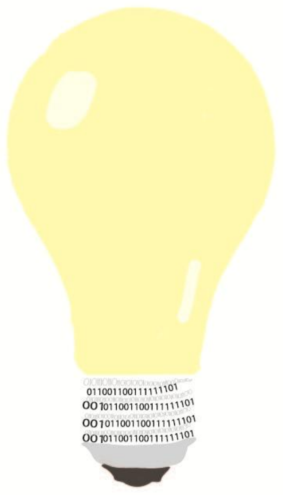
Acceptance Test

Requested By:
Dr. Darren Lim
Associate Professor
Siena College
Department of Computer Science

Hobby Information Tracker
Hobb-IT



By

Illumination Technologies

Prepared By:
Karl Appel
Connor Blakely
Jackie Hausmann
Bryan Leicht
Katie Sitaro

1. Introduction

1.1. Product Overview and Summary

As online shopping becomes an ever expanding market, consumers are continually faced with a variety of new vendors offering the same merchandise. More than ever, hobby enthusiasts are employing the internet to buy and sell their goods. Illumination Technologies will develop a program which can provide updates on changes in prices of *Magic: The Gathering* playing cards. Illumination Technologies will scrape data from a list of online websites given to Illumination Technologies by Dr. Lim. The system to be able to store and track the prices of the *Magic: The Gathering* cards from day to day. The system, the Hobby Information Tracker (Hobb-IT), will also be able to compare how the price of one card varies between retailers. Hobb-IT will be able to keep track of what cards each user has already purchased, searched for, or hopes to track.

2. Requirements Inventory

2.1. User Case Narratives

2.1.1. *Administrator* Use Case Narrative

An *Administrator* account is a built-in user in the *Hobby Information Tracker* (Hobb-IT). There is only one *Administrator* account for this system. The *Administrator* will be able to monitor and update the system. The *Administrator* will determine how to parse a website's content. The *Administrator* will be able to access the search history of all users. The *Administrator* will be able to access the database to see and to change the login credentials of all users. The *Administrator* will be able access all of the stored data.

2.1.2. *Advanced User* Use Case Narrative

An *Advanced User* is someone who has been assigned a username and password by the *Administrator*. The *Advanced User's* account will be used to login into the system. Once logged into the system, an *Advanced User* will be able to search for the real-time price of a *Magic: the Gathering (MTG)* card from any tracked website. The *Advanced User* will have saved lists of previously tracked *MTG* cards, already bought *MTG* cards, and *MTG* cards the *Advanced User* wishes to track in the future. The *Advanced User* will have the option to modify what *MTG* cards are on each list. The *Advanced User* will have the ability to access any previous searches requested from the system associated with the *Advanced User's* account. The *Advanced User's* will be able to view a visual representation of the fluctuations in prices of a *MTG* card.

2.1.3. *Guest User* Use Case Narrative

A *Guest User* is defined as any person who is not an *Advanced User* or *Administrator*. The *Guest User* will be able to access some features of Hobb-IT without logging in to the system. The *Guest User* will be able to lookup the real-time price and condition of a *MTG* card. The *Guest User* is limited to viewing the price and condition of a *MTG* card to one tracked website at a time.

2.2. Functional Requirements Inventory

General

- Hobb-IT will be compatible with current versions of Chrome, Firefox, Internet Explorer, and Safari.

Administrator

- Will be able to access all stored data on the database.
- Will be able to access the search history of all users.
- Will be able to approve usernames and passwords of new Advanced User accounts.
- Will be able to change how a website's data is parsed into the database.
- Will be able to view and change login credentials of all users.
- Will be able to clear the database history of past searches

Advanced User

- Will be able to login to Hobb-IT using a username and password approved by the Administrator.
- Will be able to search for the real-time price of a *Magic: The Gathering* card from any tracked website.
- Will be able to save a list of tracked *Magic: The Gathering* cards.
- Will be able to save a list of purchased *Magic: The Gathering* cards.
- Will be able to save a list of *Magic: The Gathering* cards the user wishes to track in the future.
- Will be able to edit the *Magic: The Gathering* cards that appear on any list.
- Will be able to access the search history associated with the Advanced User's account.
- Will be able to view a visual representation of the fluctuations in prices of a tracked *Magic: The Gathering* card.

Guest User

- Will be able to access Hobb-IT without login credentials.
- Will be able to view the real-time price and condition of a *Magic: The Gathering* card from one website at a time.

2.3. Non Functional Requirements Inventory

- Hobb-IT will be easy to maintain.
- Hobb-IT will be efficient.
- Hobb-IT will be stable.
- Hobb-IT will be user friendly.
- Hobb-IT will follow the legal processes on all websites tracked.

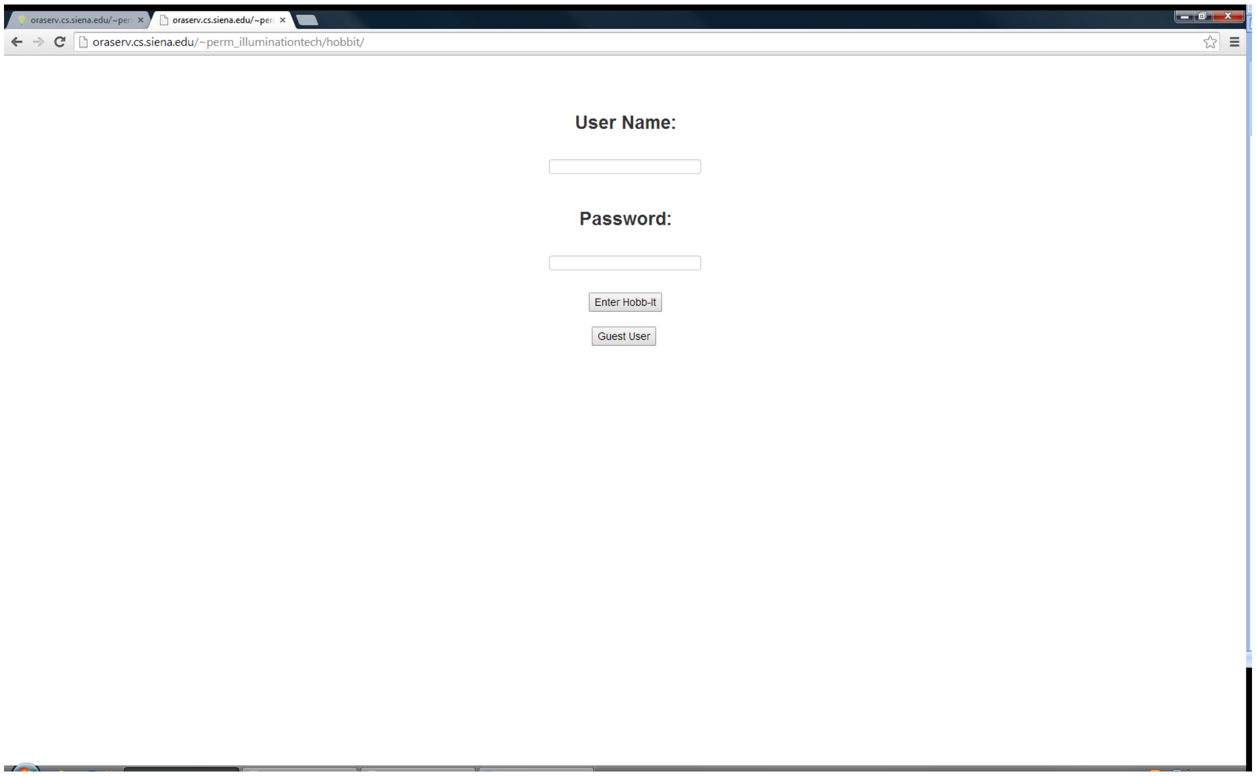
2.4. Future Enhancements

Currently, there are no future enhancements (possible additions or alterations to the application) planned for Hobb-IT.

3. External Design Specification

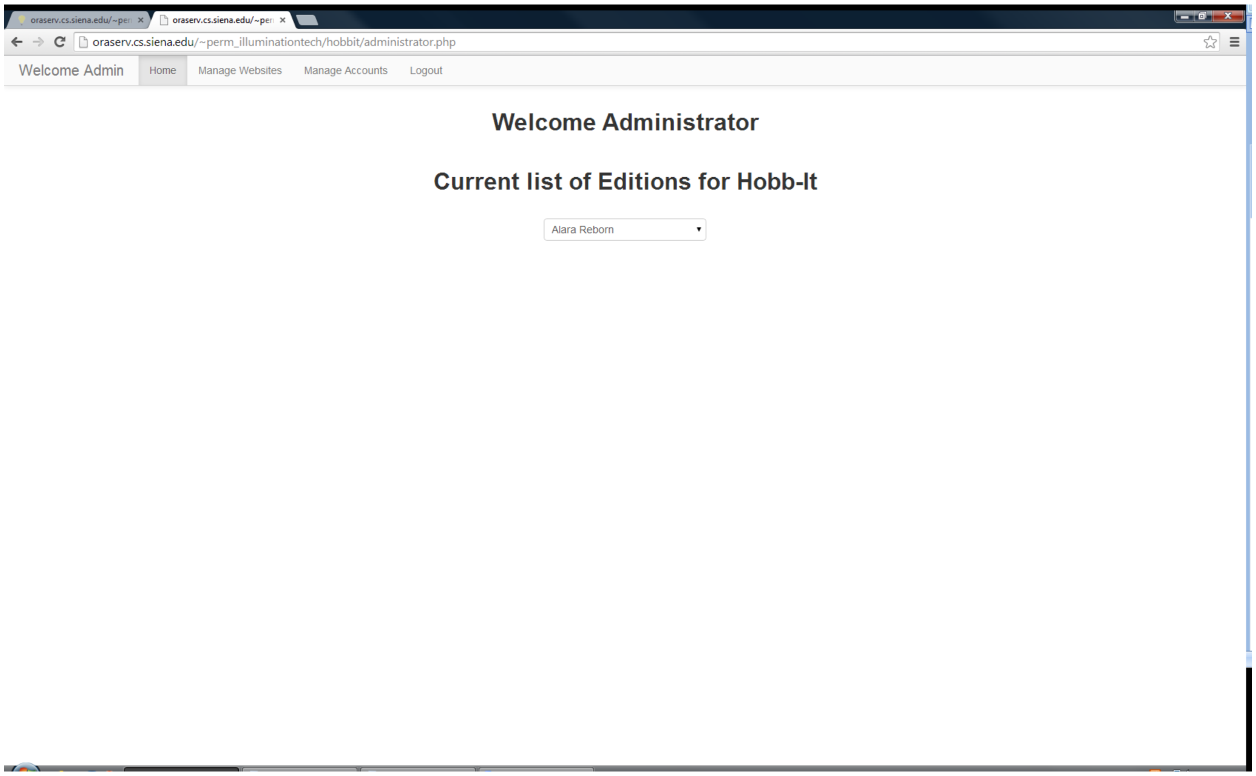
3.1. User Displays

3.1.1. Home Screen



3.1.2. Administrator Screens

3.1.2.1. Administrator Home



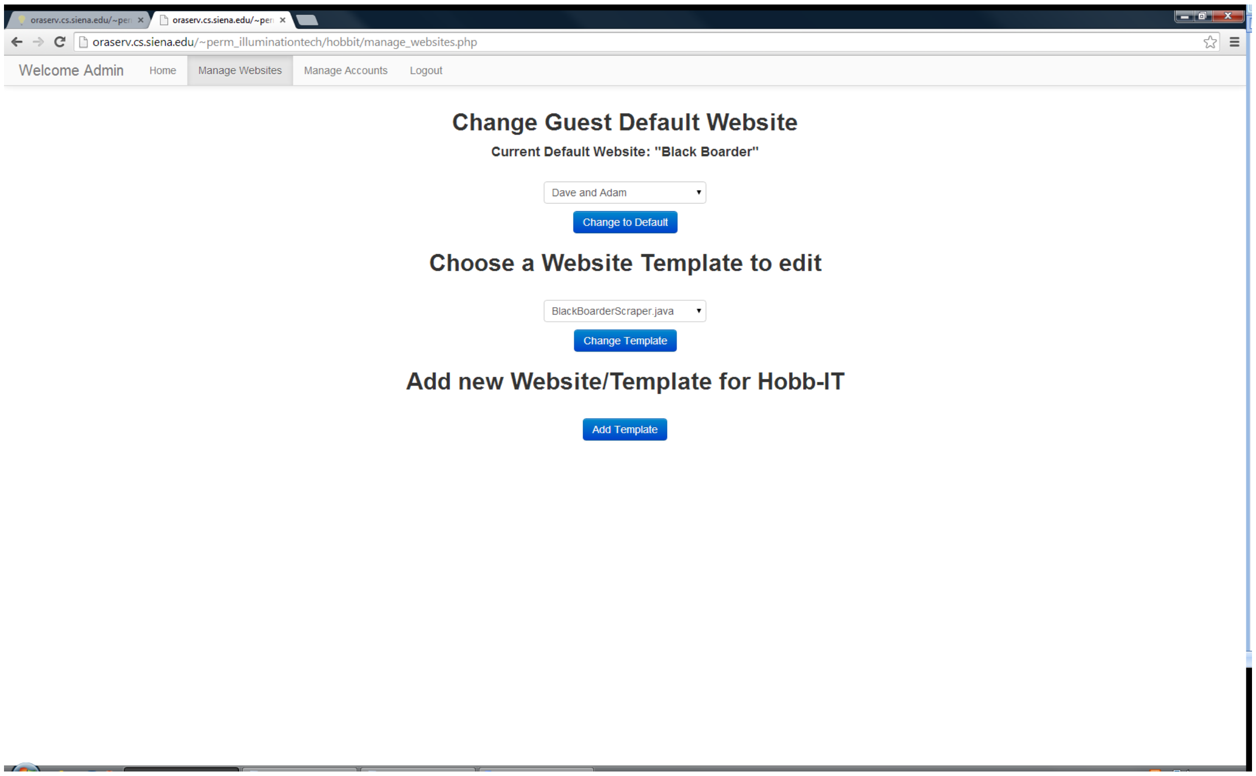
oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/administrator.php

Welcome Administrator

Current list of Editions for Hobb-It

Alara Reborn

3.1.2.2. Manage Websites



oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/manage_websites.php

Change Guest Default Website

Current Default Website: "Black Boarder"

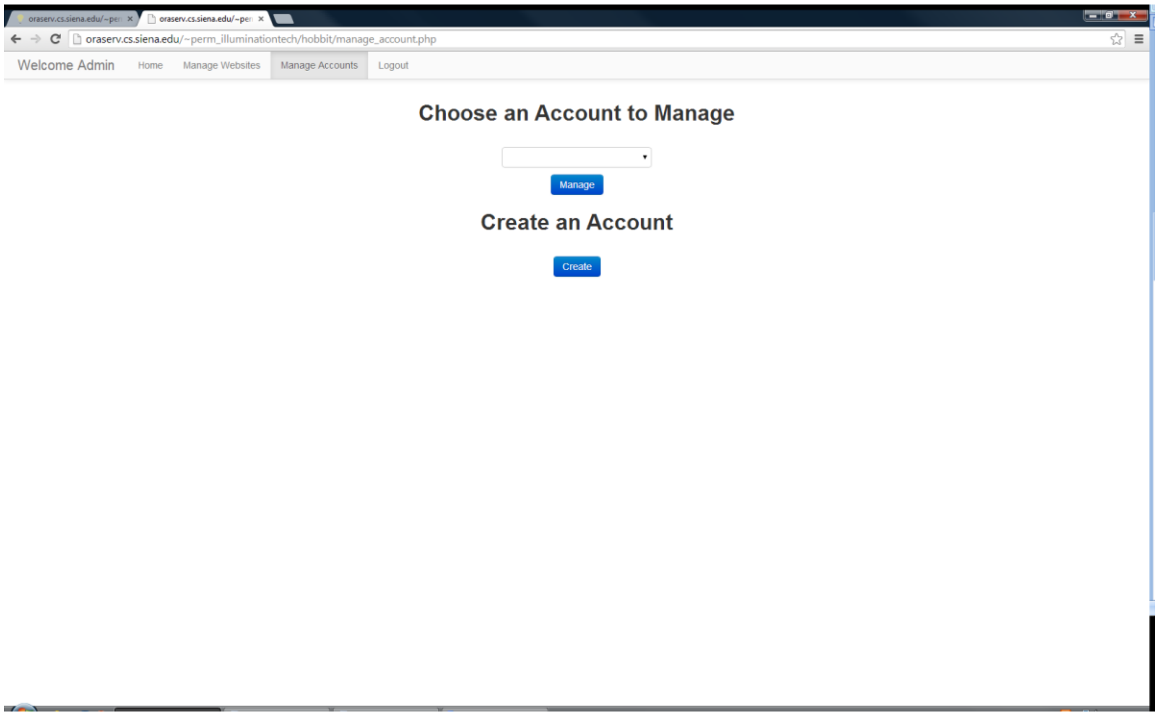
Dave and Adam

Choose a Website Template to edit

BlackBoarderScraper.java

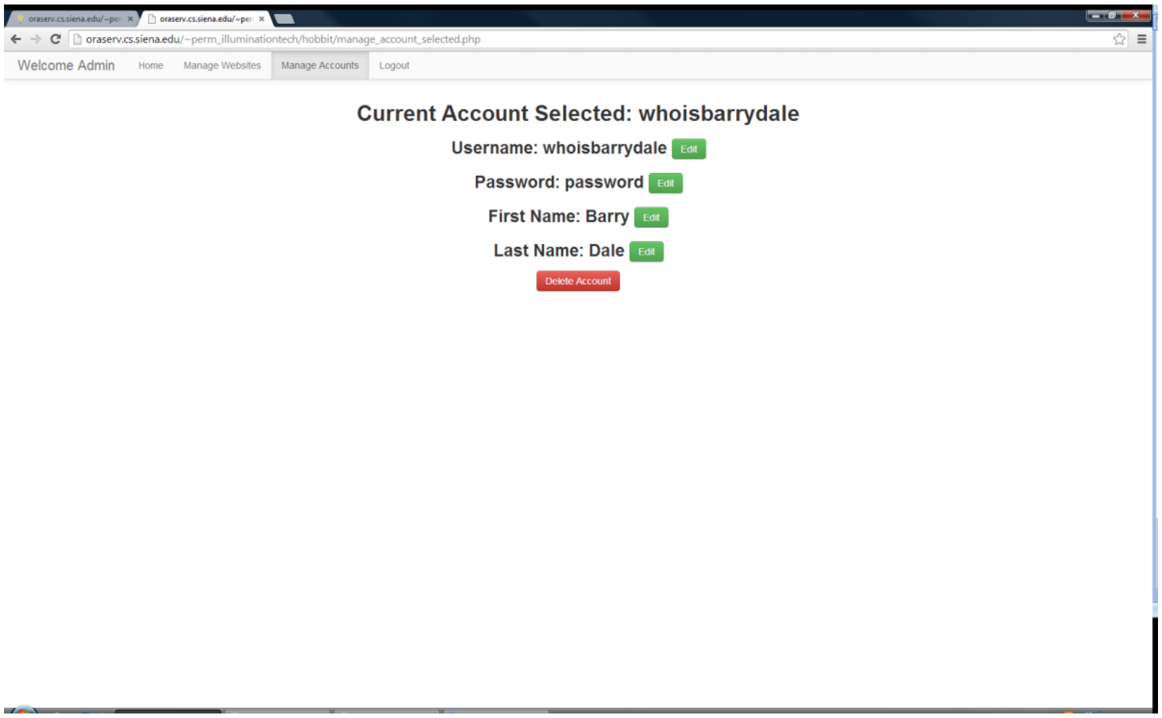
Add new Website/Template for Hobb-IT

3.1.2.3. Manage Accounts


oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/manage_account.php
Welcome Admin
Home
Manage Websites
Manage Accounts
Logout
Manage
Create

