

UPRM “La Tiendita”

Icom 5016

Roberto Guzman, Giovanni Gardon, Cristian Melendez

roberto.guzman3@upr.edu , giovanni.gardon@upr.edu, cristian.melendez@upr.edu

I. Introduction

The UPRM book store on campus tends to suffer from long waiting lines, which discourage people from purchasing. The UPRM online BookStore will provide users to view and purchase in-store items. To help promote the service, the system could offer discounts from time to time to the students; this will motivate the users to buy the available merchandise. The application will be implemented as a web application because it will be more accessible for the general public. The technologies that are going to be used to do this project will be: Java, HTML5, CSS, AngularJS, PostgreSQL and Java Play.

NOTE

Application could provide delivery service (e.g. postal service, UPS) or in store pickup. (**Subject to change**)

II. Client App Description

The user will be able to access the webpage without signing into an account. While the user is in the webpage he will be able to navigate through different links that will provide information for the products as well as a search bar and a cart to manage the items that have been selected (e.g. Books, clothing, school supplies etc). In the shopping cart, the user can put any item he desires to buy and keep navigating through the store until he decide to pay the items he currently has in the shopping cart. At the moment of paying, the user will need to sign in. If the user already has an account, the user must sign in or create a new account if he does not have one. To create an account, the user will have to put the following information:

name, lastname, username, password, email, phone number and payment method.

The webpage provides the store's general contact information and a “My Account” section. In the “My Account” section, the user can manage his personal information. To implement this webpage we will use: HTML, CSS and AngularJS and AngularMaterial.

III. Server App Description

The server app will have a connection with the client app and the Database Management System (DBMS). This server app will receive requests (REST calls) from the client app and depending on the request, the server app will access the database and return the information depending on the REST call. The user can update his information in the “My Account”. The server will send requests in order to make those changes and update the database. For verification purposes, an email will be sent to the user once he has made his account. For the purchase process, the server app will receive a request for “proceed to checkout” or delete cart. If the request received is to proceed, the server app will make an authentication of the user. After authentication, the server will send a request to the payment provider and finish the payment process. Finally, it will notify the user if the order was processed or not. The technologies to implement this server app will be SQL, PgAdmin, PostgreSQL and Java Play.

Tables to be used:

- **Item:** general item data including name, description and price
- **Book:** Books additional information (author, year, etc.)
- **Electronics:** Usb drives, calculators, etc.

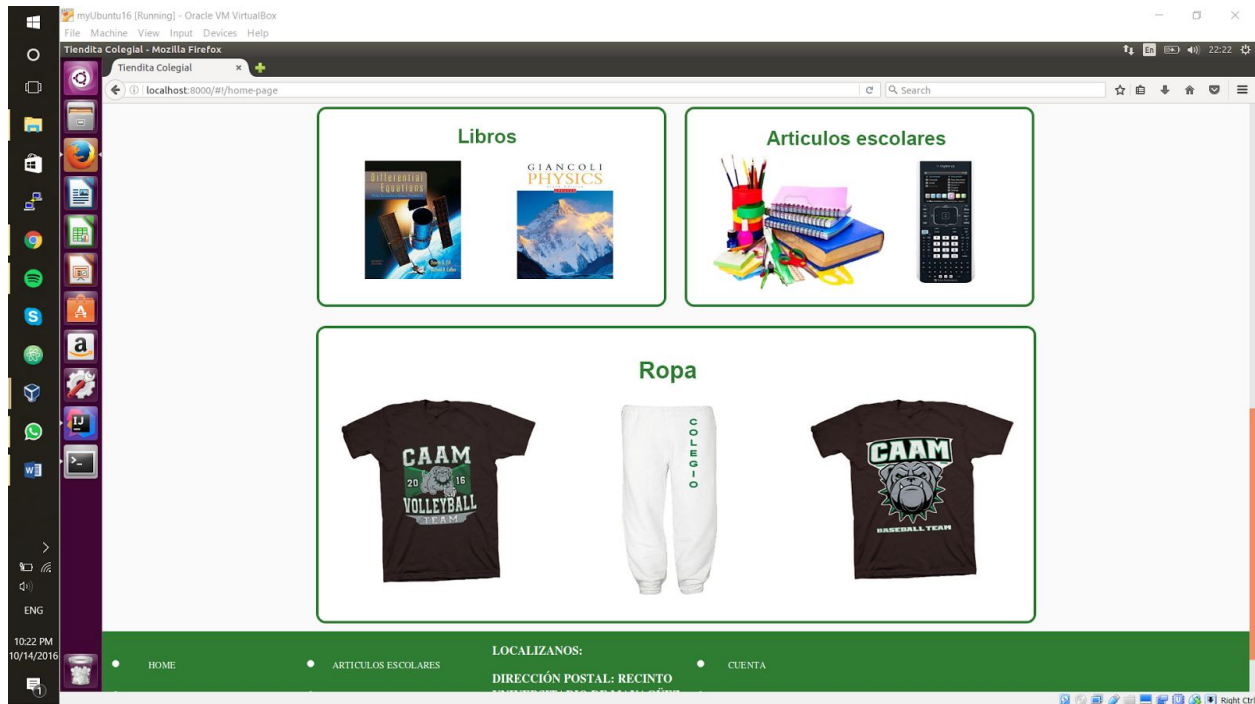
- **Users:** usernames, passwords, emails, etc.
- **History:** Sales history
- **Transactions:** transaction numbers and id's
- **Inventory:** Product availability
- **Shirts:** List of available shirts and additional information
- **Pants:** List of available pants and additional information
- **Hats:** List of available hats and additional information
- **Payment method:** Credit card information
- **Miscellaneous:** Key chains, umbrellas, etc.
- **Returned:** List of all the items that users have returned.

