
6.4 What I have learned

I learned mostly that anything takes time and diligence, The amount of effort spent will always reflect the quality show. It also takes far more work to be on a solo project then with others. For this I will always be glad to work in a group setting. As this is my senior project and I will be graduating CSUB this winter, I think the most important less learned from all this is the ability to work with people and without people. I know moving forward most jobs will have a heavy group work environment. Succeeding previously with a group and now as a solo project, I know I can handle anything out there in the workforce today.