Election Fraud

To combat election fraud, your city is instituting a new voting procedure. The ballot has a letter associated with every selection a voter may make. A sample ballot is shown.

1. VOTE FOTR MAYOR

A. Pincher, Penny 🞎

B. Dover, Skip 🞎

C. Perman, Sue 🞎

2. PROPOSITION 17

D. YES 🞎

E. NO 🞎

3. MEASURE 1

F. YES 🞎

G. NO 🞎

4. MEASURE 2

H. YES 🞎

I. NO 🞎

After submitting their ballot, every voter receives a receipt that has a unique ID number and a record of their voting selections. For example, a voter that submits a ballot for Sue Perman, Yes on Proposition 17, No on Measure 1, and Yes on Measure 2 might receive a receipt with

ID 4925 : CDGH

The next day, the city posts all votes on their web page sorted by ID number. This allows a voter to confirm their submission and allows anyone to count the vote totals for themselves. A sample list for the sample ballot is shown.

ID VOTES

4925 CDGH

4926 AEGH

4927 CDGI

4928 BEGI

4929 ADFH

Write a program that reads the posted voting list from a file and outputs the percent of votes cast for each ballot item. You may assume that the file does not have any header lines. Each line will contain a voter ID and a string representing votes. Define a class named Voter that stores an individual’s voting record. The class should have a constructor that takes as input a string of votes (e.g. “CDGH”) and a voter ID. It should have accessors for the voter ID and the vote for a specific question.

Online Supermarket – TDD Case Study

An ASP.NET MVC web based application is needed to allow shoppers to shop online for specific grocery items. The application’s features are described below.

# Admin CRUD Features

## Create Item: /item/create/

## Read Item: /item/details/*<item id>*

## Update Item: /item/edit/*<item id>*

## Delete Item: /item/delete/*<item id>*

There must be a built in user in an ‘admin’ role. This user must have email ‘admin@user.com’ and password ‘Pass123!’ Only the admin user is allowed to CRUD supermarket items. When the admin user logs in, he or she is redirected to the items index page (/items/index/). The admin user has no supermarket profile (see below) to maintain. You are allowed to scaffold the controller and views to CRUD the items.

Each item has the following properties:

* **Id** – the item’s id, primary key, auto-increment
* **Name** – the item’s name, required, maximum length: 50
* **Type** – the item’s type, one of: Cereal, Can Goods, Bread, Candy
* **Price** – the item’s price, greater than zero
* **Amount in stock** – must be at least 0 and at most 1000

# Title Bar Feature

The title bar is visible on all pages.

* A nice image (you decide on something appropriate)
* Name of the application: **Online Supermarket**
* ‘Home’ link – takes the user to the Home page
* ‘View Profile’ link – takes the shopper to the view profile page (not visible to admin users)
* ‘About’ link – takes the user to the about page – the about page simply states the name and email of the developer – the email is a mailto link

# Main Shopping

/home/index/

This action redirects admin users to the CRUD items index page.

If a shopper has no supermarket profile, he or she is redirected to create his or her supermarket profile.

The shopper is presented with:

* 4 ‘aisles’
  + Cereal, Can Goods, Bread, Candy
  + Each aisle lists the associated items with the following: name, price, amount in stock, and a **Buy** button. If the amount in stock is zero, then the Buy button is disabled.
* Number of items in shopping cart and the shopping cart’s total. This is displayed near the top and bottom of the view. Note that if the user logs out and then logs back in then the shopping cart data is kept.
* A **View Shopping Cart** link

# Buy Item

/shoppingcart/buy/***<item id>***

Admin users cannot access this action.

This action displays the details of the item and allows the user to enter the quantity to buy. The quantity to buy cannot exceed the item’s amount in stock. When the shopper submits this data, the item is added to the shopping cart, and the amount in stock is reduced by the quantity to buy. The shopper is redirected to the main shopping page after purchase.

There should be a link back to main shopping.

# Shopping Cart

/shoppingcart/index/

Admin users cannot access this action.

This action lists all items in the shopping cart along with the total price for each item bought. There is to be a ‘**Remove**’ link next to each item. There should also be a grand total price of the items in the shopping cart.

There should be 3 links: **Empty cart (No Purchase)**, **Checkout**, and Back to main shopping

# Remove Item

/shoppingcart/remove/***<item id>***

Admin users cannot access this action.

This action displays the details of the item and allows the user to confirm removal. If the shopper confirms removal, the item is removed from the shopping cart, and the amount in stock increased by the quantity to buy. The shopper is redirected to the shopping cart page after removal.

There should be a link back to the shopping cart page.

# Empty Cart

/shoppingcart/empty/

Admin users cannot access this action.

This action allows the user to confirm emptying the cart. If the shopper confirms emptying, the items are removed from the shopping cart, and the amount in stock for each item is increased by the quantity to buy for that item. The shopper is redirected to the main shopping page.

There should be a link back to the shopping cart page.

# Checkout

/shoppingcart/checkout/

Admin users cannot access this action.

This action displays all items and the grand total price for all items in the cart and allows the user to confirm paying for the items in the cart. If the shopper confirms paying, the items are removed from the shopping cart. The shopper is redirected to the main shopping page.

There should be a link back to the shopping cart page.

# Create Profile

/profile/create/

Admin users cannot access this action.

This action allows a shopper to create his or her supermarket profile. The shopper is not allowed back here once profile is created.

* **Email** – Primary key, maps to the identity email (but not a foreign key)
* **Username** – Unique to each shopper, required, does not change once created, not the same as the username in the identity
* **First name** – required, maximum length 50
* **Last name** – required, maximum length 50
* **Street address** – required
* **State** – required, must be two characters, and must be one of the standard US state codes
* **Zip** – required, must be 5 digits

The shopper is redirected to main shopping after the profile is created.

# View Profile

/profile/details/

Admin users cannot access this action.

This action allows the shopper to view his or her supermarket profile:

* Full name (Last, First)
* Street address
* State
* Zip

There must be links to **edit profile**, **delete profile**, and back to main shopping

# Edit Profile

/profile/edit/

Admin users cannot access this action.

This action allows the shopper to edit his or her supermarket profile:

* **First name** – required, maximum length 50
* **Last name** – required, maximum length 50
* **Street address** – required
* **State** – required, must be two characters, and must be one of the standard US state codes
* **Zip** – required, must be 5 digits

There must be a link back to main shopping.

The shopper is redirected to view profile after completion.

# Delete Profile

/profile/delete/

Admin users cannot access this action.

This action allows the shopper to delete his or her profile after confirmation.

There must be a link back to main shopping.

The shopper is redirected to main shopping after completion.