V. Client Side Screen Shots

1. Home Page

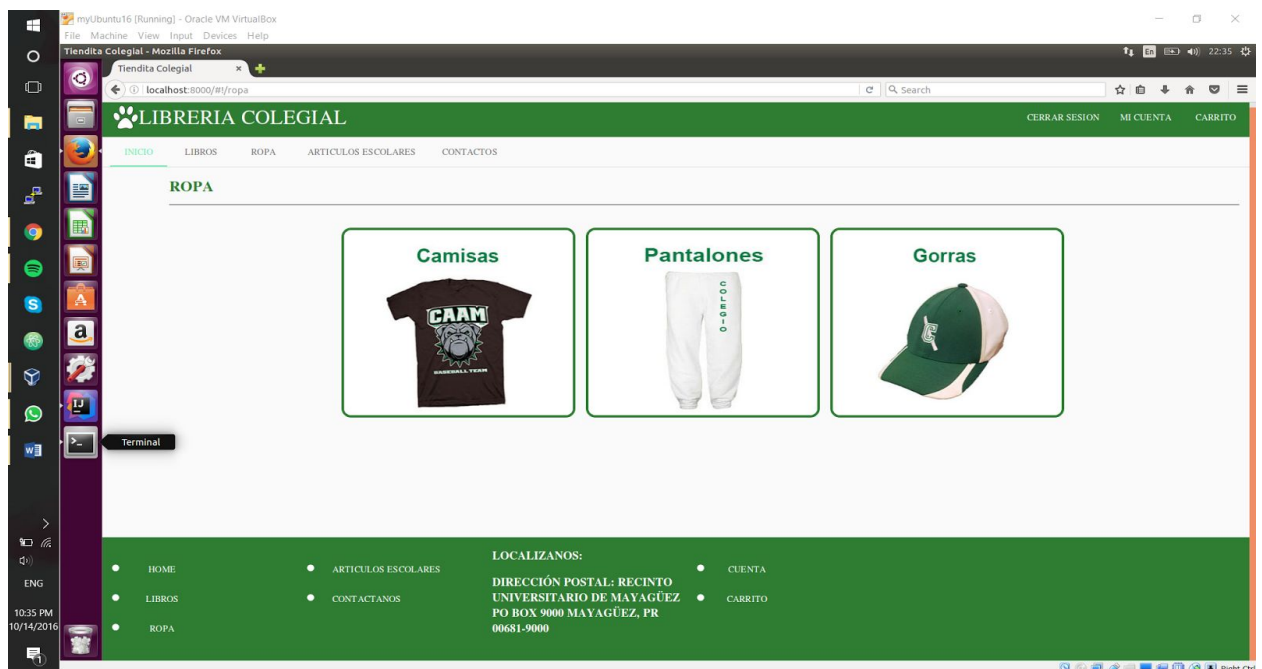
In the home page, the user can choose where to browse. As well as buttons to navigate through the web app



