Capstone Project

Jeremiah Lateef
Mohammed Khursiwala
Tyler Rosario

Agenda

Initial Impressions

Design Approach

Challenges

Overview UML

Shopper & User Interface

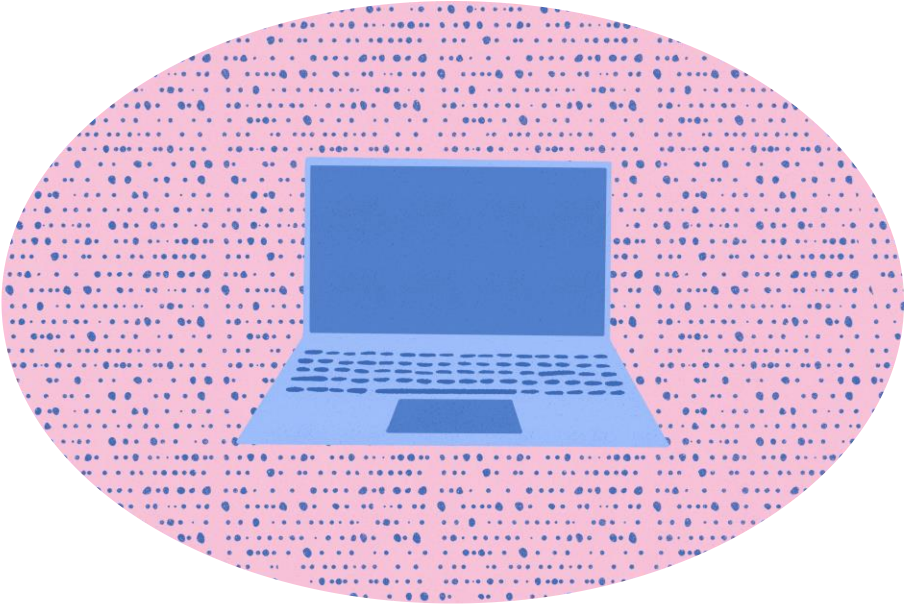
Children Extension

Software Engineering Metrics

Code Review

Takeaways

App Demo



Initial Impressions

- Duplicate code
- Different naming conventions
- Different designs
- Methods expected different parameters
- Lots of errors
- We each wrote code for ourselves without knowing other people would use it
- Nothing worked!