Choose an Account to Manage

Create an Account



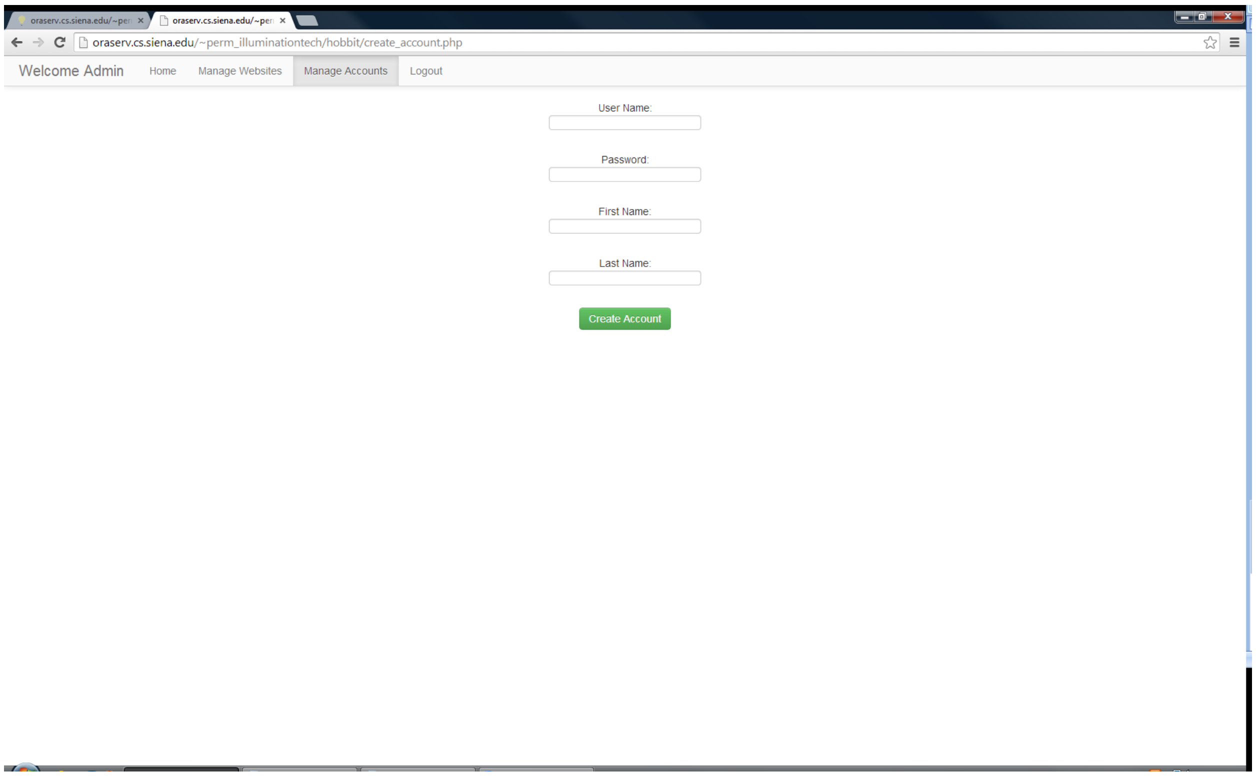
Current Account Selected: whoisbarrydale

Username: whoisbarrydale

Password: password

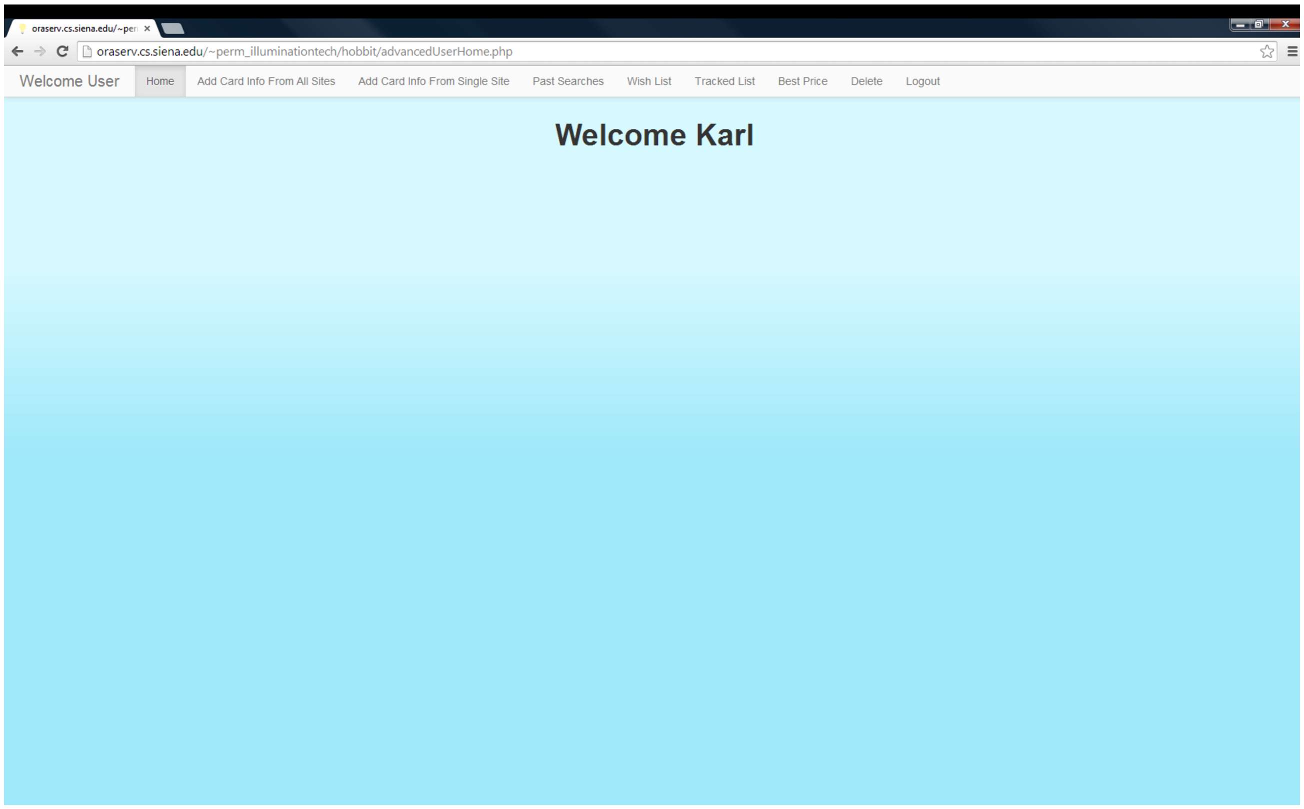
First Name: Barry

Last Name: Dale


oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/create_account.php
Welcome Admin
Home
Manage Websites
Manage Accounts
Logout
User Name:
Password:
First Name:
Last Name:
Create Account

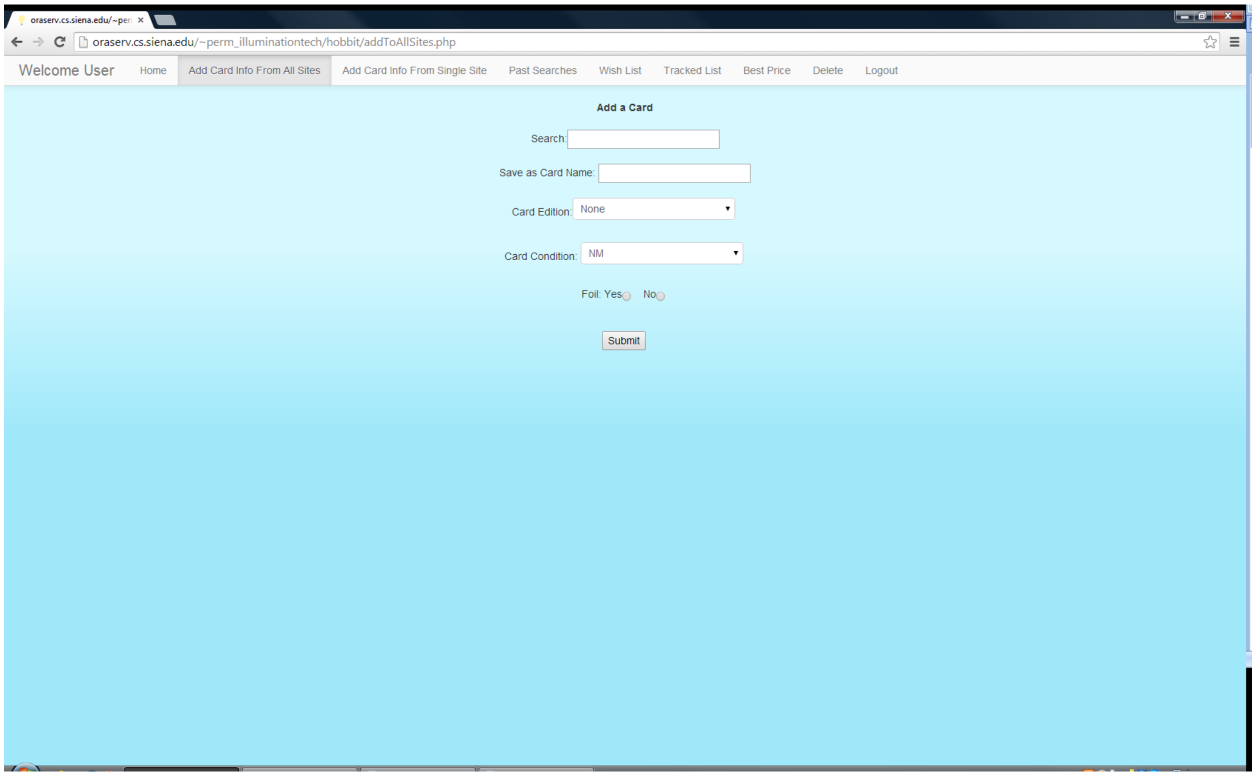
3.1.3. Advanced User Screens

3.1.3.1. Advanced User Home


oraserv.cs.siena.edu/~per
oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/advancedUserHome.php
Welcome User
Home
Add Card Info From All Sites
Add Card Info From Single Site
Past Searches
Wish List
Tracked List
Best Price
Delete
Logout

Welcome Karl

3.1.3.2. Add Card Info From All Sites


oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/addToAllSites.php
Welcome User
Home
Add Card Info From All Sites
Add Card Info From Single Site
Past Searches
Wish List
Tracked List
Best Price
Delete
Logout

Add a Card

Search:

Save as Card Name:

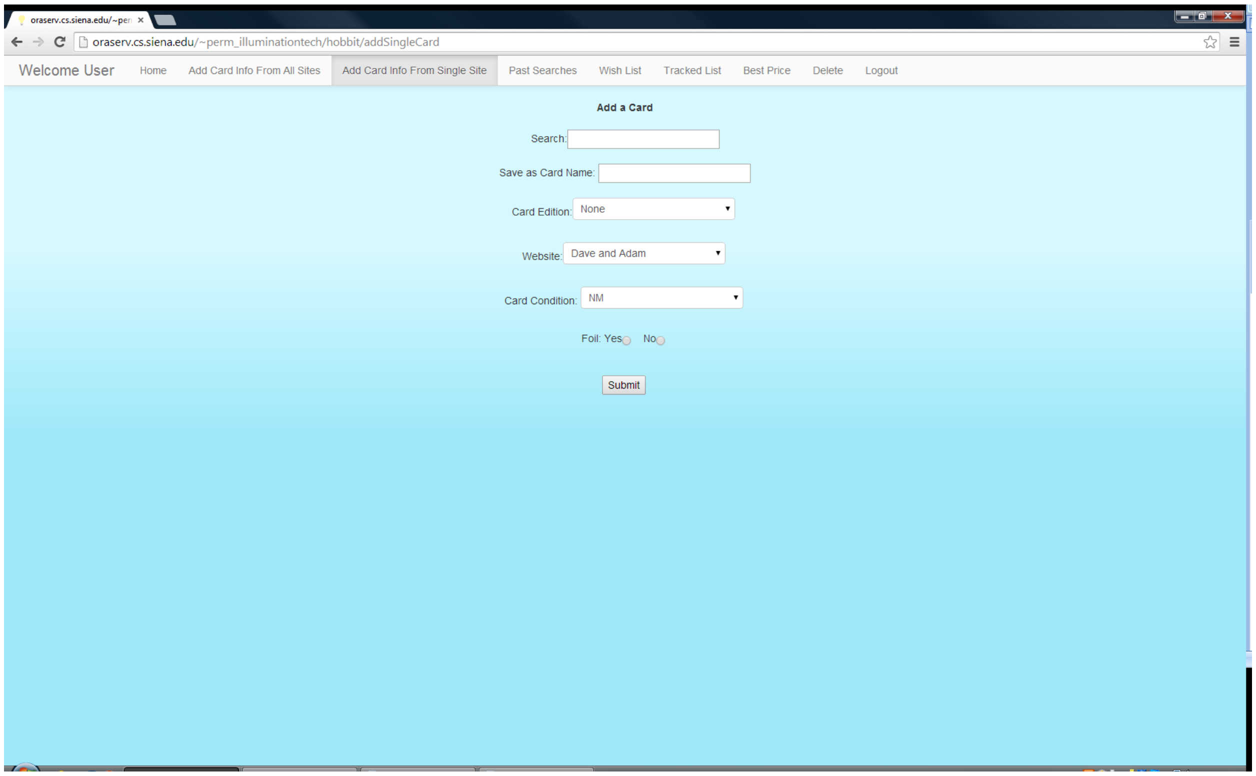
Card Edition: None

Card Condition: NM

Foil: Yes No

Submit

3.1.3.3. Add Card Info From Single Site


oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/addSingleCard
Welcome User
Home
Add Card Info From All Sites
Add Card Info From Single Site
Past Searches
Wish List
Tracked List
Best Price
Delete
Logout

Add a Card

Search:

Save as Card Name:

Card Edition: None

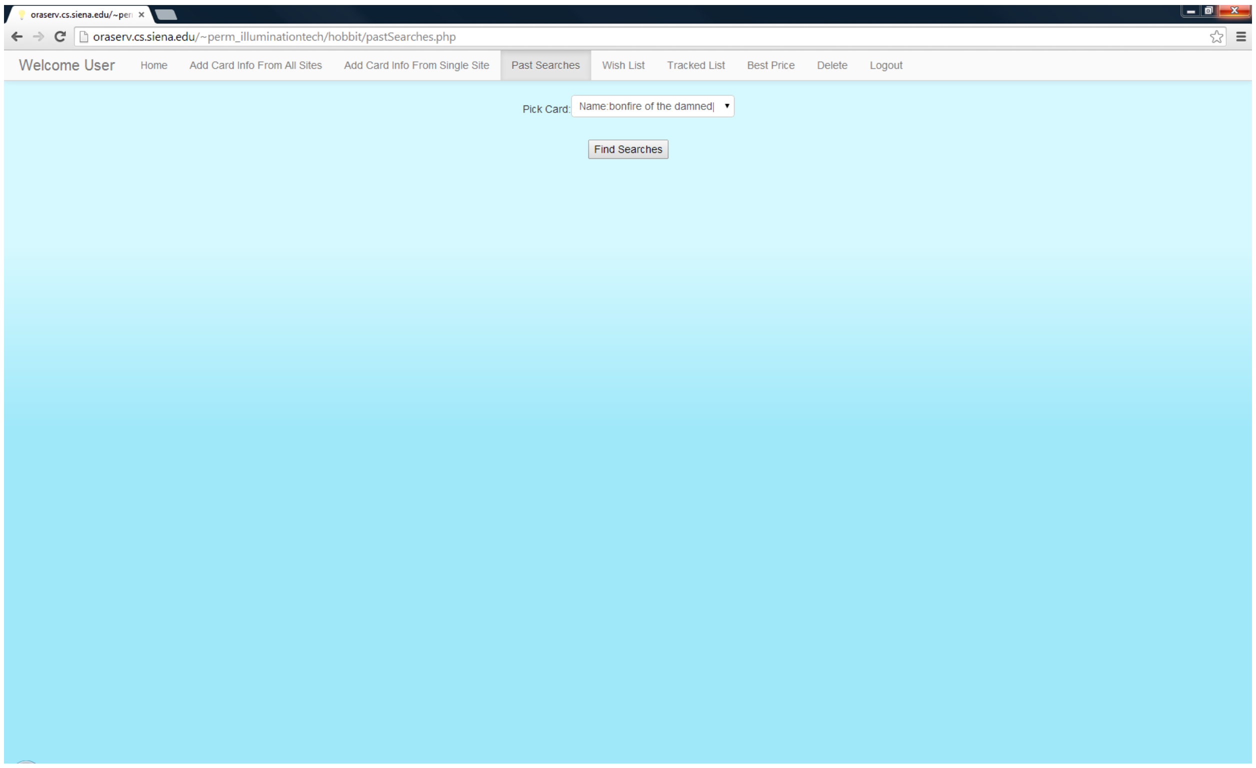
Website: Dave and Adam

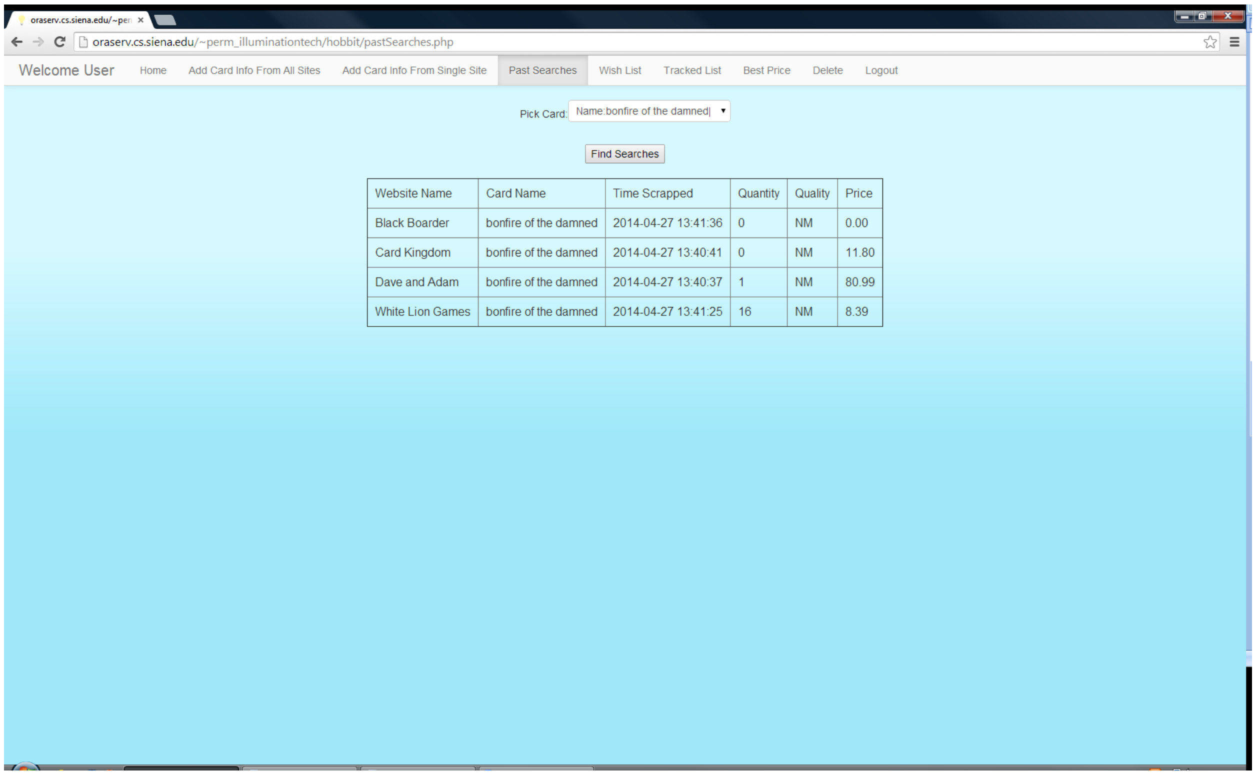
Card Condition: NM

Foil: Yes No

Submit

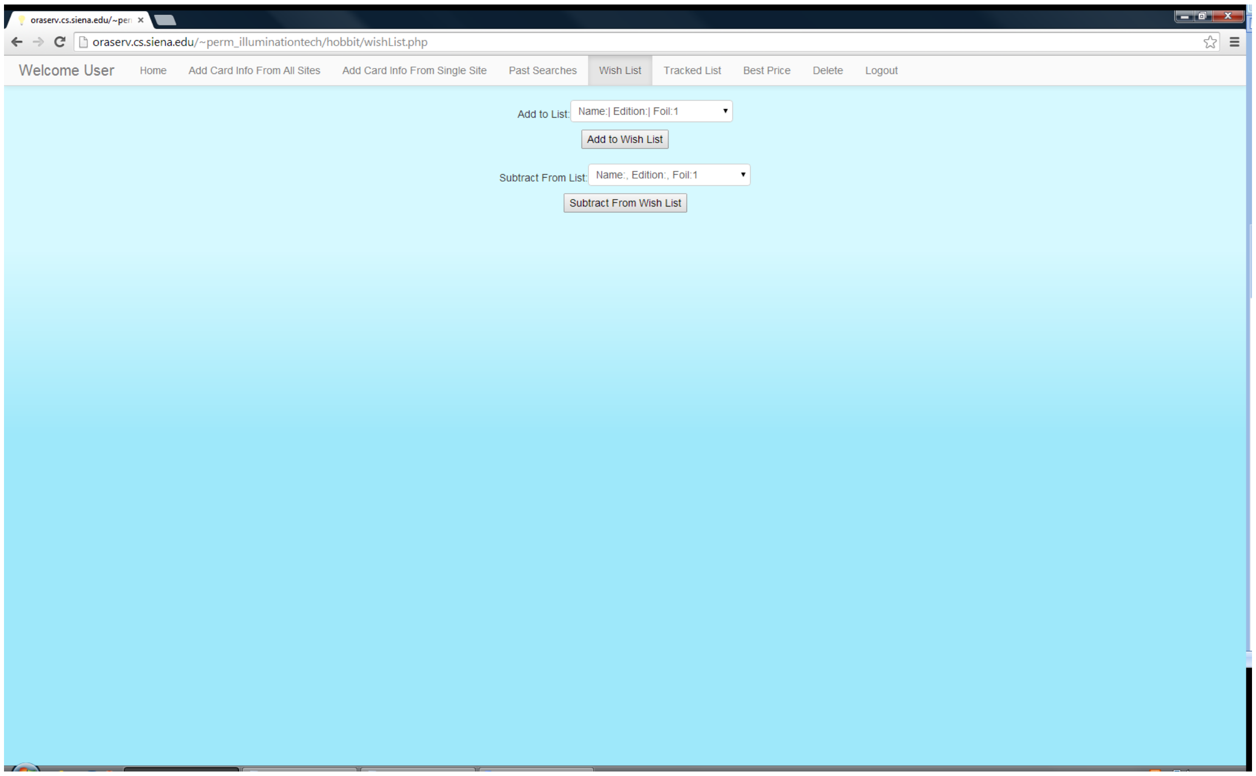
3.1.3.4. Past Searches


oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/pastSearches.php
Welcome User
Home
Add Card Info From All Sites
Add Card Info From Single Site
Past Searches
Wish List
Tracked List
Best Price
Delete
Logout
Pick Card: Name:bonfire of the damned|
Find Searches


oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/pastSearches.php
Welcome User
Home
Add Card Info From All Sites
Add Card Info From Single Site
Past Searches
Wish List
Tracked List
Best Price
Delete
Logout
Pick Card: Name:bonfire of the damned|
Find Searches

Website Name	Card Name	Time Scrapped	Quantity	Quality	Price
Black Boarder	bonfire of the damned	2014-04-27 13:41:36	0	NM	0.00
Card Kingdom	bonfire of the damned	2014-04-27 13:40:41	0	NM	11.80
Dave and Adam	bonfire of the damned	2014-04-27 13:40:37	1	NM	80.99
White Lion Games	bonfire of the damned	2014-04-27 13:41:25	16	NM	8.39

3.1.3.5. Wish List

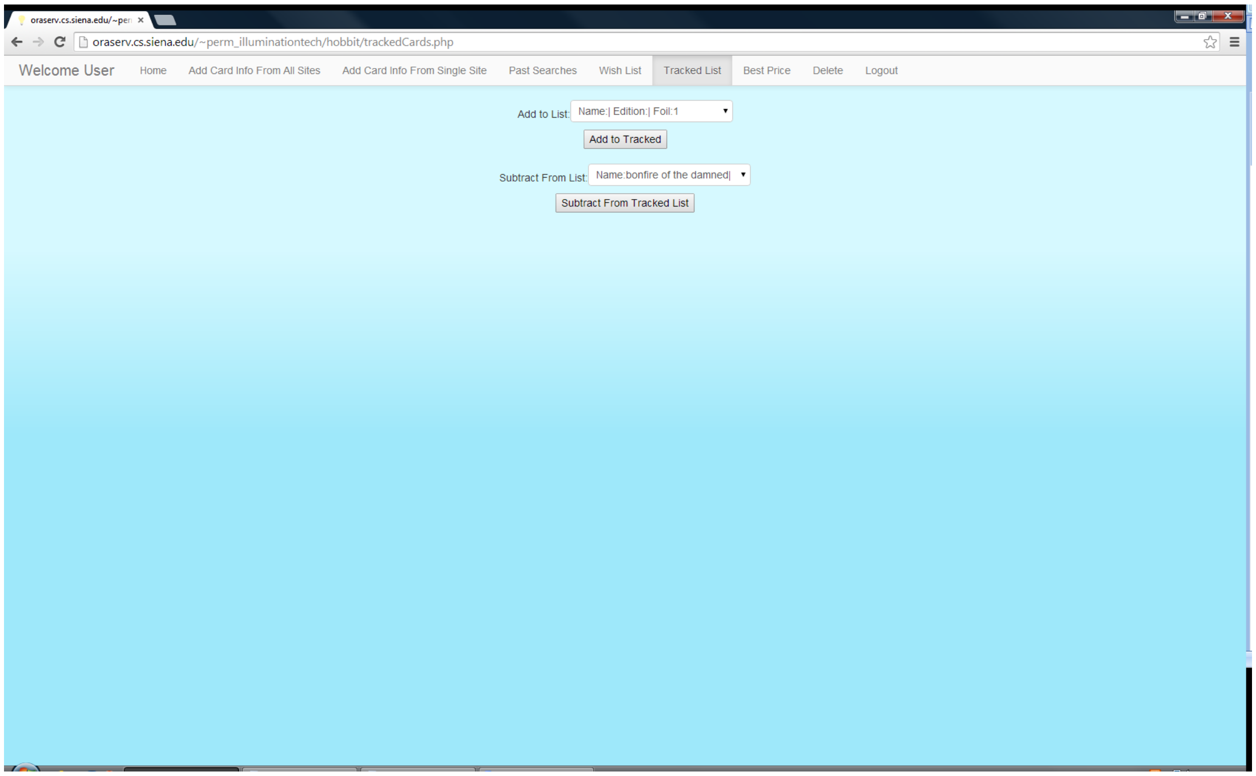


oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/wishList.php

Add to List: Name:| Edition:| Foil:1

Subtract From List: Name:, Edition:, Foil:1

3.1.3.6. Tracked List

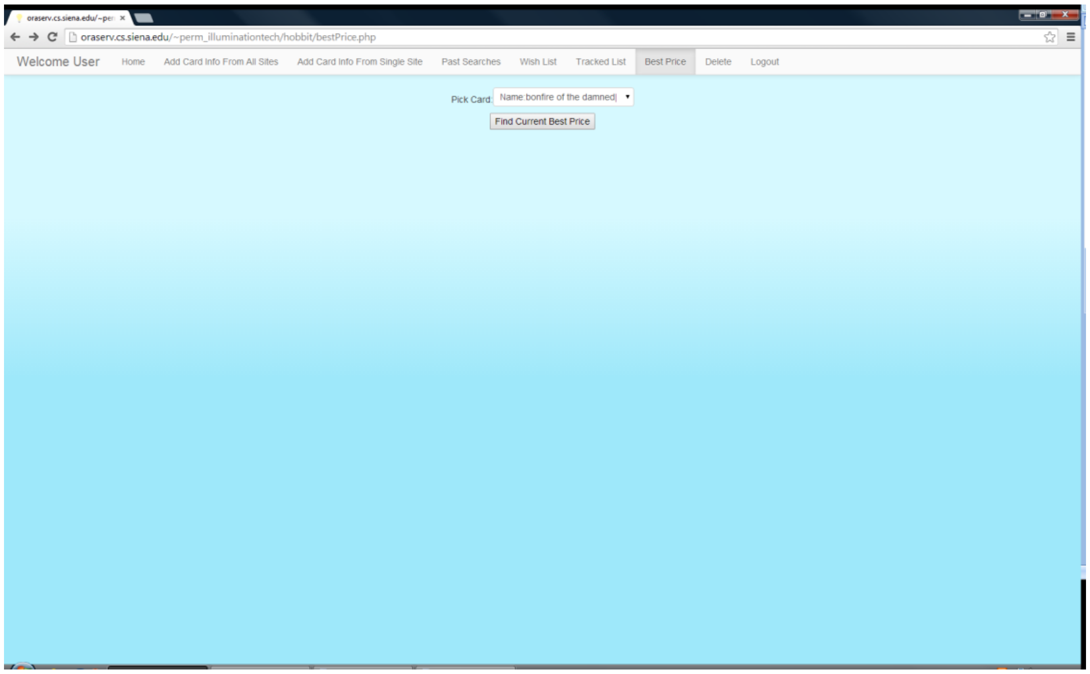

oraserv.cs.siena.edu/~per
Welcome User
Home
Add Card Info From All Sites
Add Card Info From Single Site
Past Searches
Wish List
Tracked List
Best Price
Delete
Logout
Add to Tracked
Subtract From Tracked List

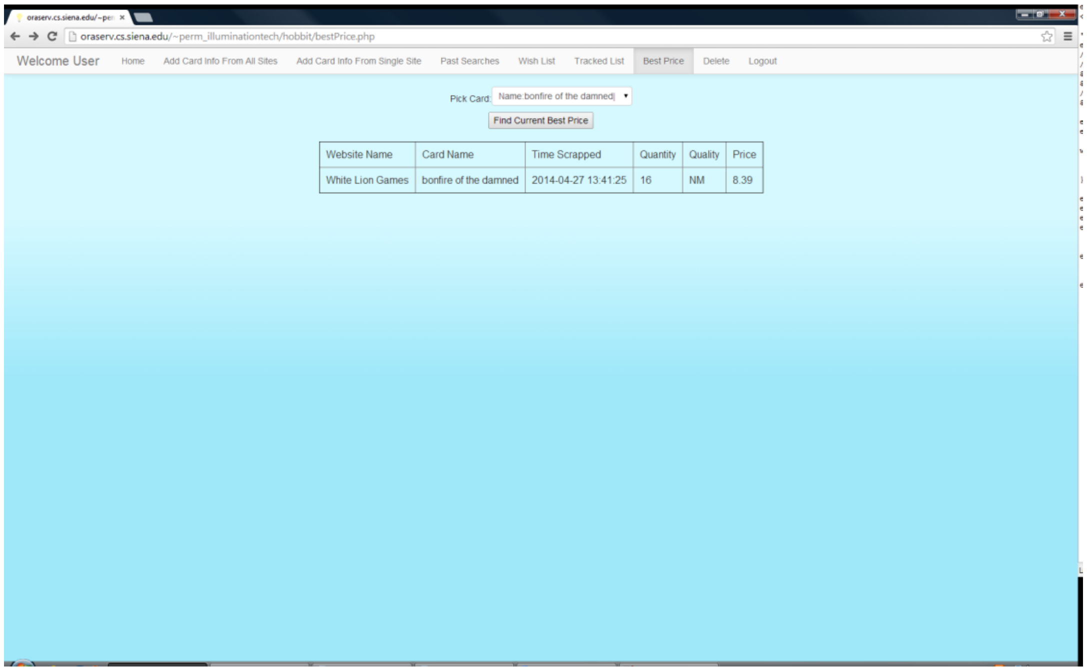
oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/trackedCards.php

Add to List: Name:| Edition:| Foil:1

Subtract From List: Name:bonfire of the damned|

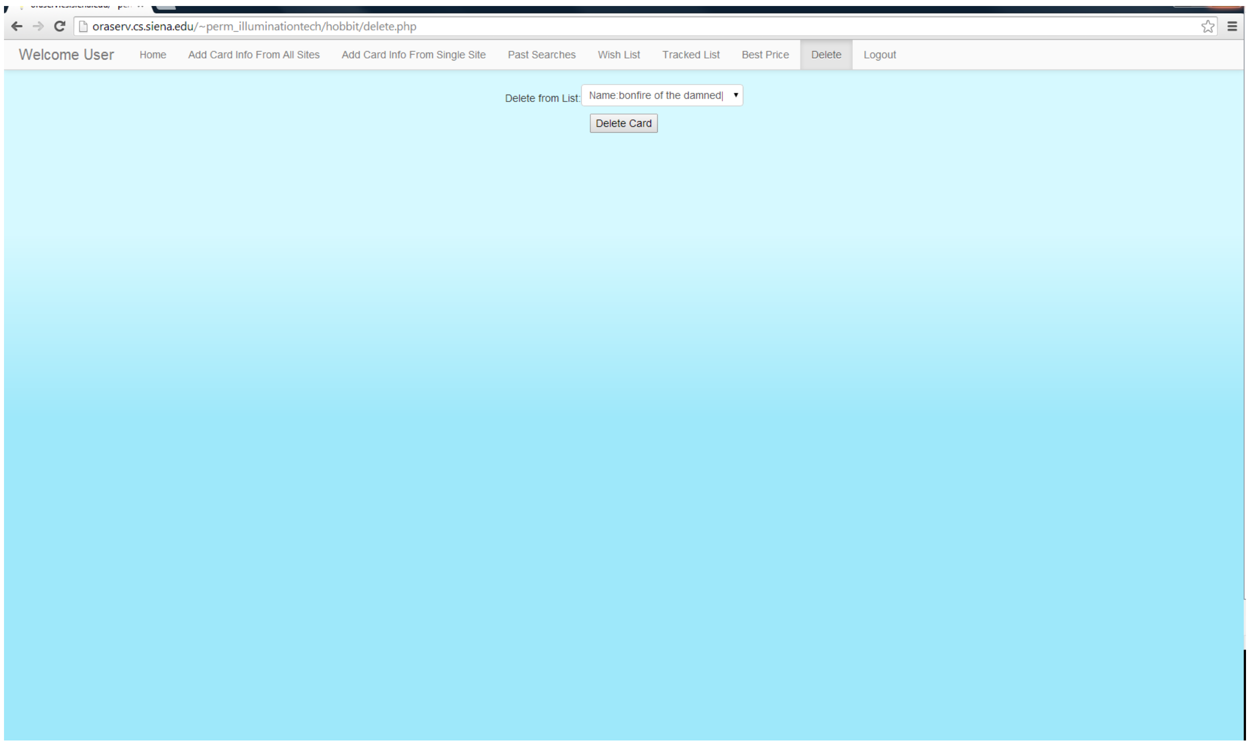
3.1.3.7. Best Price


oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/bestPrice.php
Welcome User
Home
Add Card Info From All Sites
Add Card Info From Single Site
Past Searches
Wish List
Tracked List
Best Price
Delete
Logout
Pick Card: Name: bonfire of the damned|
Find Current Best Price


oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/bestPrice.php
Welcome User
Home
Add Card Info From All Sites
Add Card Info From Single Site
Past Searches
Wish List
Tracked List
Best Price
Delete
Logout
Pick Card: Name: bonfire of the damned|
Find Current Best Price

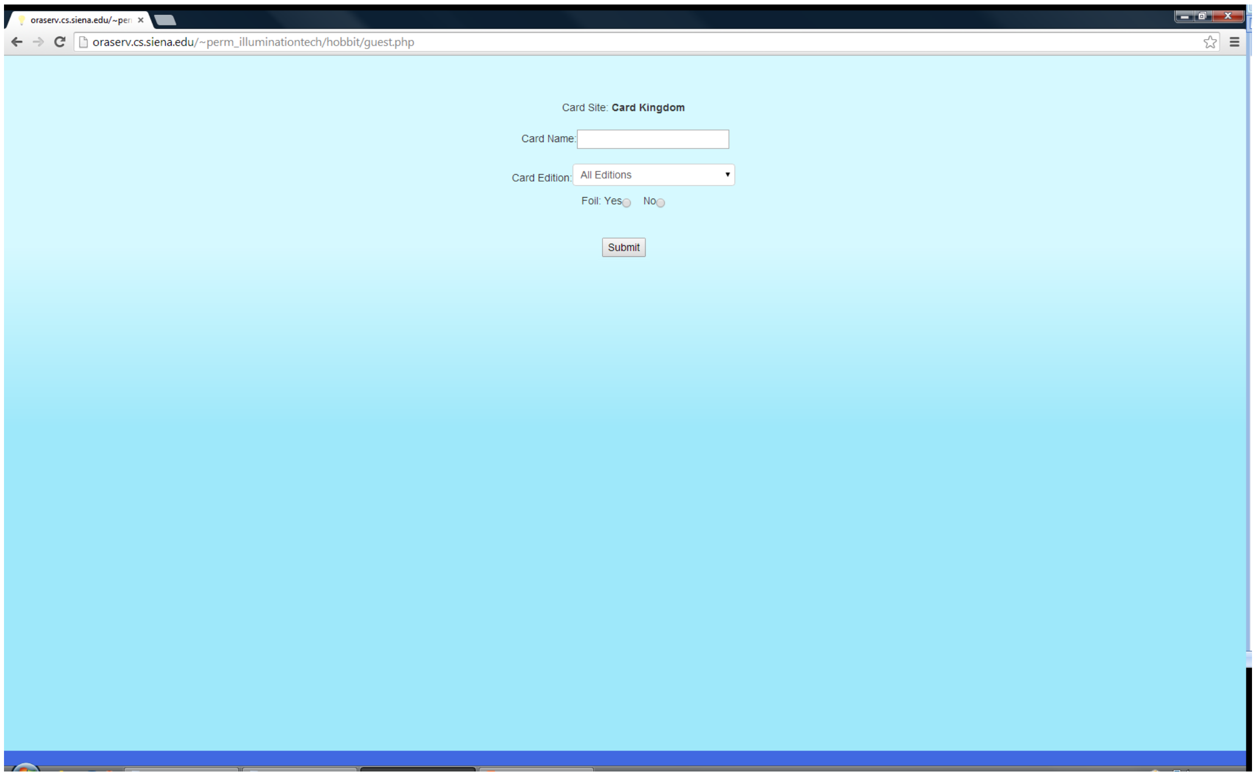
Website Name	Card Name	Time Scrapped	Quantity	Quality	Price
White Lion Games	bonfire of the damned	2014-04-27 13:41:25	16	NM	8.39