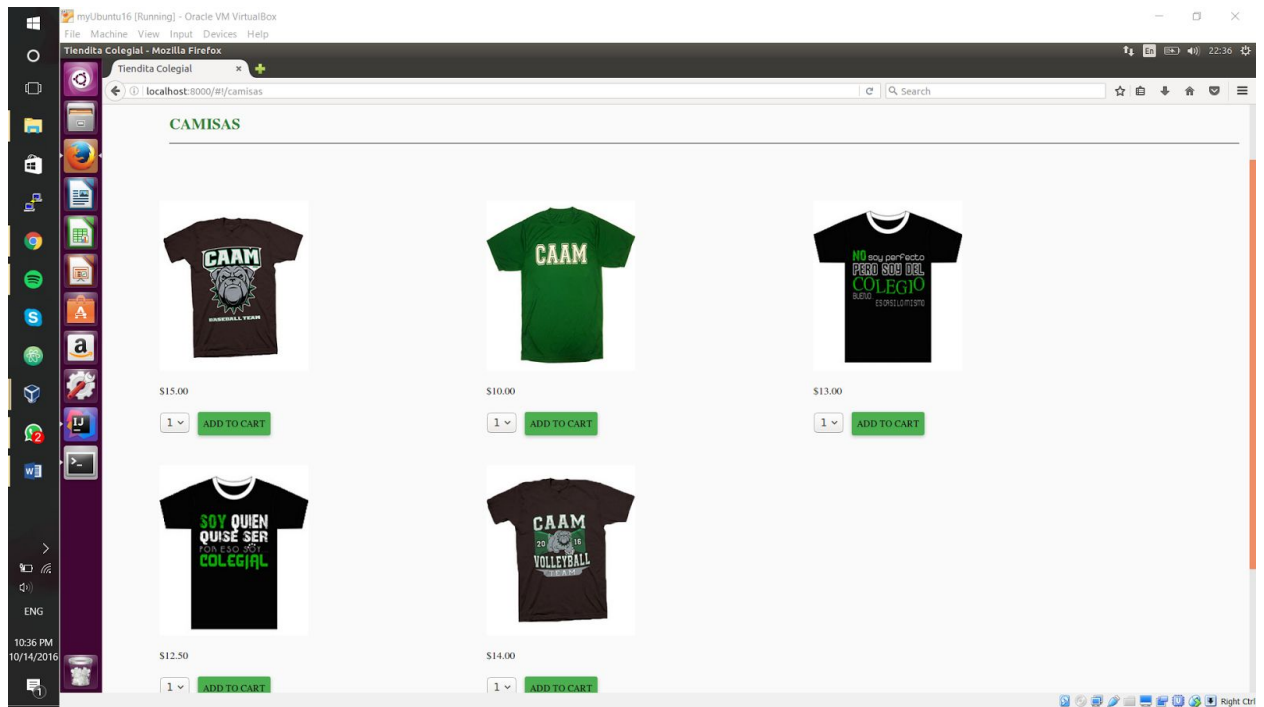


2. Items

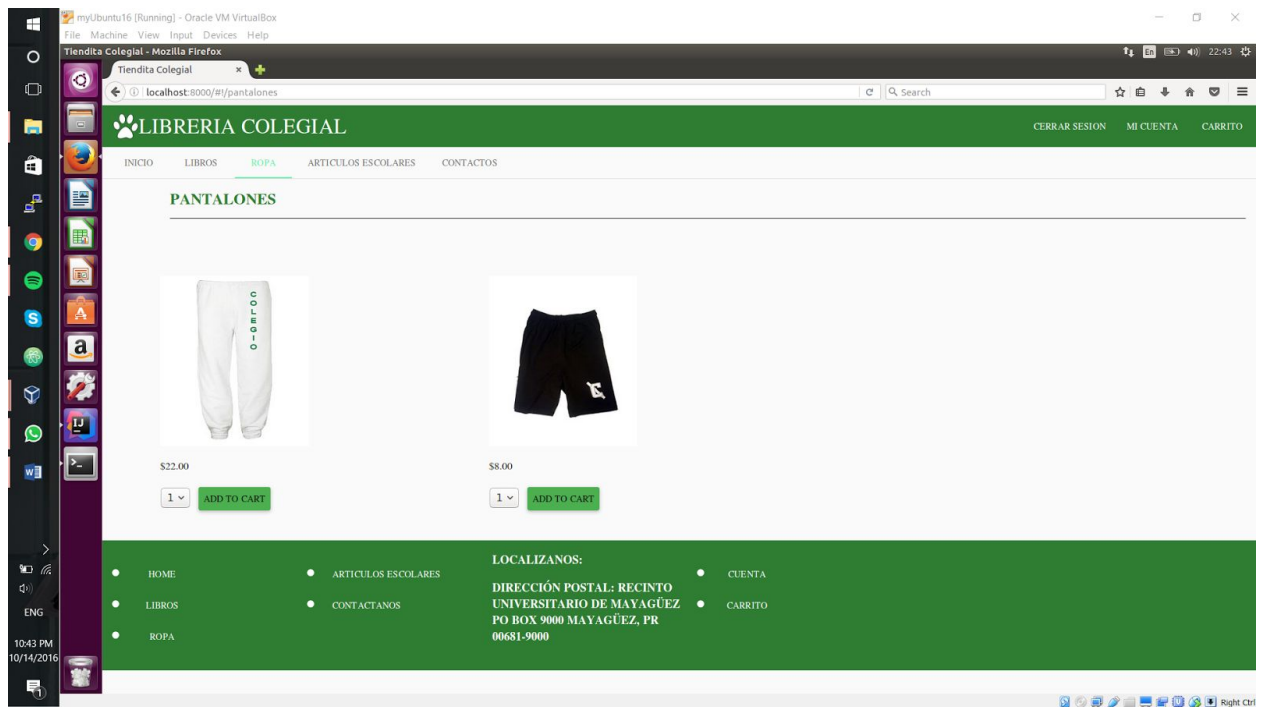
Here the user can decide to which category he/she wants to navigate. There are options presented in which the item can be added to the cart and proceed to checkout. These options are also available in the pages where there is merchandise.