3.1.3.8. Delete


oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/delete.php
Welcome User
Home
Add Card Info From All Sites
Add Card Info From Single Site
Past Searches
Wish List
Tracked List
Best Price
Delete
Logout
Delete from List:
Name:bonfire of the damned|
Delete Card

3.1.4. Guest User Screens

3.1.4.1. Guest User Home Screen



oraserv.cs.siena.edu/~perm_illuminationtech/hobbit/guest.php

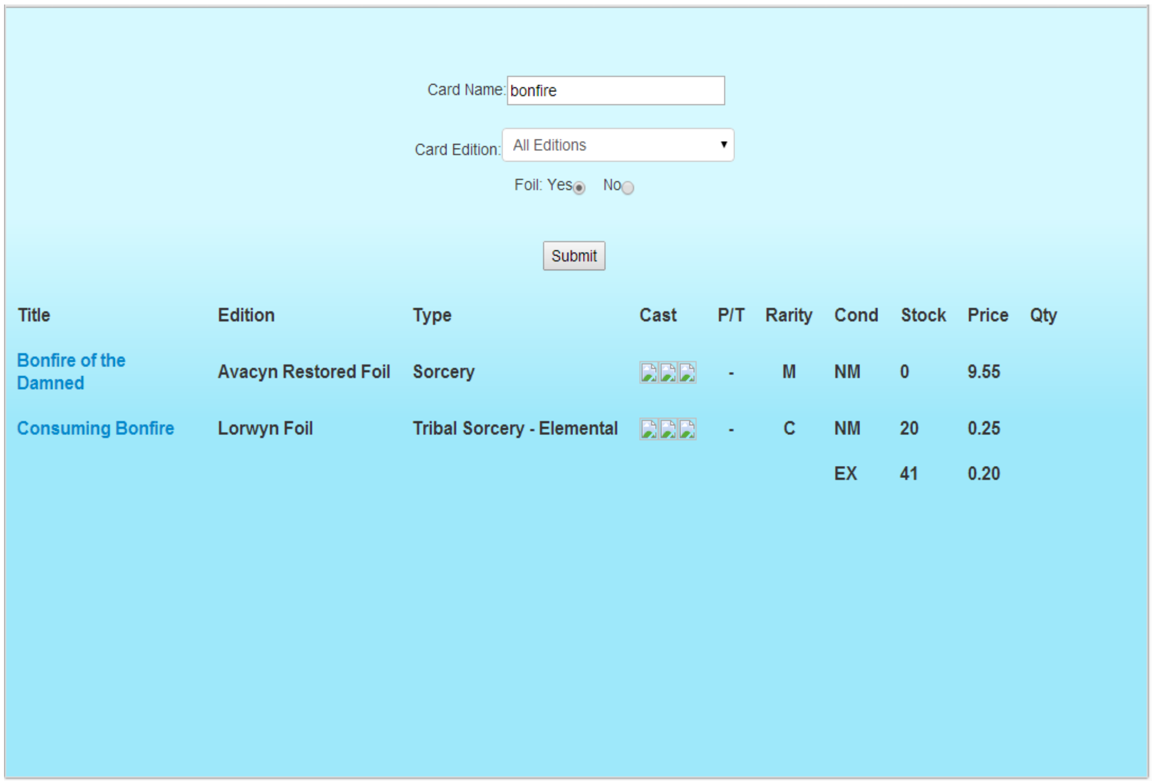
Card Site: **Card Kingdom**

Card Name:

Card Edition: All Editions

Foil: Yes No

3.1.4.2. Guest User Search Results Screen



Card Name: bonfire

Card Edition: All Editions

Foil: [✓ Yes] No

Title	Edition	Type	Cast	P/T	Rarity	Cond	Stock	Price	Qty
Bonfire of the Damned	Avacyn Restored Foil	Sorcery		-	M	NM	0	9.55	
Consuming Bonfire	Lorwyn Foil	Tribal Sorcery - Elemental		-	C	NM	20	0.25	
						EX	41	0.20	

3.2. Logical Data Dictionary

See attached Appendix B for full dictionary

Definition: a collection of data entities such that the name, definition, type, size, description, and acceptability of each data entity is defined

Key

Data Name: The name of the data entity stored within the system

Applicable to: The screens for which this data entity will be used

Data Type: The type of data for each data entity

Data Size: The length or size for the data entity's type

Description: A explanation of how and what this data entity will store

Acceptable Input: This explains the limits of appropriate input in order for data to be accepted

Good Example of Input: An example accepted and stored by the system

Bad Example of Input: An example of data not stored and rejected by the system

Notes: Other information pertaining to the data in the system

3.3. Logical Format of Data Files and Databases

3.3.1. ER Diagram

An E-R diagram is a visual representation of a database. This diagram consists of several items: entities, relationships and attributes. An entity is anything in the database that can be uniquely identified and can exist on its own. A relationship is a connection between two entities that describes how the two entities interact. An attribute is any data item that is connected to an entity or a relationship. A primary key attribute(s) is the attribute(s) that uniquely identify the entity.

Our ER Diagram can be viewed in Appendix C at the end of this document.

3.3.2. Database Tables

Website:

Website Name	Primary Key, varChar
Address	varChar
Template ID	Foreign Key, Unique, Not Null, int
Guest User Default	boolean

Template:

Template ID	Primary Key, int
Previous Template Id	Foreign Key, Unique, int
Template Content	varChar
Reload State	boolean

Admin:

Admin User Name	Primary Key, varChar
User Password	varChar
First Name	varChar
Last Name	varChar

Magic Card:

ID	Primary Key, int
Card Name	varChar
Card Edition	varChar
Foil	boolean
Misc Description	varChar

Advanced:

User Name	Primary Key, varChar
User Password	varChar
First Name	varChar
Last Name	varChar

Search:

Website Name	Primary Key, Foreign Key, varChar
ID	Primary Key, Foreign Key, int
Card Time Scraped	Primary Key, Time
Date Scraped	Primary, Date
Quantity	Int
Quality	varChar
Price	Int

On List:

User Name	Primary Key, Foreign Key, varChar
ID	Primary Key, Foreign Key, int
Wish List	boolean
Tracked	boolean
Bought	boolean
Past	boolean

4. Architectural Design Specification

4.1. Development & Production Environments

4.1.1. Development Environment

The development environment of Hobb-IT is the environment which Illumination Technologies will develop the software and test the software. The production environment is the environment where Hobb-IT will be tested and will be put into production. The Development Environment includes a Macintosh Computer, a Windows Computer running Vista, and a Windows Computer running Windows 8. The Production Environment includes the server on Oraserv, which will run Apache, PHP, and MySQL.

4.1.2. Production Environment

Macintosh Computer:
- Operating System: OSX Version 10.7.4
- Model: Mac
- Processor: Intel Core i5 @ 2.5Ghz
- Memory: 4GB 1333 MHz DDR3
- HD Size: 500GB

Windows Computer:
- Operating System: Windows Vista
- Model: Dell
- Processor: Intel Core 2 Duo E7500 @ 2.93GHz
- Memory: 4.00 GB
- HD Size: 297GB1

Windows Computer:
- Operating System: Windows 8
- Model: ASUS
- Processor: Intel Core i5 2.4Ghz
- Memory: 4GB 133MHz DDR3
- HD Size: 450GB

4.2. Deliverables

Deliverables are the products that are handed over to our client, Dr. Lim, and our professor, Dr. Lederman, upon completion of our project. Our deliverables will include all of the source code, documentation, database scripts, website code, deployment description in a README file, which will also contain any login credentials needed to interact with the system, the lyrics to our team song, and the video recording of our team song stored on a DVD.

4.3. Data Flow Diagrams

Our Data Flow Diagrams can be found in Appendix D at the end of this document. Within these diagrams one can see the movement of data between processes in the system and the external entities of the system.

4.4. Source Code

Below is a sample of the source code of Hobb-IT, including HTML, CSS, PHP, Java, and SQL.

HTML/CSS from Home Page:

```
<html>
<head>
        <style type="text/css">
    html{
                background-color:4169E1;
        }

    #buttons{
                text-align:center;
    }

    form{
                display: block;
                text-align:center;
    }

    #menu{
                text-align:center;
    }

    #userGreeting{
                text-align:center;
                display:block;
    }
    </style>
</head>
<body>
    <div id="userGreeting"> HOBB-IT </div>
    </br>
    </br>
    <form>
        Username:<input type="username" name="uname"></br>
        Password:<input type="edition" name="ed"></br>
        <div id="buttons">
                <input type="submit" value="ENTER"> </br>
                <input type="submit" value="FORGOT PASSWORD"></br>
                <input type="submit" value="LOGIN AS GUEST">
        <div>
    </form>
</body>
```

```
        </html>
```

PHP from Edit Account Page:

```
<?php

session_start(); // Start a new session

if($_SESSION['usertype'] == "Administrator"){

        require ('connect.php');

        $input = $_POST['input'];
        $username = $_POST['username'];
        $password = $_POST['password'];
        $column = $_POST['column'];

        $link = my_connect();
        $editAccount = $link->query("update AdvancedUser set $column = '".$input."' where
        UserName = '".$username."' and UserPassword = '".$password."'");

        $editAccount->close();
        $link->close();
        }
        else{
                session_destroy();
                header("Location: index.php");

        }
?>
```

Java Code for CardKingdom.com:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

import java.io.IOException;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;

/**
 *
 * @author karlappel
 */
public class CardKingdomScraper {

    public static void main(String args[]) throws IOException {

//        String arg1[] = {"searchResults" , "hellfire", "" , "" , ""};
//        args = arg1;
        String type = args[0];
        String searchResultValue = "";
        ArrayList<String> priceResultValue = new ArrayList<String>();
        CardKingdomScraper k = new CardKingdomScraper();

        if (type.equals("searchResults")) {
            try {
                searchResultValue = k.searchResults(args);
            } catch (Exception e) {
                Formatter f = new Formatter();
                ArrayList<MagicCard> m = new ArrayList<MagicCard>();
                m.add(new MagicCard());
                searchResultValue = f.format(m);
            }
        } else if (type.equals("priceResults")) {
            //String url = "http://www.cardkingdom.com/catalog/item/186773";
            try {
                String url = args[1];
                priceResultValue = k.priceResults(url);
            } catch (Exception e) {
                System.out.println("null");
                System.out.println("|");
                System.out.println("null");
            }
        }

    }

    private ArrayList<String> priceResults(String url) throws IOException {

        ArrayList<String> priceResults = new ArrayList<String>();
        Document doc = Jsoup.connect(url).get();
        Element priceTable = doc.select("table.grid.compact").first();
        Element bestCondition = priceTable.select("tr").get(1);
        Element price = bestCondition.select("td").get(1);
        Element stock = bestCondition.select("td").get(2);
        String stockString = stock.text().trim();
        String priceString = price.text().trim();

        try {
            Integer.parseInt(stockString);
        } catch (NumberFormatException e) {
            stockString = "0";
        }

        System.out.println(priceString);
        System.out.println("|");
        System.out.println(stockString);
        priceResults.add(priceString);
        priceResults.add(stockString);
        return priceResults;
    }

    public String searchResults(String args[]) throws IOException {

        String url =
"http://www.cardkingdom.com/catalog/view?search=basic&filter%5Bname%5D=";

        url += args[1];

        for (int j = 2; j < args.length; j++) {
            url += "+" + args[j];
        }

        ArrayList<MagicCard> magicCards = new ArrayList<MagicCard>();
        //System.out.println(url);

        Document doc = Jsoup.connect(url).get();

        Elements tables = doc.select(".grid");
        if (tables == null) {
            System.out.println("No valid output found");
        } else if (tables.size() <= 2) {
            System.out.println("No valid output found");
        }

        for (int i = 0; i < tables.size(); i++) {

            Elements rows = tables.get(i).select("tr");
            //System.out.println("table size " + tables.size());
            //System.out.println("row sizes " + rows.size());

            for (int j = 1; j < rows.size(); j++) {

                Element row = rows.get(j);
                Elements columns = row.select("td");
                if (columns.size() == 5) {
                    continue;
                }

                Element name = columns.get(0);
                // System.out.println("here " + j);
                Element edition = columns.get(1);
                // System.out.println("here " + j);
                Element condition = columns.get(6);
                //  System.out.println("here " + j);
                Element stock = columns.get(7);
                //  System.out.println("here " + j);
                Element price = columns.get(8);
                //  System.out.println("here " + j);
                String cardURL = name.select("a").first().attr("href");

//                  System.out.println("Name " + name.text());
//                  System.out.println("edition " + edition.text());
//                  System.out.println("condition " + condition.text());
//                  System.out.println("stock " + stock.text());
//                  System.out.println("price " + price.text());
//                  System.out.println("Card URL " + cardURL);
                boolean foil = edition.text().contains("Foil");
//                  System.out.println("foil " + foil);
//              public MagicCard(String name, String edition, String price, boolean foil,
//          int quantity, String condition , String URL)
            MagicCard m = new MagicCard(name.text(), edition.text(), price.text(),
                    foil, stock.text(), condition.text(),
                    cardURL);
            magicCards.add(m);
            //System.out.println("Iteration " + j);

          }

        }

        Formatter f = new Formatter();
        String tableHTML = f.format(magicCards);
        System.out.println(tableHTML);
        return tableHTML;
    }

}
```

SQL Create Table Statements for all tables:

Administrator Table

```
delimiter $$

CREATE TABLE `Administrator` (
  `AdminUserName` varchar(45) NOT NULL,
  `UserPassword` varchar(45) default NULL,
  `First Name` varchar(45) default NULL,
  `Last Name` varchar(45) default NULL,
  PRIMARY KEY  (`AdminUserName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1$$
```

Advanced User Table

```
delimiter $$

CREATE TABLE `AdvancedUser` (
  `UserName` varchar(45) NOT NULL,
  `UserPassword` varchar(45) default NULL,
  `FirstName` varchar(45) default NULL,
  `LastName` varchar(45) default NULL,
  PRIMARY KEY  (`UserName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1$$
```

Editions Table

```
delimiter $$

CREATE TABLE `Editions` (
  `EditionName` varchar(100) NOT NULL,
  `BlockName` varchar(45) NOT NULL,
  PRIMARY KEY  (`EditionName`),
  UNIQUE KEY `EditionName_UNIQUE` (`EditionName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1$$
```

Magic Card Table

```
delimiter $$

CREATE TABLE `MagicCard` (
  `ID` int(11) NOT NULL auto_increment,
  `CardName` varchar(45) default NULL,
  `CardEdition` varchar(45) default NULL,
  `Foil` tinyint(1) default NULL,
  `DefaultQuality` varchar(45) default NULL,
  PRIMARY KEY  (`ID`),
  UNIQUE KEY `CardName_UNIQUE` (`CardName`,`CardEdition`,`Foil`)
) ENGINE=InnoDB AUTO_INCREMENT=50 DEFAULT CHARSET=latin1$$
```

OnList Table

```
delimiter $$

CREATE TABLE `OnList` (
  `UserName` varchar(45) NOT NULL,
  `ID` int(11) NOT NULL,
  `WishList` tinyint(1) default NULL,
  `Tracked` tinyint(1) default NULL,
  `Bought` tinyint(1) default NULL,
  `Past` tinyint(1) default NULL,
  PRIMARY KEY  (`UserName`,`ID`),
  KEY `ID_idx` (`ID`),
  CONSTRAINT `MagicCardKey` FOREIGN KEY (`ID`) REFERENCES `MagicCard` (`ID`)
ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `UserName` FOREIGN KEY (`UserName`) REFERENCES `AdvancedUser`
(`UserName`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1$$
```

Search Table

```
delimiter $$

CREATE TABLE `Search` (
  `WebsiteName` varchar(20) NOT NULL,
  `ID` int(11) NOT NULL,
  `DateTimeScraped` datetime NOT NULL,
  `Quantity` int(11) default NULL,
  `Quality` varchar(45) default NULL,
  `Price` decimal(6,2) default NULL,
  `UserName` varchar(45) NOT NULL,
  `URL` varchar(250) default NULL,
  PRIMARY KEY  (`WebsiteName`,`ID`,`DateTimeScraped`),
  KEY `ID_idx` (`ID`),
  KEY `UserName_idx` (`UserName`),
  CONSTRAINT `UserName1` FOREIGN KEY (`UserName`) REFERENCES `AdvancedUser`
(`UserName`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `WebsiteName` FOREIGN KEY (`WebsiteName`) REFERENCES `Website`
(`WebsiteName`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1$$
```

Template Table

```
delimiter $$

CREATE TABLE `Template` (
  `TemplateID` int(11) NOT NULL,
  `Name` varchar(1024) default NULL,
  `ReloadState` tinyint(1) default NULL,
  `PreviousTemplateID` int(11) default NULL,
  PRIMARY KEY  (`TemplateID`),
  UNIQUE KEY `PreviousTemplateID_UNIQUE` (`PreviousTemplateID`),
  CONSTRAINT `PreviousTemplateID` FOREIGN KEY (`PreviousTemplateID`)
REFERENCES `Template` (`TemplateID`) ON DELETE NO ACTION ON UPDATE NO
ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1$$
```

Website Table

```
delimiter $$

CREATE TABLE `Website` (
  `WebsiteName` varchar(20) NOT NULL,
  `Websitecol` varchar(45) default NULL,
  `GuestUserDefault` tinyint(4) default NULL,
  `CurrentTemplateID` int(11) NOT NULL,
  `TemplateID` int(11) default NULL,
  PRIMARY KEY  (`WebsiteName`),
  UNIQUE KEY `TemplateID_UNIQUE` (`CurrentTemplateID`),
  CONSTRAINT `CurrentTemplateID` FOREIGN KEY (`CurrentTemplateID`) REFERENCES
`Template` (`TemplateID`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1$$
```

5. Test Requirements & Results

5.1. Explanation of Test Plan/Strategy

Hobb-IT will be tested to ensure that all the functional requirements have been met. Hobb-IT will be compatible with current versions of Chrome, Firefox, Internet Explorer, Safari, and Mobile Safari. Testing will be limited to these browsers as requested by the client. Each functional requirement will be split into an individual unit and will be tested for success. Once this process is complete, all functional requirements will be tested together.

In addition, Hobb-IT will also test all non-functional requirements needed for Hobb-IT. Hobb-IT will have website templates that are easy to maintain. Hobb-IT will also be efficient, stable, user friendly, and will follow the legal processes on all websites tracked.

5.1.1. Unit Tests

Each unit test case in the system will be tested separately. Once each test passes their respectively requirements the system will be tested as a whole. Pleased see the excel spread sheet for the parse information, request real time search, and login unit tests.

5.1.2. Test Cases

A test case is a set of conditions or variables under which a tester will determine whether a system under test satisfies requirements or works correctly. In the case of Hobb-IT, Team Illumination will outline Unit Tests for the system to determine whether the system meets the requirements of our client, Dr. Darren Lim, and the criteria of good software engineering practices.

5.1.3. Approach to Testing

Illumination Technologies will test Hobb-IT using the Unit Tests in our Detailed Design document. Once each unit is completed and has passed every step, the units will be integrated together and tested as a whole. This way, all bugs will be found and corrected in the individual units before the whole system is tested. Testing will occur on a regular basis throughout development in order to assure that Illumination Technologies is developing working software. The development and testing process, along with any bugs that are found, will be discussed on a regular basis with our client, Dr. Darren Lim.

5.2. Test Results

General

- **PASSED**/FAILED Hobb-IT will be compatible with current versions of Chrome, Firefox, Internet Explorer, and Safari.

Administrator

- **PASSED**/FAILED Will be able to access all stored data on the database.
- PASSED/**FAILED** Will be able to access the search history of all users.
  - This functionality was removed in the development process.
- **PASSED**/FAILED Will be able to approve usernames and passwords of new Advanced User accounts.
- **PASSED**/FAILED Will be able to change how a website's data is parsed into the database.
- **PASSED**/FAILED Will be able to view and change login credentials of all users.
- PASSED/**FAILED** Will be able to clear the database history of past searches.