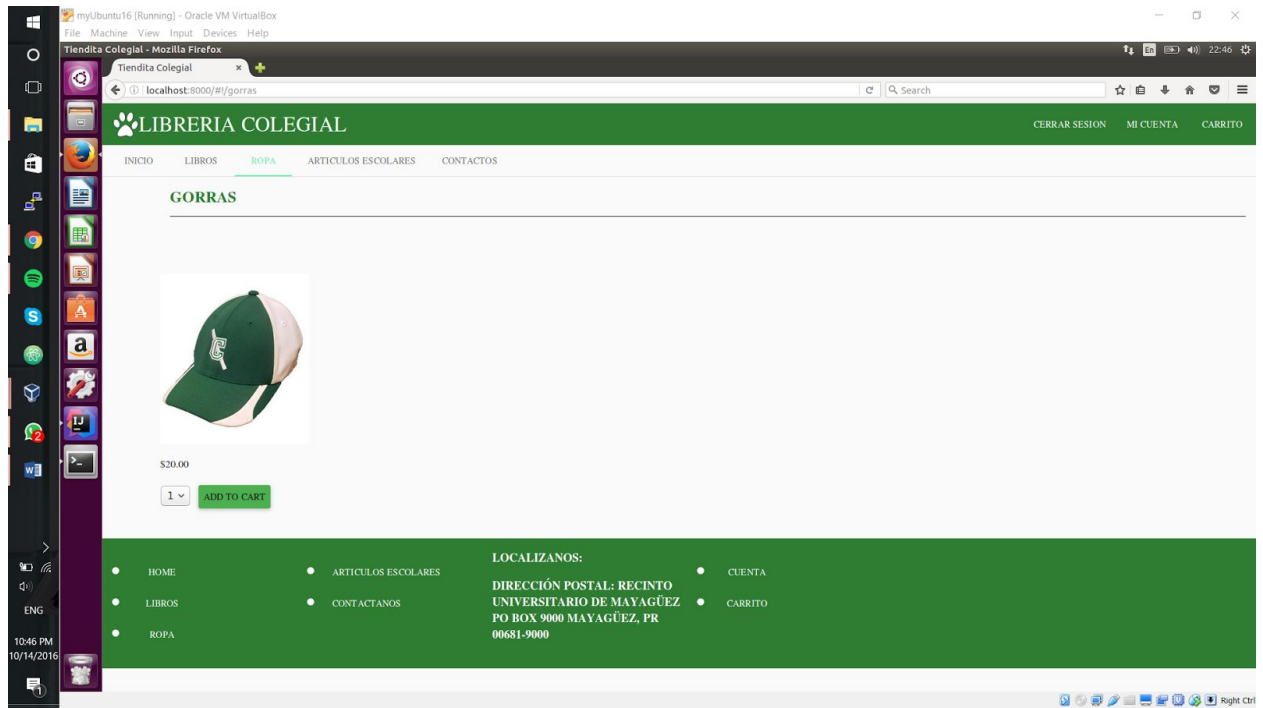
Shirts Page:



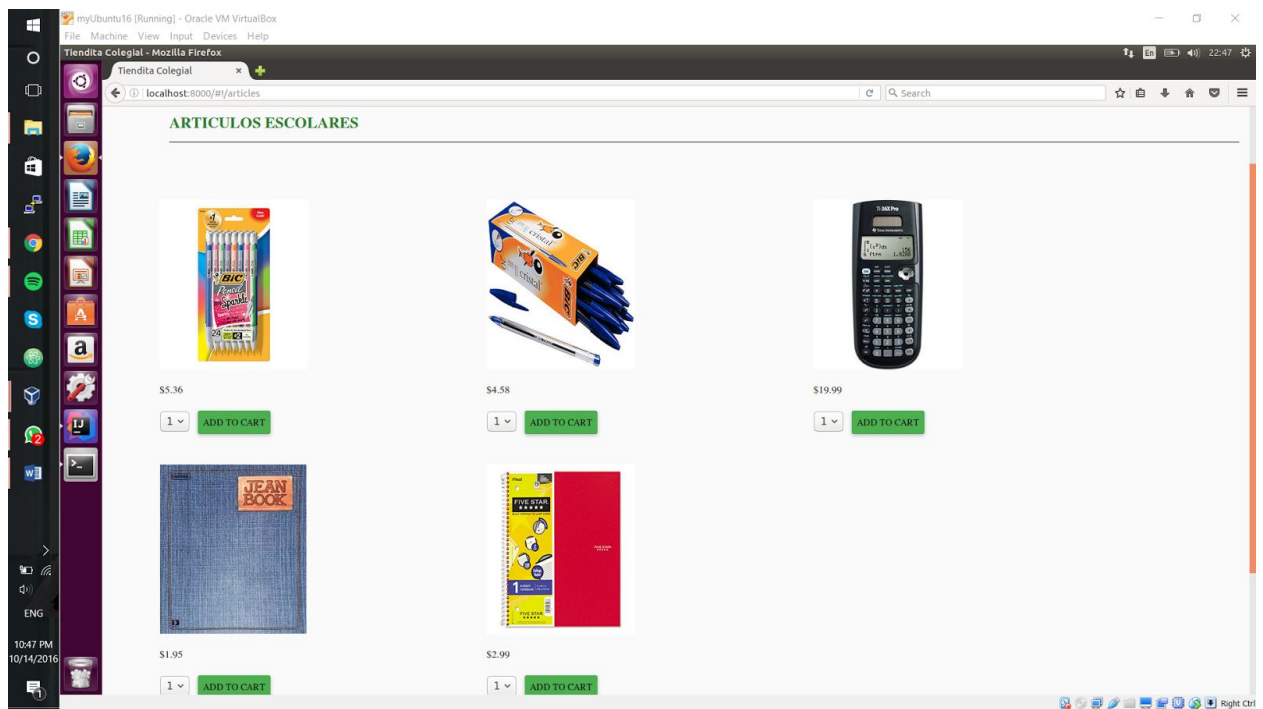
Pants Page:



Hats Page:




School Supplies Page:



3. Cart and Checkout

When items are chosen, these can be managed before proceeding to the checkout and pay.

Checkout:

LIBRERIA COLEGIAL

INICIAR SESIONCARRITO

INICIO LIBROS ROPA ARTICULOS ESCOLARES CONTACTOS

CARRITO






	Quantity	Amount	Total
X Chemistry by Silberberg	- 3	+ \$100.00	\$300.00
X camisa Verde CAAM	- 1	+ \$10.00	\$10.00
X No soy perfecto camisa	- 1	+ \$13.00	\$13.00
X Bic Pens	- 2	+ \$4.58	\$9.16
X Volleyball team camisa	- 1	+ \$14.00	\$14.00
X Soy quienquise ser camisa	- 2	+ \$12.50	\$25.00
		Total:	\$371.16

Checkout

CHECKOUT

PayPal

Buy Now



Payment Page:

LIBRERIA COLEGIAL

INICIAR SESIONCARRITO

INICIO LIBROS ROPA ARTICULOS ESCOLARES CONTACTOS

Pagar

Tarjeta de Credito

Card Number

123456789

CVV

222

Expiration Date

05/18

Resumen de Carrito:

10 items

\$371.16

PAGAR AHORA

4. Sign in process and registration

This page contains 2 forms, Registration and Sin In (written in spanish). If the user signs in he/she will be redirected to the his/hers account information. In the account information page, The user will be able to update his/hers information.

Login and register:

The screenshot shows the 'TIENDITA COLEGIAL' website interface. The header is green with the site name and a paw print logo on the left, and 'INICIAR SESION' and 'CARRITO' on the right. Below the header is a navigation bar with links: 'INICIO', 'LIBROS', 'ROPA', 'ARTICULOS ESCOLARES', and 'CONTACTOS'. To the right of these links is a search bar with the placeholder text 'nombre de articulo' and a green 'BUSQUEDA' button. The main content area is divided into two columns. The left column is titled 'INICIAR SESION' and contains two input fields: 'Email' and 'Contraseña', followed by a green 'INICIAR SESION' button. The right column is titled 'CREA TU CUENTA' and contains six input fields: 'Nombre', 'Apellidos', 'Nombre de Usuario', 'Contraseña', 'Reescribir Contraseña', 'Email', and 'Telefono', followed by a green 'CREAR CUENTA' button. The footer is a light gray bar with the text 'Copyright 2016 La Tiendita Colegial'.

TIENDITA COLEGIAL

INICIO LIBROS ROPA ARTICULOS ESCOLARES CONTACTOS

nombre de articulo **BUSQUEDA**

INICIAR SESION

Email

Contraseña

INICIAR SESION

CREA TU CUENTA

Nombre

Apellidos

Nombre de Usuario

Contraseña

Reescribir Contraseña

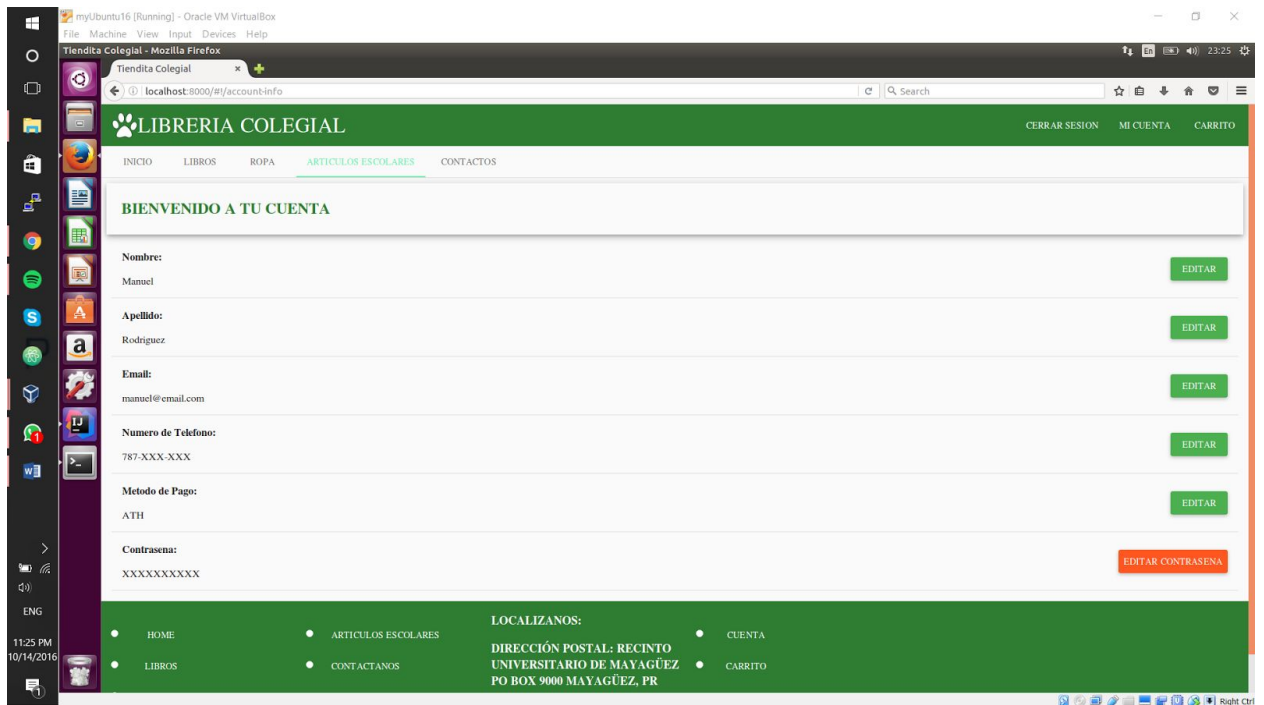
Email

Telefono

CREAR CUENTA

Copyright 2016 La Tiendita Colegial

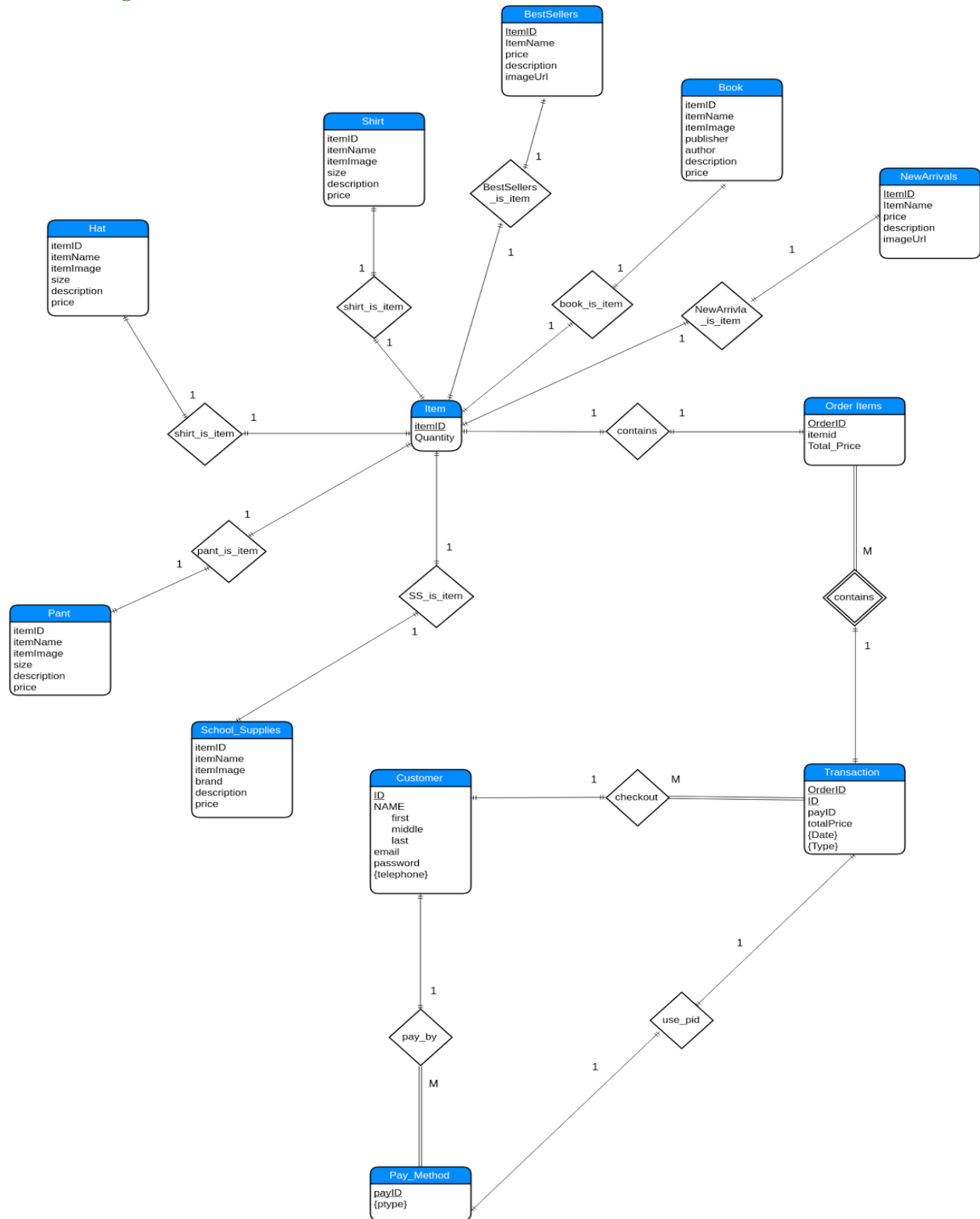
Account Information:



5. Admin account add item



VI. E-R Diagram



VII Tables

```

CREATE TABLE public.best sellers(
  itemid integer NOT NULL DEFAULT nextval('best sellers_itemid_seq'::regclass),
  itemname character varying(30),
  price numeric(10,2),
  description character varying(30),
  imageurl character varying(100),
  CONSTRAINT best sellers_pkey PRIMARY KEY (itemid),
  CONSTRAINT best sellers_itemid_fkey FOREIGN KEY (itemid)
    REFERENCES public.item (itemid) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION
)

```

```

CREATE TABLE public.book
(
  itemid integer NOT NULL DEFAULT nextval('book_itemid_seq'::regclass),
  price numeric(10,2) NOT NULL,
  itemname character varying(30) NOT NULL,
  author character varying(30),
  publisher character varying(30),
  description text,
  imageurl character varying(100),
  CONSTRAINT book_pkey PRIMARY KEY (itemid),
  CONSTRAINT book_itemid_fkey FOREIGN KEY (itemid)
    REFERENCES public.item (itemid) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION
)

```

```

CREATE TABLE public.customer(
  cid integer NOT NULL DEFAULT nextval('customer_cid_seq'::regclass),
  cname character varying(50) NOT NULL,
  password character varying(20),
  ctelephone character varying(10),
  cemail character varying(50),
  CONSTRAINT customer_pkey PRIMARY KEY (cid)
)

```

```

CREATE TABLE public.hat(
  itemid integer NOT NULL DEFAULT nextval('hat_itemid_seq'::regclass),
  price numeric(10,2) NOT NULL,
  itemname character varying(30) NOT NULL,
  itemimage bytea,
  size character varying(5),
  description text,
  CONSTRAINT hat_pkey PRIMARY KEY (itemid),
  CONSTRAINT hat_itemid_fkey FOREIGN KEY (itemid)
    REFERENCES public.item (itemid) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION)
CREATE TABLE public.item(