  - This functionality was removed in the development process.

Advanced User

- **PASSED**/FAILED Will be able to login to Hobb-IT using a username and password approved by the Administrator.
- **PASSED**/FAILED Will be able to search for the real-time price of a *Magic: The Gathering* card from any tracked website.
- **PASSED**/FAILED Will be able to save a list of tracked *Magic: The Gathering* cards.
- **PASSED**/FAILED Will be able to save a list of purchased *Magic: The Gathering* cards.
- **PASSED**/FAILED Will be able to save a list of *Magic: The Gathering* cards the user wishes to track in the future.
- **PASSED**/FAILED Will be able to edit the *Magic: The Gathering* cards that appear on any list.
- **PASSED**/FAILED Will be able to access the search history associated with the Advanced User's account.
- PASSED/**FAILED** Will be able to view a visual representation of the fluctuations in prices of a tracked *Magic: The Gathering* card.
  - This functionality was removed in the development process.

Guest User

- **PASSED**/FAILED Will be able to access Hobb-IT without login credentials.
- **PASSED**/FAILED Will be able to view the real-time price and condition of a *Magic: The Gathering* card from one website at a time.

Non Functional Requirements Inventory

- **PASSED**/FAILED Hobb-IT will be easy to maintain.
- **PASSED**/FAILED Hobb-IT will be efficient.
- **PASSED**/FAILED Hobb-IT will be stable.
- **PASSED**/FAILED Hobb-IT will be user friendly.
- **PASSED**/FAILED Hobb-IT will follow the legal processes on all websites tracked.

6. Appendixes

A. Glossary of Terms

Administrator: A singular user of Hobb-IT defined in the User Case Narrative

Advanced User: A user of Hobb-IT defined in the User Case Narrative

Boolean: A true or false value for a data type

Chrome: A web browser created by Google Inc.

DFD: **D**ata **F**low **D**iagram; A representation of how data will move and interact throughout a system

ER Diagram: **E**ntity **R**elation diagram; A data model that shows the relationships between different database tables

Firefox: A web browser created by Mozilla Foundation

Foreign Key: A field in one database table that uniquely identifies a row in another database table

Guest User: A user of Hobb-IT defined in the User Case Narrative

Hobb-IT: **Hobb**y **I**nformation **T**racker; The name of the project

HTML: **H**yper**T**ext **M**arkup **L**anguage; A markup language used to structure website pages

HTTP: **H**yper**T**ext **T**ransfer **P**rotocol; A protocol used to move hypertext request and information between browsers and servers

Int: A data type represented by a numerical value with no decimal places

Internet Explorer: A web browser created by Microsoft Inc.

Java: An object oriented programming language owned and developed by Oracle Corporation

JDBC: **J**ava **D**ata**B**ase **C**onnectivity; A standard API used for connecting to a database via Java

Jsoup: A Java library written to work with real word html applications such as extracting and manipulating data from web browsers.

Magic Card: A paper card from the game Magic: The Gathering

MTG: **M**agic: **T**he **G**athering; a game created by Richard Garfield published by Wizards of the Coast

Mobile Safari: A web browser used on Apple's iOS mobile devices (ex. iPhone, iPad , iPod Touch)