```

```

    itemid integer NOT NULL DEFAULT nextval('item_itemid_seq'::regclass),
    quantity numeric(10,0),
    CONSTRAINT item_pkey PRIMARY KEY (itemid)
)

CREATE TABLE public.itemordered(
    orderid integer NOT NULL DEFAULT nextval('itemordered_orderid_seq'::regclass),
    itemid integer NOT NULL DEFAULT nextval('itemordered_itemid_seq'::regclass),
    CONSTRAINT itemordered_itemid_fkey FOREIGN KEY (itemid)
        REFERENCES public.item (itemid) MATCH SIMPLE
        ON UPDATE NO ACTION ON DELETE NO ACTION,
    CONSTRAINT itemordered_orderid_fkey FOREIGN KEY (orderid)
        REFERENCES public.transactions (orderid) MATCH SIMPLE
        ON UPDATE NO ACTION ON DELETE NO ACTION
)

CREATE TABLE public.newarrivals(
    itemid integer NOT NULL DEFAULT nextval('newarrivals_itemid_seq'::regclass),
    itemname character varying(30),
    price numeric(10,2),
    description character varying(30),
    imageurl character varying(100),
    CONSTRAINT newarrivals_pkey PRIMARY KEY (itemid),
    CONSTRAINT newarrivals_itemid_fkey FOREIGN KEY (itemid)
        REFERENCES public.item (itemid) MATCH SIMPLE
        ON UPDATE NO ACTION ON DELETE NO ACTION
)

CREATE TABLE public.pant(
    itemid integer NOT NULL DEFAULT nextval('pant_itemid_seq'::regclass),
    price numeric(10,2) NOT NULL,
    itemname character varying(30) NOT NULL,
    itemimage bytea,
    size character varying(5),
    description text,
    CONSTRAINT pant_pkey PRIMARY KEY (itemid),
    CONSTRAINT pant_itemid_fkey FOREIGN KEY (itemid)
        REFERENCES public.item (itemid) MATCH SIMPLE
        ON UPDATE NO ACTION ON DELETE NO ACTION
)

CREATE TABLE public.paymentinfo(

```

```

payid integer NOT NULL DEFAULT nextval('paymentinfo_payid_seq'::regclass),
cid integer NOT NULL DEFAULT nextval('paymentinfo_cid_seq'::regclass),
cardnum numeric(16,0) NOT NULL,
expdate character(5) NOT NULL,
ownerofcard character varying(40) NOT NULL,
address character varying(100) NOT NULL,
securitynum numeric(3,0),
CONSTRAINT paymentinfo_pkey PRIMARY KEY (payid),
CONSTRAINT paymentinfo_cid_fkey FOREIGN KEY (cid)
    REFERENCES public.customer (cid) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION
)

```

```

CREATE TABLE public.schoolsupplies(
    itemid integer NOT NULL DEFAULT nextval('schoolsupplies_itemid_seq'::regclass),
    price numeric(10,2) NOT NULL,
    itemname character varying(30) NOT NULL,
    itemimage bytea,
    brand character varying(30),
    description text,
    CONSTRAINT schoolsupplies_pkey PRIMARY KEY (itemid),
    CONSTRAINT schoolsupplies_itemid_fkey FOREIGN KEY (itemid)
        REFERENCES public.item (itemid) MATCH SIMPLE
        ON UPDATE NO ACTION ON DELETE NO ACTION
)

```

```

CREATE TABLE public.shirt(
    itemid integer NOT NULL DEFAULT nextval('shirt_itemid_seq'::regclass),
    price numeric(10,2) NOT NULL,
    itemname character varying(30) NOT NULL,
    itemimage bytea,
    size character varying(5),
    description text,
    CONSTRAINT shirt_pkey PRIMARY KEY (itemid),
    CONSTRAINT shirt_itemid_fkey FOREIGN KEY (itemid)
        REFERENCES public.item (itemid) MATCH SIMPLE
        ON UPDATE NO ACTION ON DELETE NO ACTION
)

```

```

CREATE TABLE public.transactions(

```

```
orderid integer NOT NULL DEFAULT nextval('transactions_orderid_seq'::regclass),
cid integer NOT NULL DEFAULT nextval('transactions_cid_seq'::regclass),
tdate character(10),
typeoftransaction character varying(20),
payid integer NOT NULL DEFAULT nextval('transactions_payid_seq'::regclass),
totalprice numeric(10,0),
CONSTRAINT transactions_pkey PRIMARY KEY (orderid),
CONSTRAINT transactions_cid_fkey FOREIGN KEY (cid)
    REFERENCES public.customer (cid) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION,
CONSTRAINT transactions_payid_fkey FOREIGN KEY (payid)
    REFERENCES public.paymentinfo (payid) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION
)
```