PHP: **P**HP **H**ypertext **P**reprocessor; A recursive acronym, it is a server side language generally used to generate HTML and CSS code.

Primary Key: A row or rows in a database table that uniquely identify that database table

Safari: A web browser created by Apple Inc.

SCP: **S**ecure **C**opy **P**rotocol; A protocol used for data communication

Test Case: An individual part of a Unit Test that focuses on one module of the software

Unit Test: A form of software testing in which individual parts of the software are tested for their functionality

varChar: A data type represented by an undefined amount of characters, such as letters, numbers, and special symbols

Website Map: a hierarchy chart that shows the different pages within Hobb-IT that users can access

Web Parsing: The process of breaking the information on a website page into a more usable format

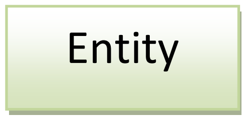
Web Scraping: The process of retrieving information on a website page

B. Data Dictionary

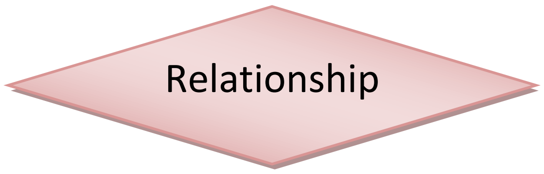
Please see the attached excel file for the complete Data Dictionary.

C. ER Diagram

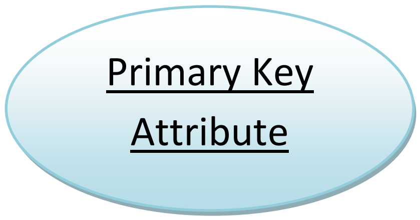
Legend



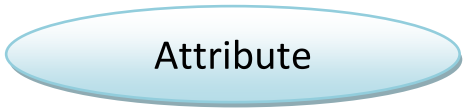
Entity: An entity is something within the database that can exist on its own and that can be uniquely identified.



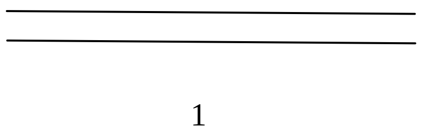
Relationship: A relationship is something that connects two entities and describes how they are related.



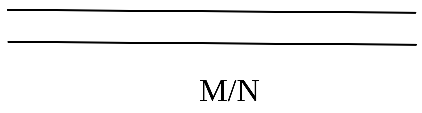
Primary Key Attribute: This is an attribute of an entity that uniquely identifies the entity along with any other primary key attributes.



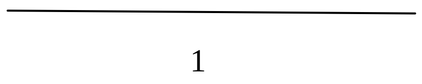
Attribute: Any data item that is associated with the connected entity.



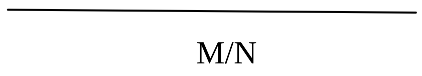
Connections-
Double Line 1: Exactly 1 in the relationship



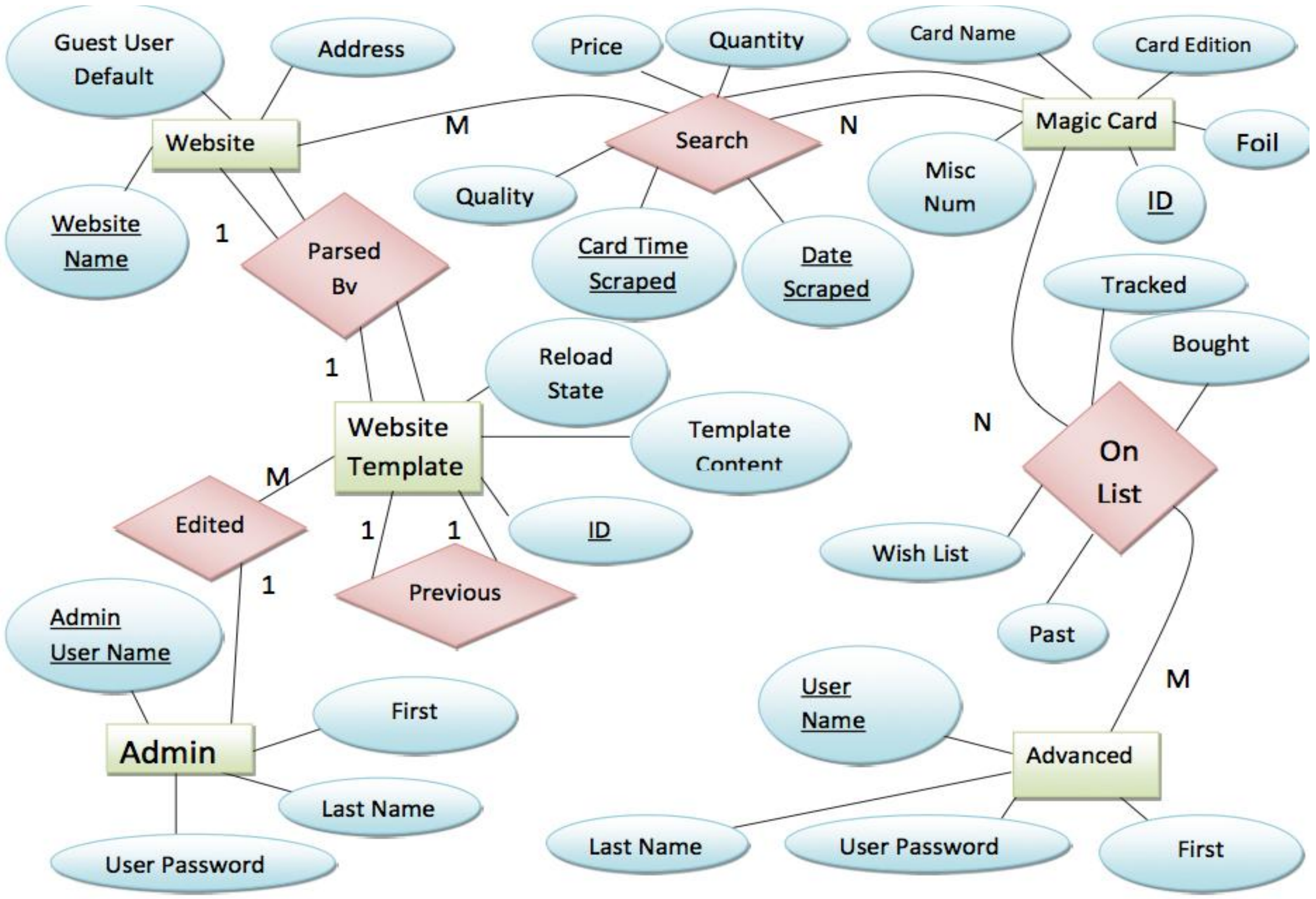
Double Line M/N: At least 1 in the relationship



Single Line 1: 0 or 1 in the database



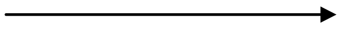
Single Line M/N: Any number


Guest User Default
Address
Price
Quantity
Card Name
Card Edition
Website
M
Search
N
Magic Card
Foil
Misc Num
ID
Quality
Website Name
1
Parsed By
Card Time Scraped
Date Scraped
Tracked
Bought
1
Reload State
N
Website Template
Template Content
On List
M
Edited
1
1
ID
Wish List
1
Previous
Past
Admin User Name
M
User Name
First
Admin
Advanced
Last Name
User Password
Last Name
User Password
First

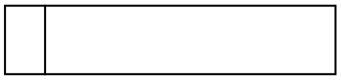
D. Data Flow Diagrams

Diagram Legend

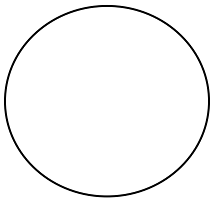
External Entity: Represents outside sources of data to and from the system



Data Flow: Represents the movement of data



Data Store: Represents data that is not moving or at rest

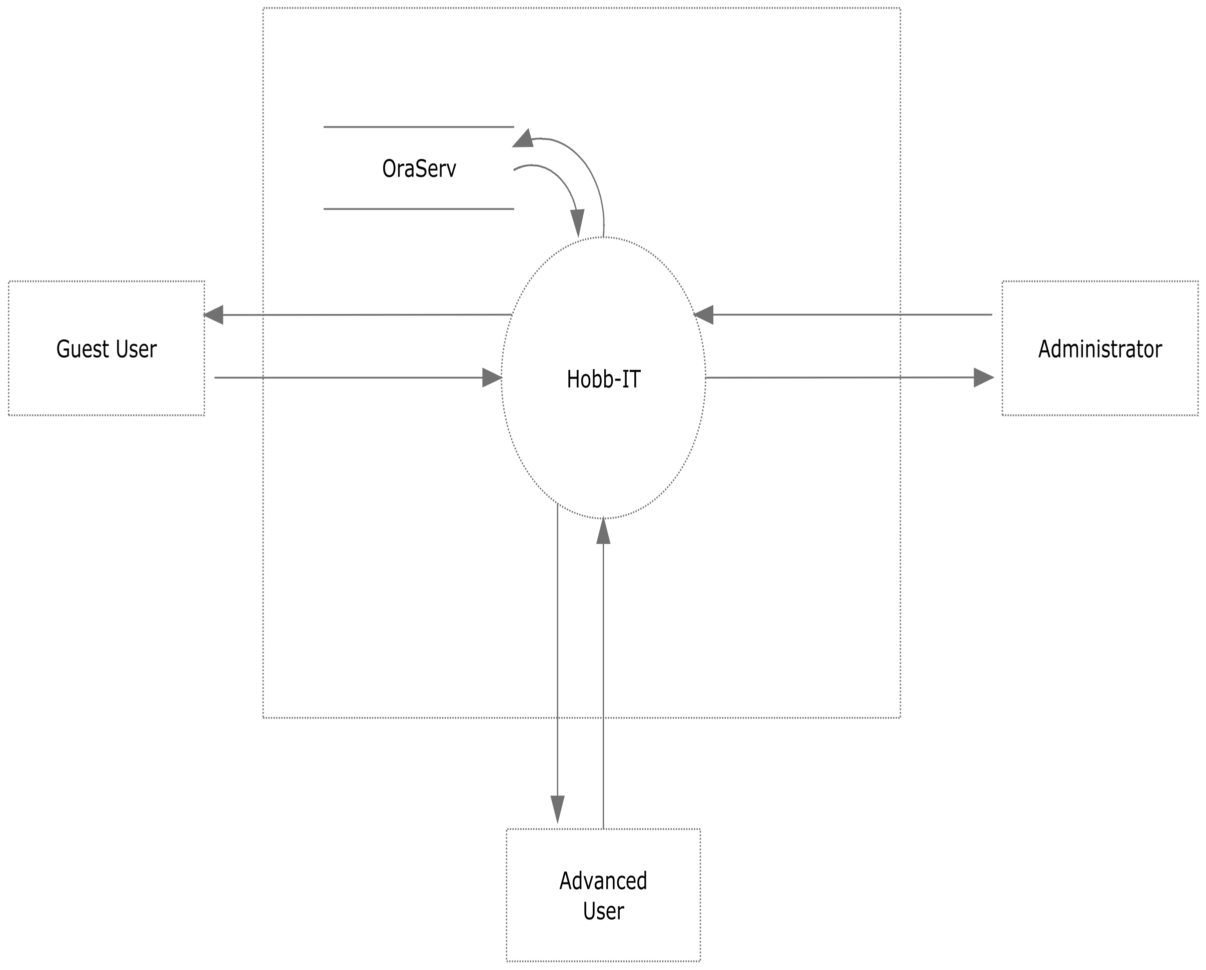


Process: Represents an activity that manipulates the data

Context Diagram

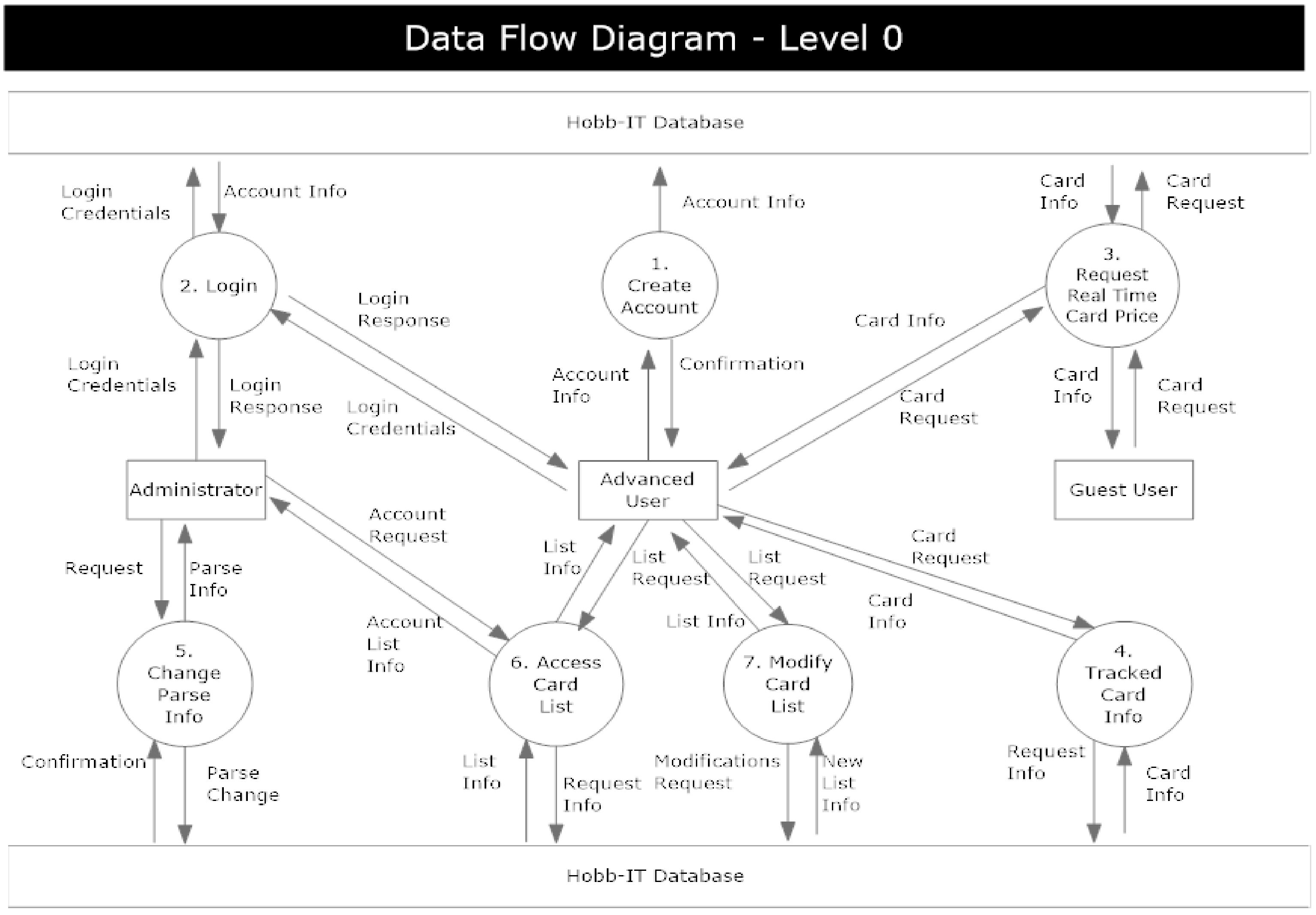
The Context Diagram is a diagram that shows the system described as a single entity and the boundary of the system. In this diagram the interaction of the system between external or internal sources of information are defined as separate entities.

Data Flow Diagram - Context Diagram



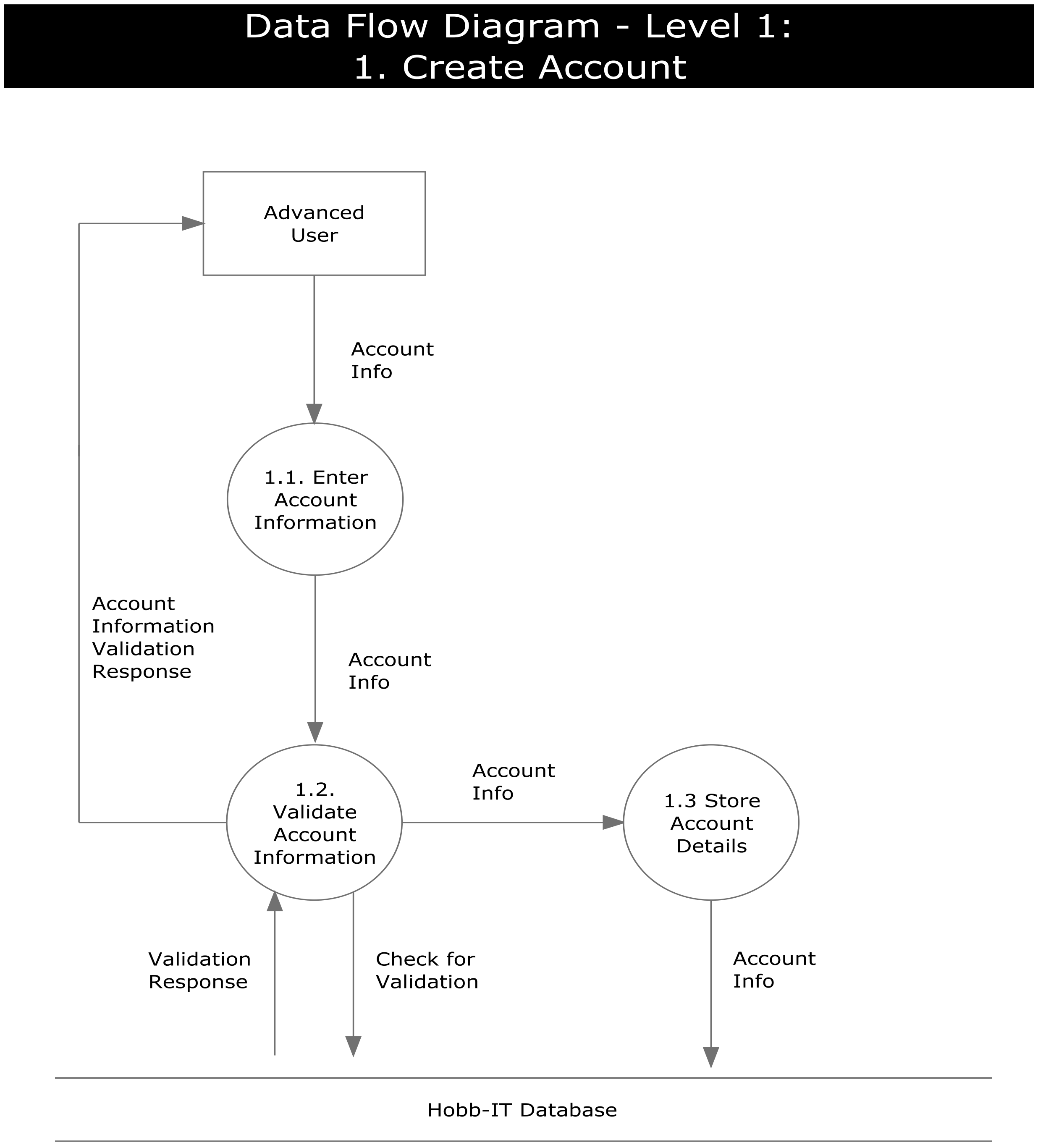
Level 0 Diagram

The Level 0 diagram shows a simplified overview of the major players and processes that occur for the system and the interaction of these processes among each other. This is an expansion upon the Context Diagram.

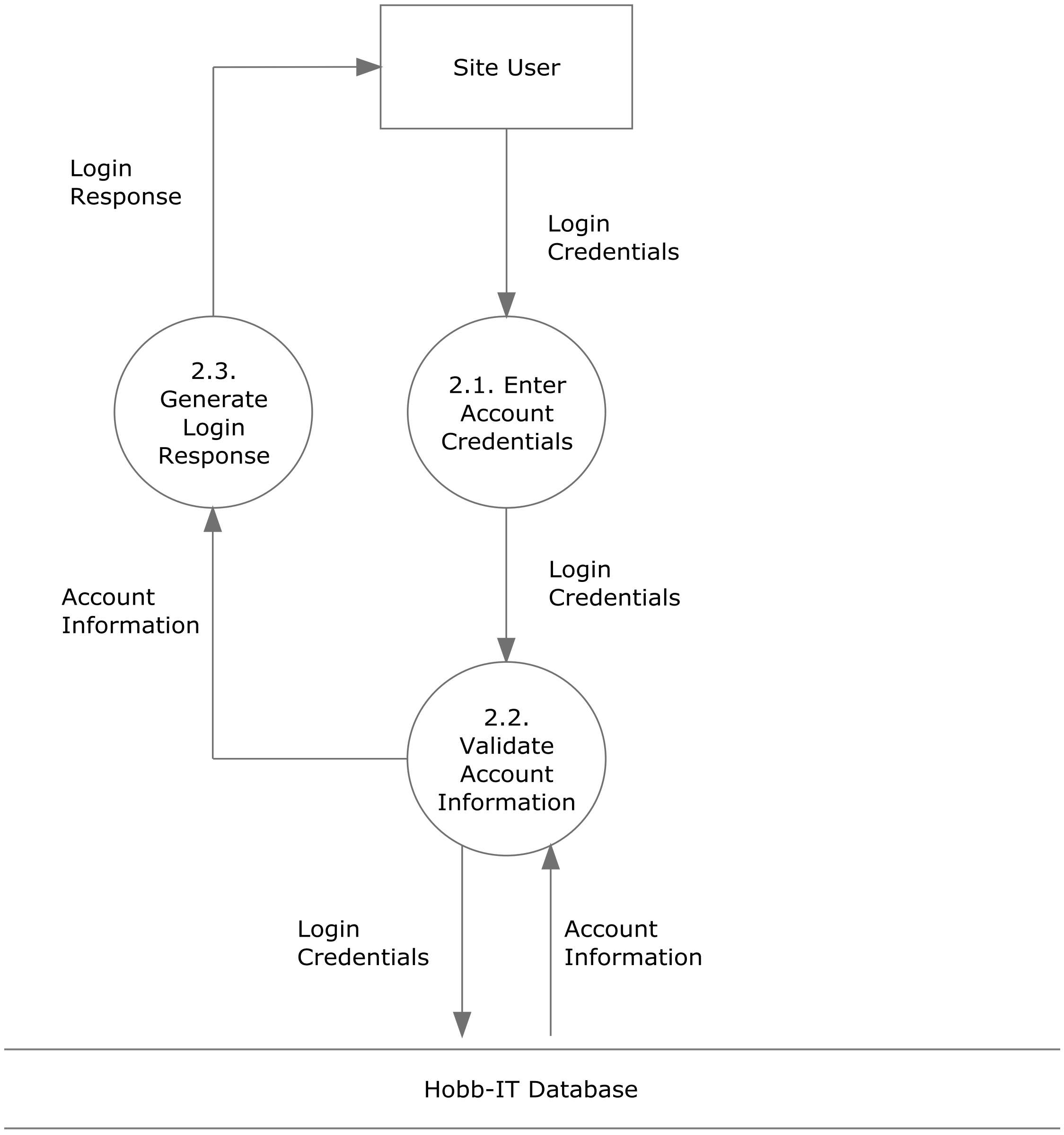


Level 1 Diagrams

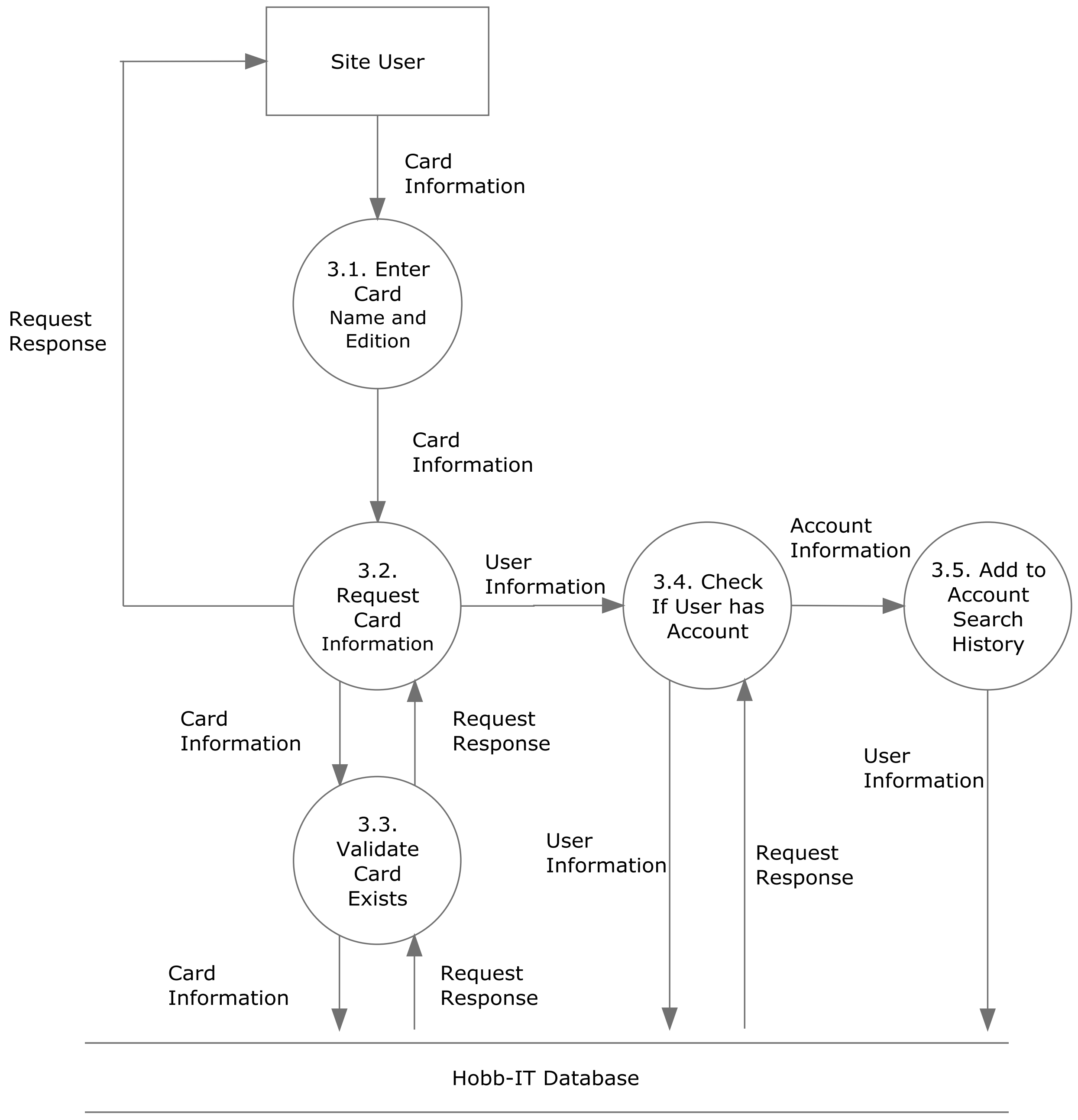
The level 1 diagrams expands upon a particular process within a level 0 diagram and shows the major players and processes associated with this break down of this level 0 process.


Data Flow Diagram - Level 1:
1. Create Account
Advanced User
Account Info
1.1. Enter Account Information
Account Information Validation Response
Account Info
Account Info
1.2. Validate Account Information
1.3 Store Account Details
Validation Response
Check for Validation
Account Info
Hobb-IT Database

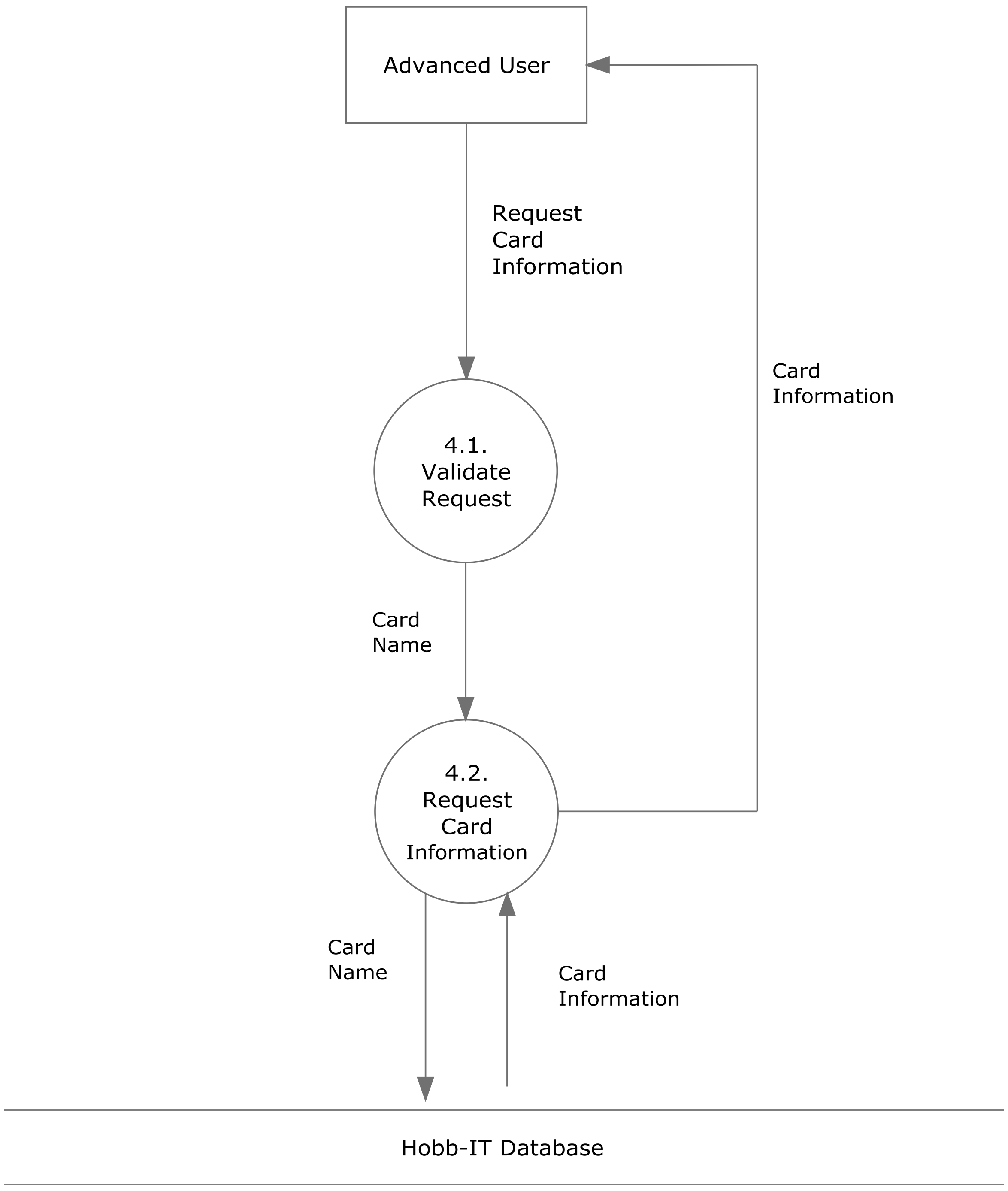
Data Flow Diagram - Level 1:
2. Login



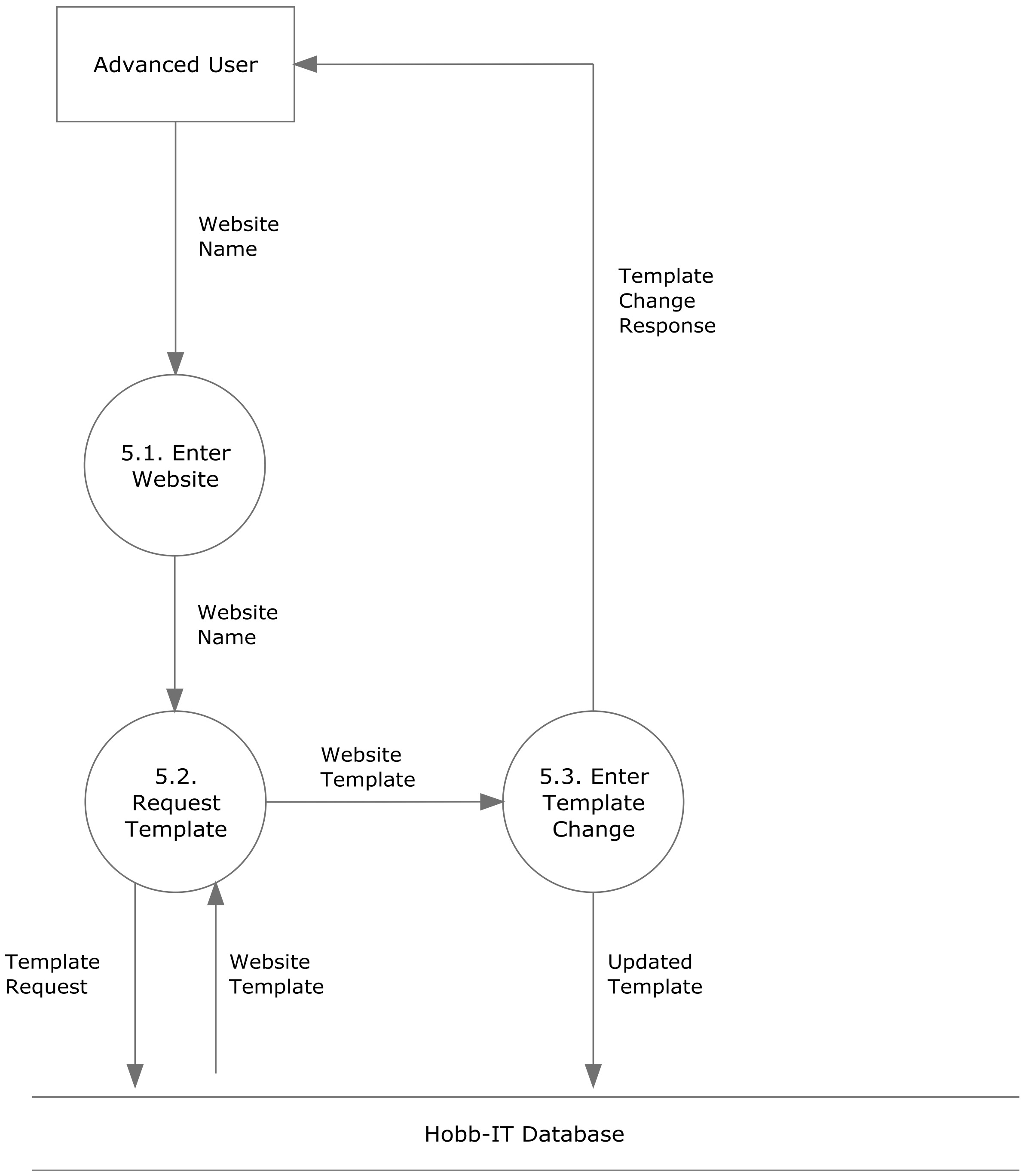
Data Flow Diagram - Level 1:
3. Request Real Time Card Price



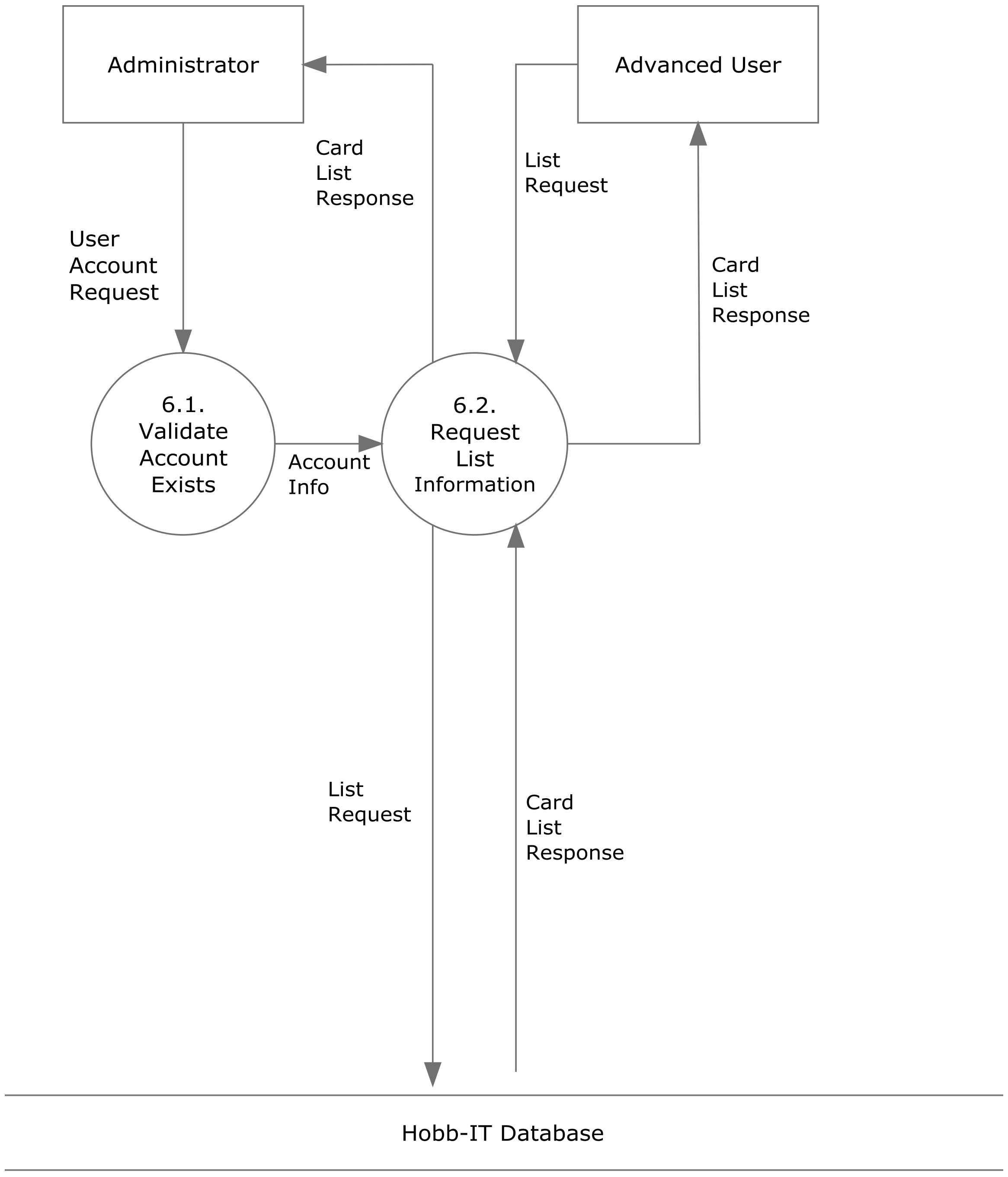
Data Flow Diagram - Level 1: 4. Tracked Card Information



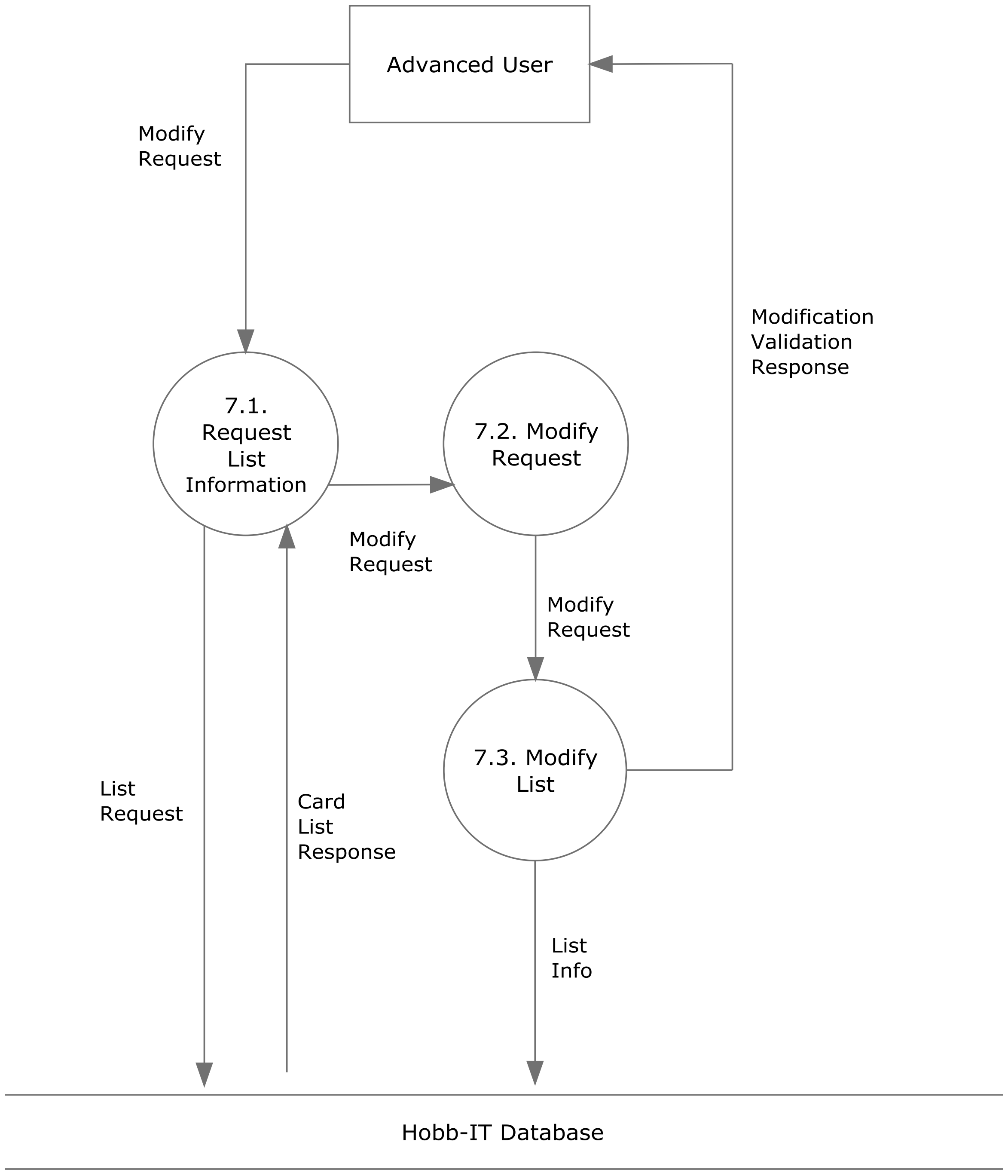
Data Flow Diagram - Level 1: 5. Change Parse Information



Data Flow Diagram - Level 1: 6. Access Card List

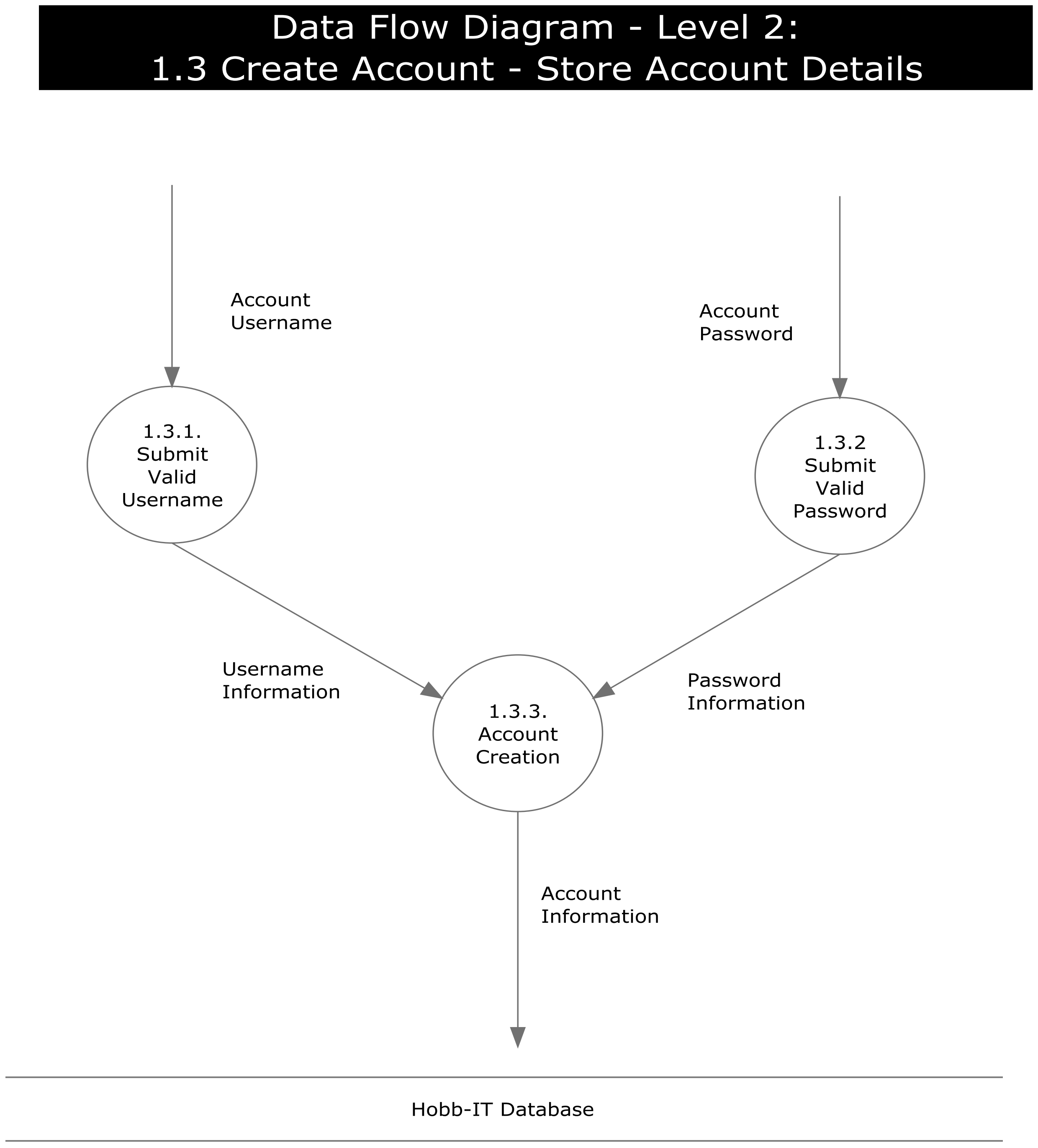


Data Flow Diagram - Level 1: 7. Modify Card List

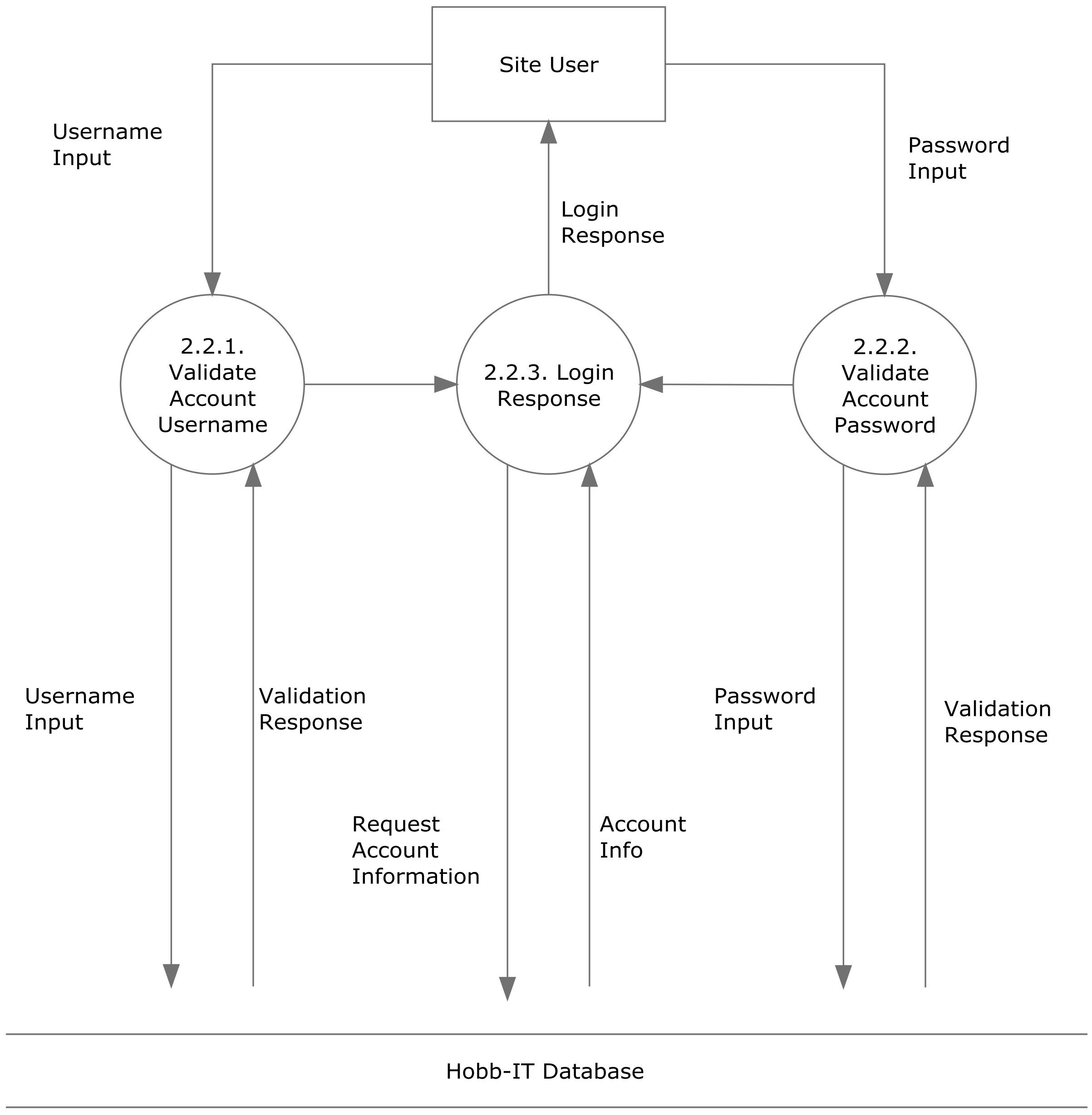


Level 2 Diagrams

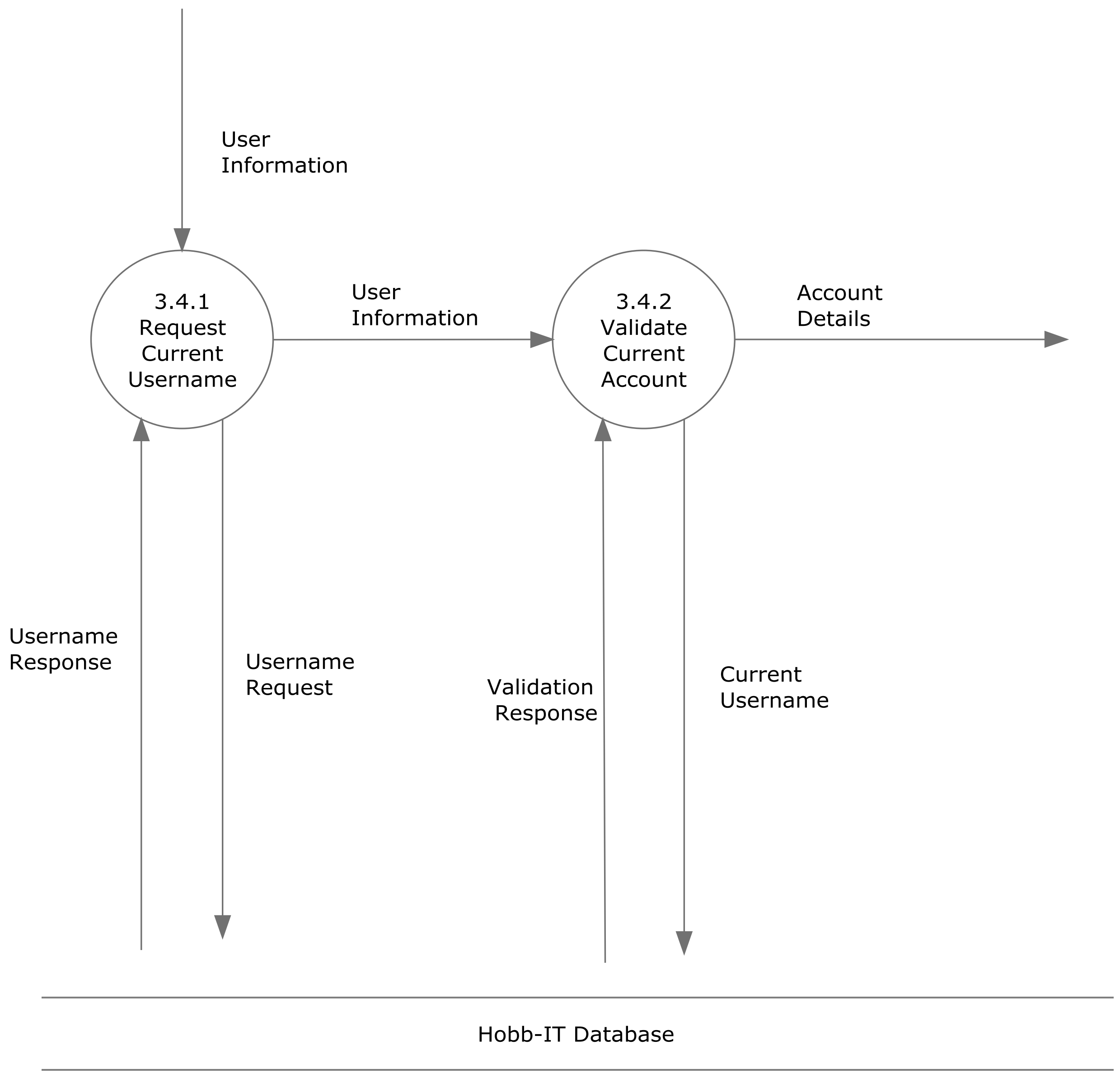
The level 2 diagrams expands upon a particular process within a level 1 diagram and shows the major players and processes associated with the breakdown of this level 1 process.



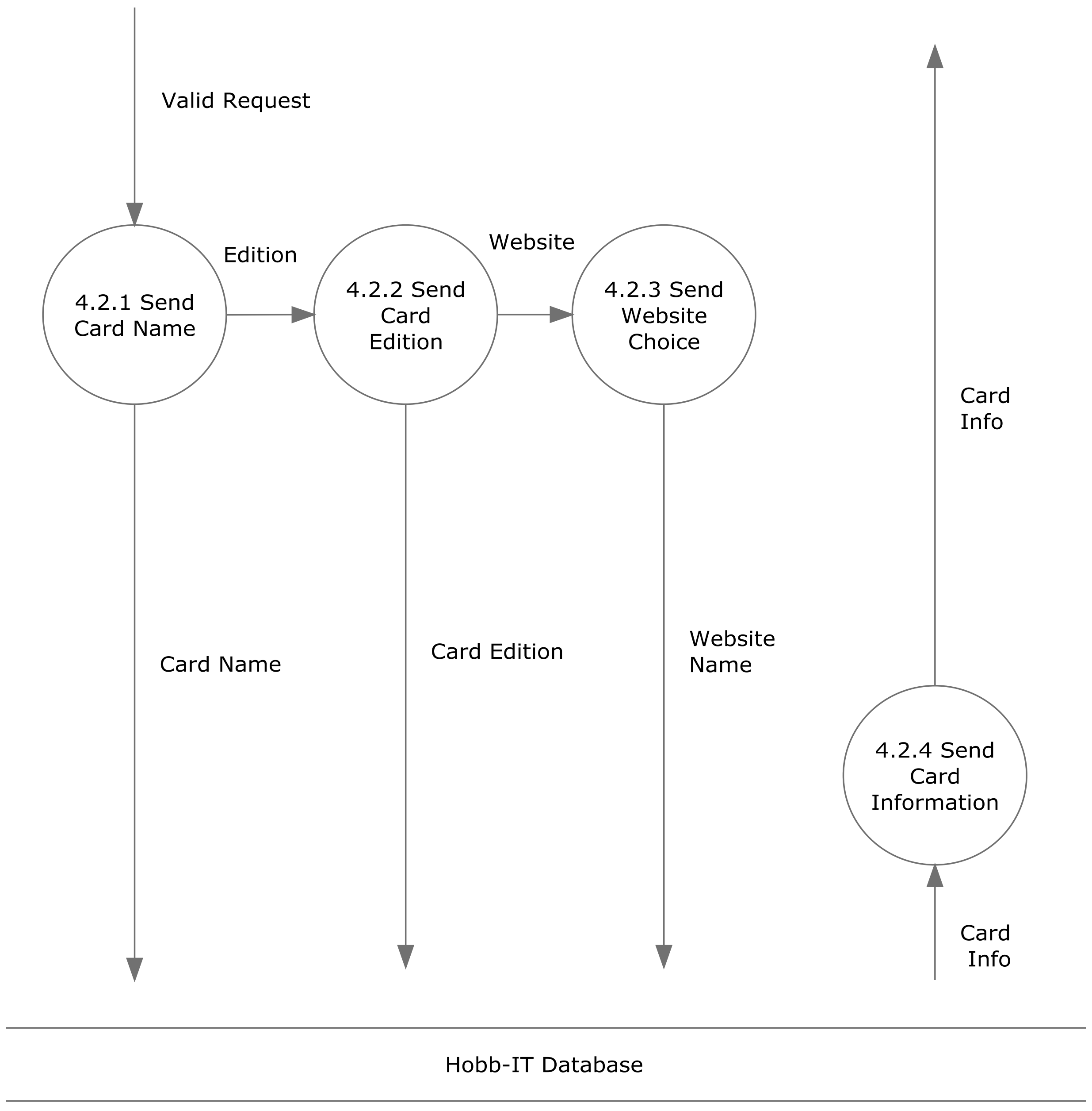
Data Flow Diagram - Level 2: 2.2 Login - Validate Account Information



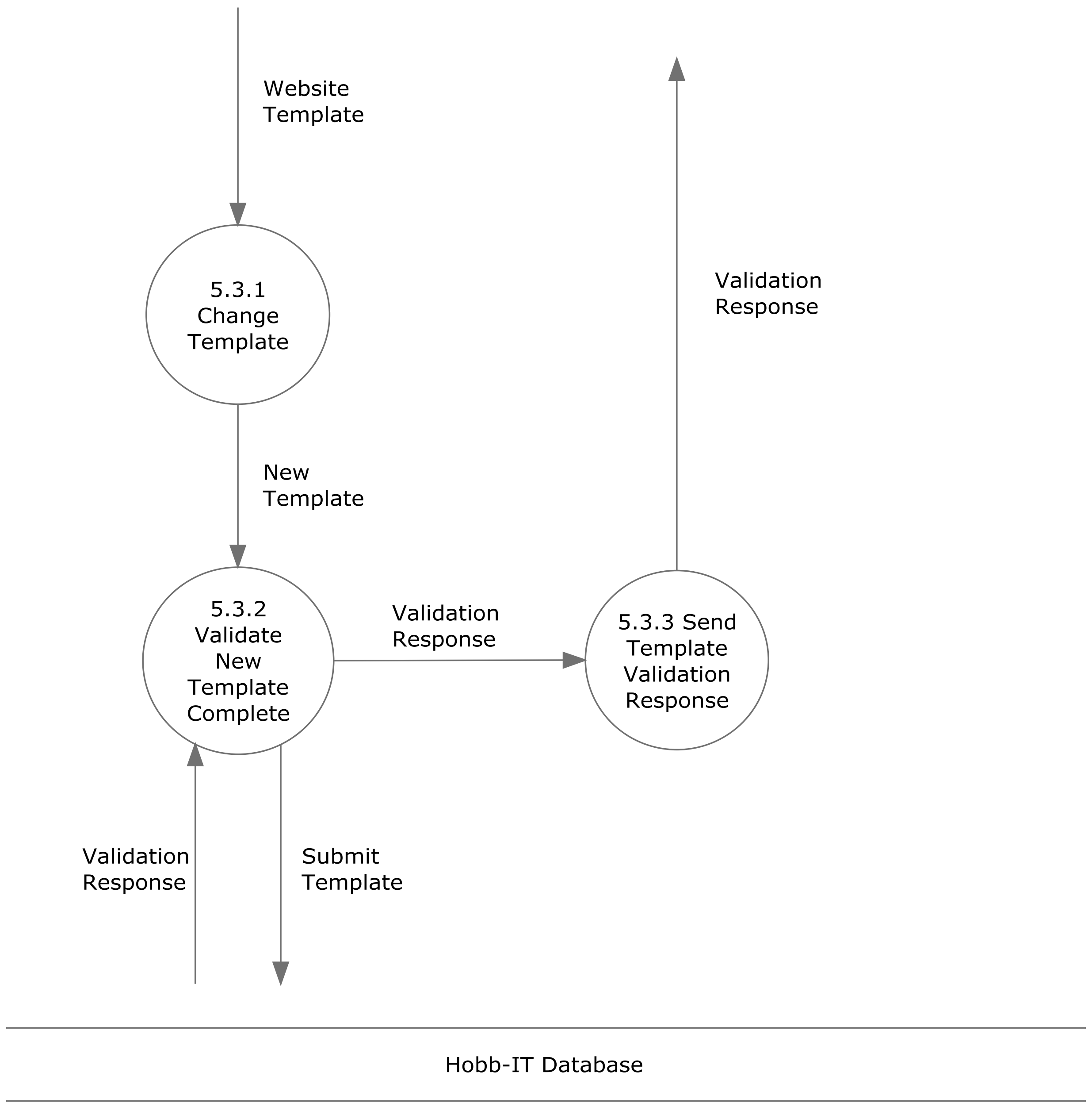
Data Flow Diagram - Level 2: 3.4 Request Real Time Card Price Check If User Has Account



Data Flow Diagram - Level 2: 4.2 Tracked Card Information Request Card Information



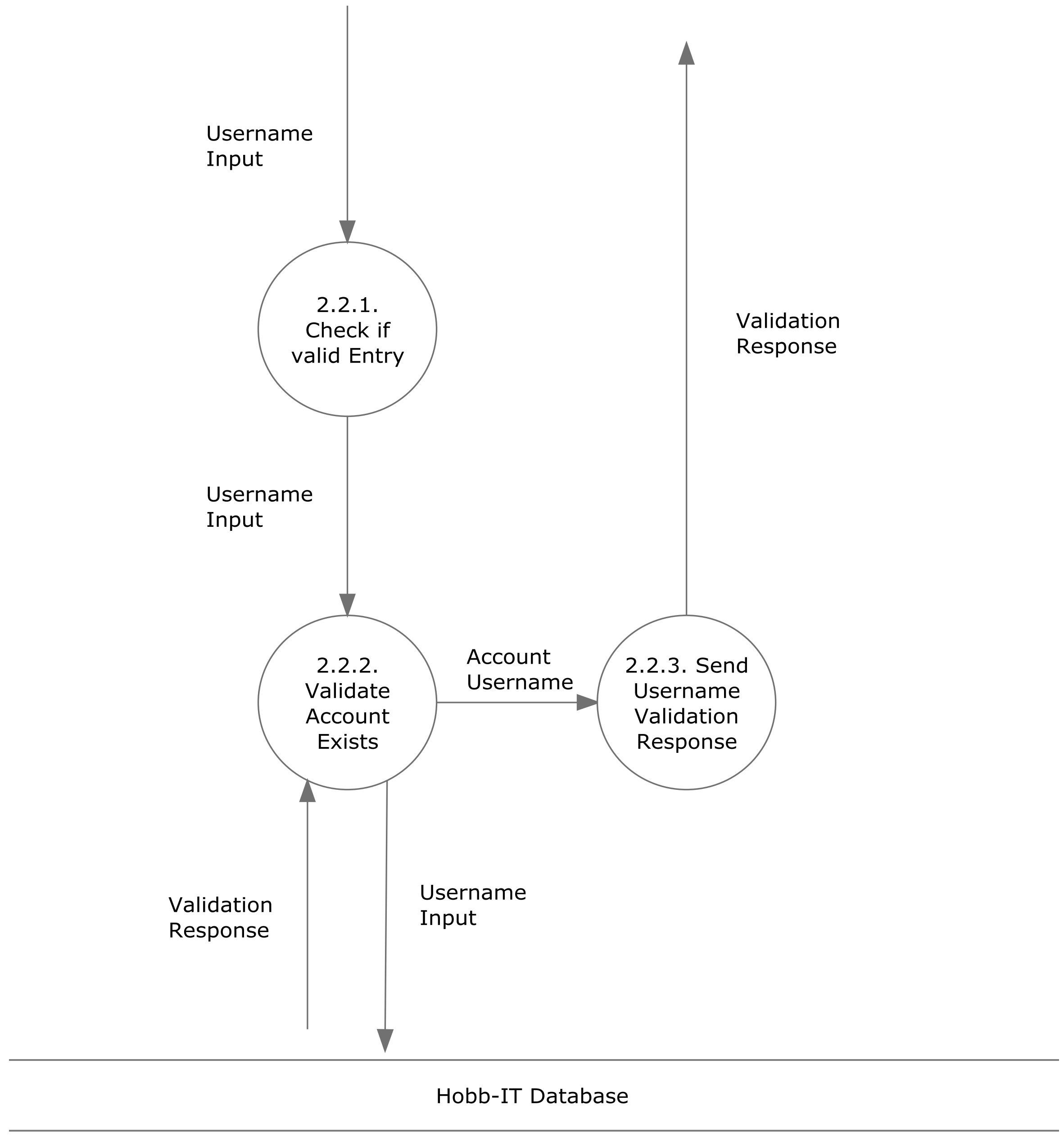
Data Flow Diagram - Level 2: 5.3 Change Parse Information Enter Template Change



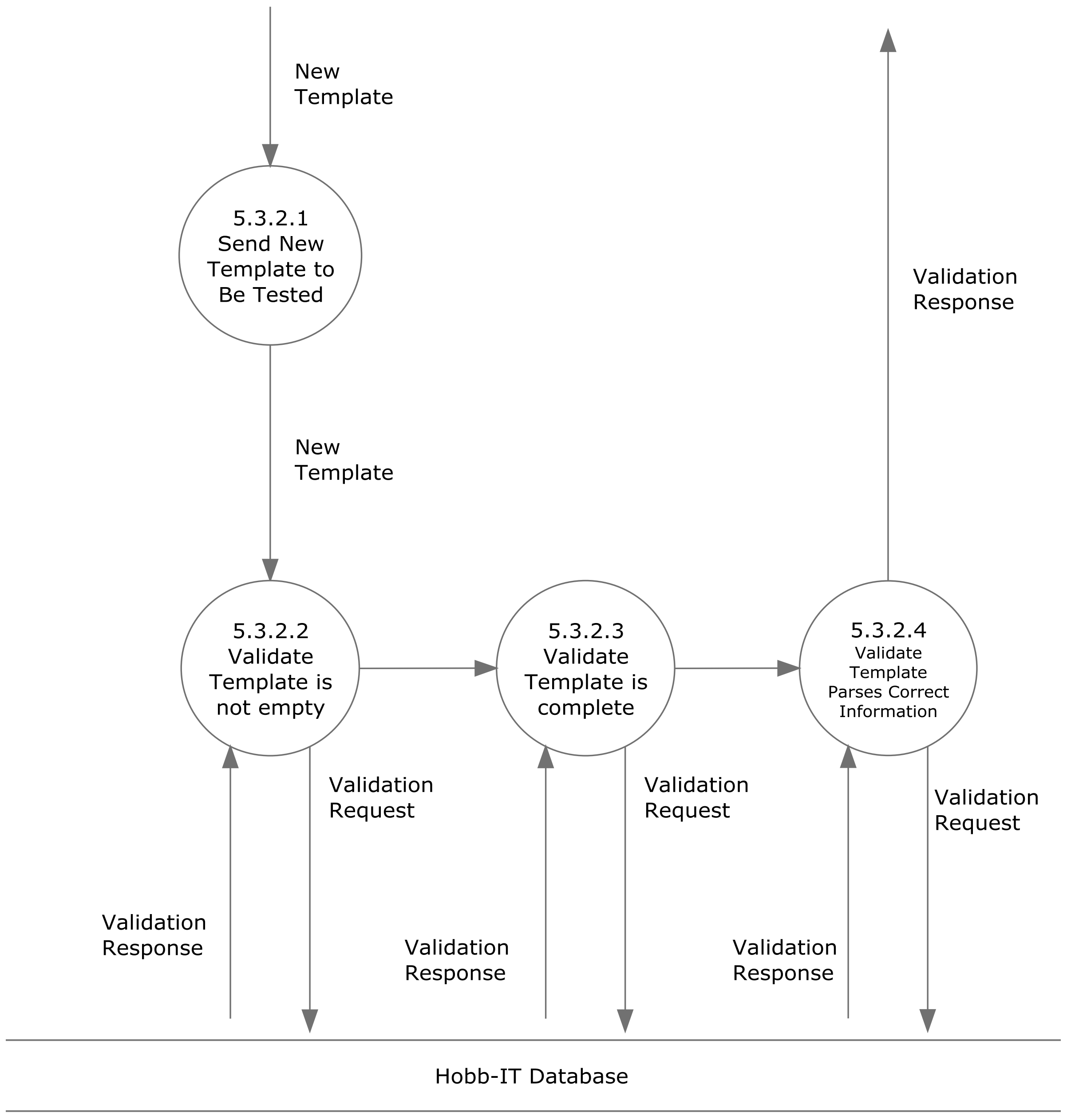
Level 3 Diagrams

The level 3 diagrams expand upon a particular process within a level 2 diagram and shows the major players and processes associated with the breakdown of this level 2 process.

Data Flow Diagram - Level 3:
2.2.1 Login - Validate Account Information
Validate Account Username



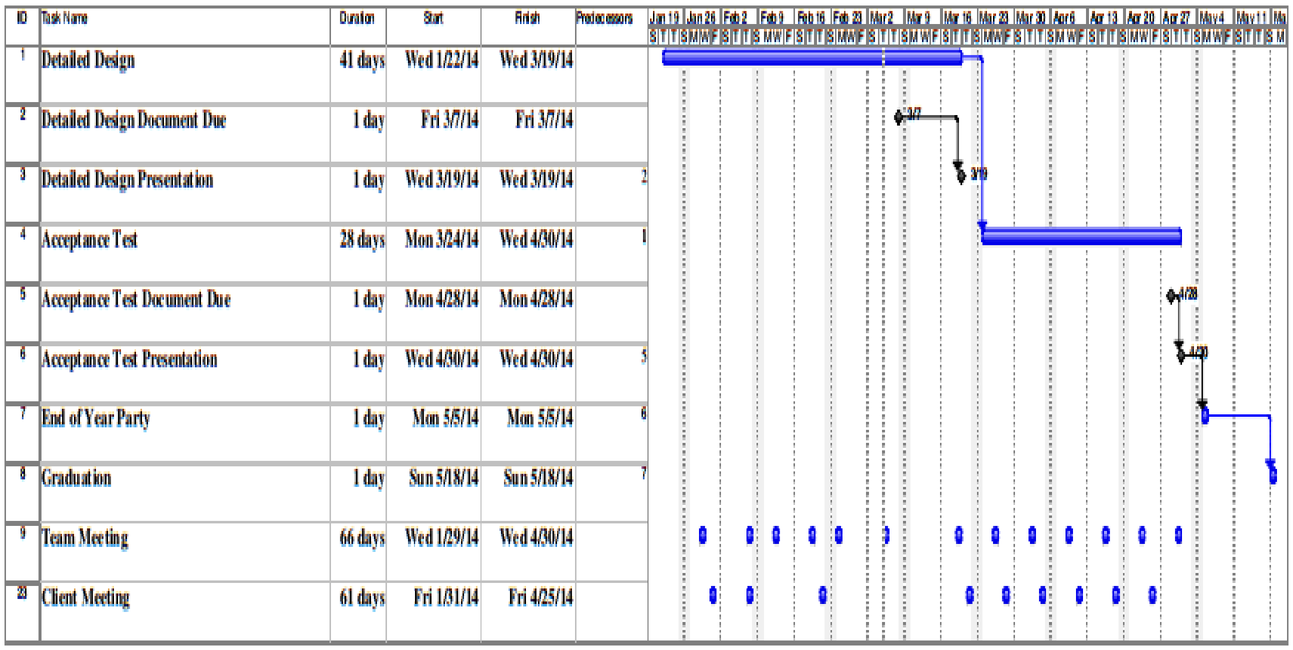
Data Flow Diagram - Level 3:
5.3.2 Change Parse Information
Enter Template Change
Validate New Template Change



E. Test Results

The test results for Hobb-IT are in the attached Excel spreadsheet. The description of the Test Plan can be viewed above in section 5.

F. Timeline



ID	Task Name	Duration	Start	Finish	Predecessors
1	Detailed Design	41 days	Wed 1/22/14	Wed 3/19/14	
2	Detailed Design Document Due	1 day	Fri 3/7/14	Fri 3/7/14	
3	Detailed Design Presentation	1 day	Wed 3/19/14	Wed 3/19/14	2
4	Acceptance Test	28 days	Mon 3/24/14	Wed 4/30/14	1
5	Acceptance Test Document Due	1 day	Mon 4/28/14	Mon 4/28/14	
6	Acceptance Test Presentation	1 day	Wed 4/30/14	Wed 4/30/14	5
7	End of Year Party	1 day	Mon 5/5/14	Mon 5/5/14	6
8	Graduation	1 day	Sun 5/18/14	Sun 5/18/14	7
9	Team Meeting	66 days	Wed 1/29/14	Wed 4/30/14	
23	Client Meeting	61 days	Fri 1/31/14	Fri 4/25/14	