Jeremiah – Grocery Code

Mohammed – Person Code

Tyler – Course code

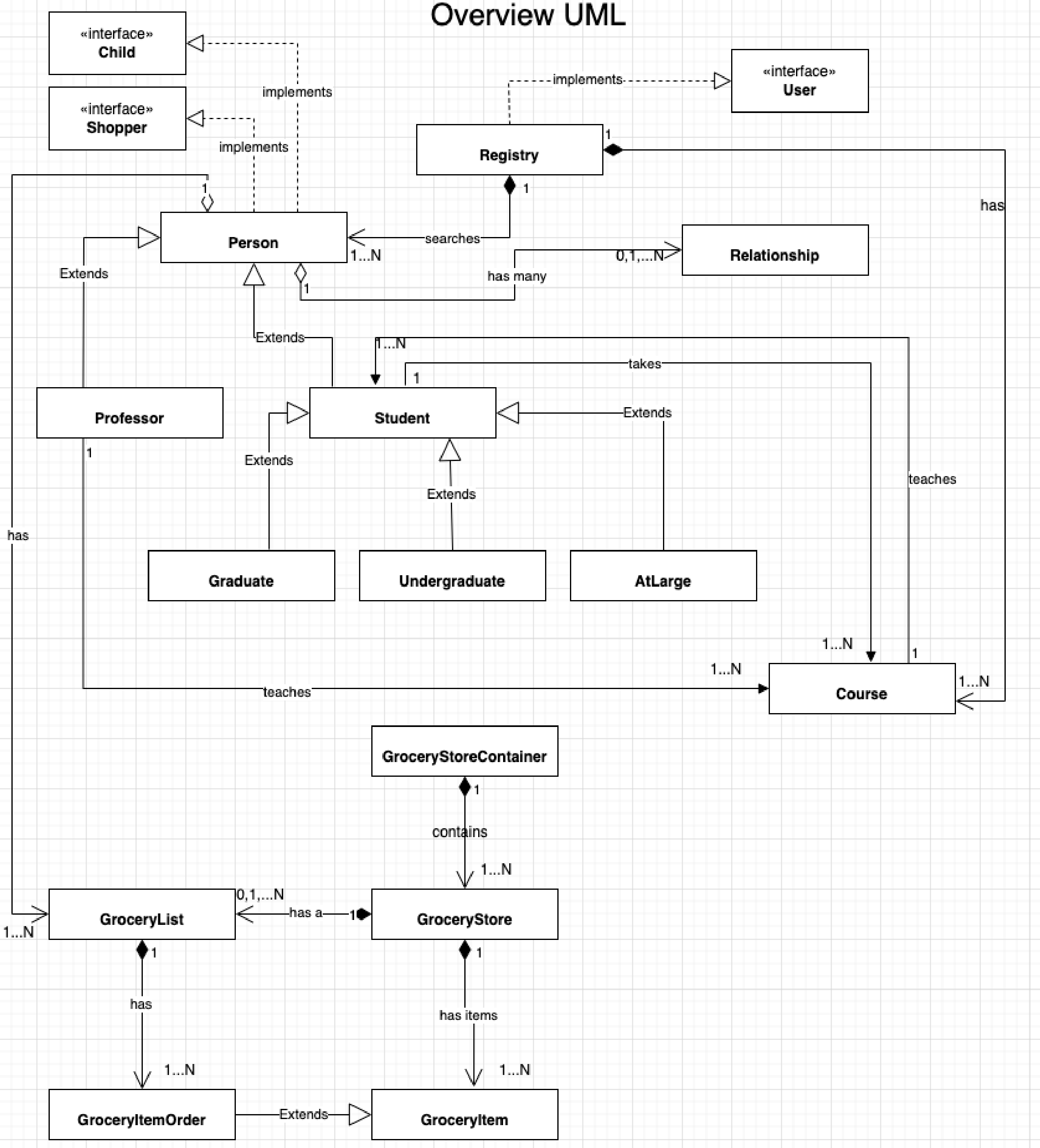
Design Approach

1. Meet regularly
2. Divide & Conquer
3. Use shared GitHub repo
4. Encapsulation, Inheritance, & Polymorphism
5. Code review
6. Documentation

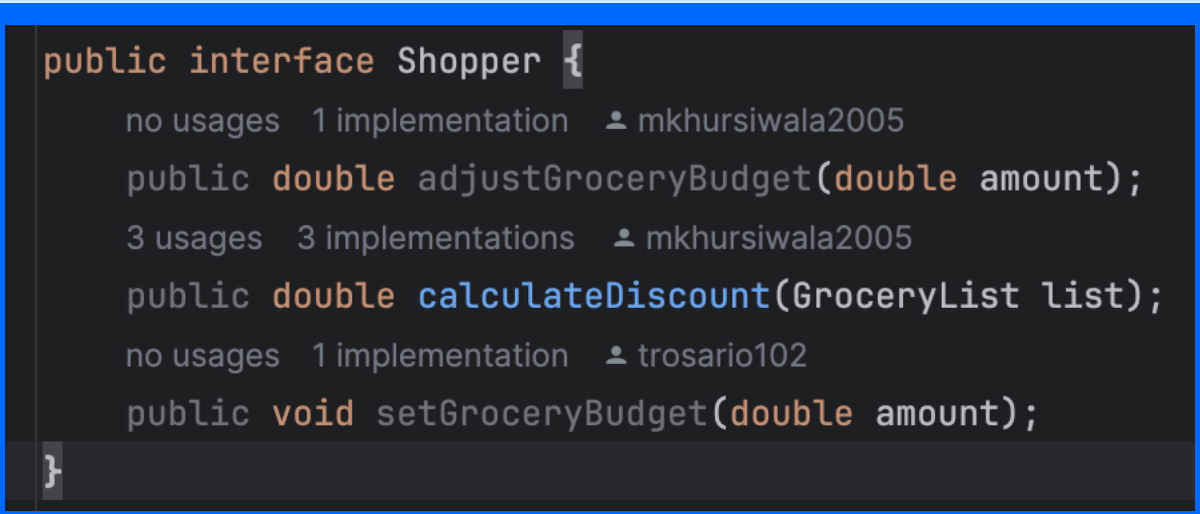
Challenges

- Finding times to meet outside of class
- We had to modify our code to work with each other as opposed to working from the same design
- Our design decisions were made in isolation
- Lots of things had to be changed like return types, names of methods, duplicate code
- Coding as a team with a shared repo
- Code showed remnants of working individually like of comments or clear names for things

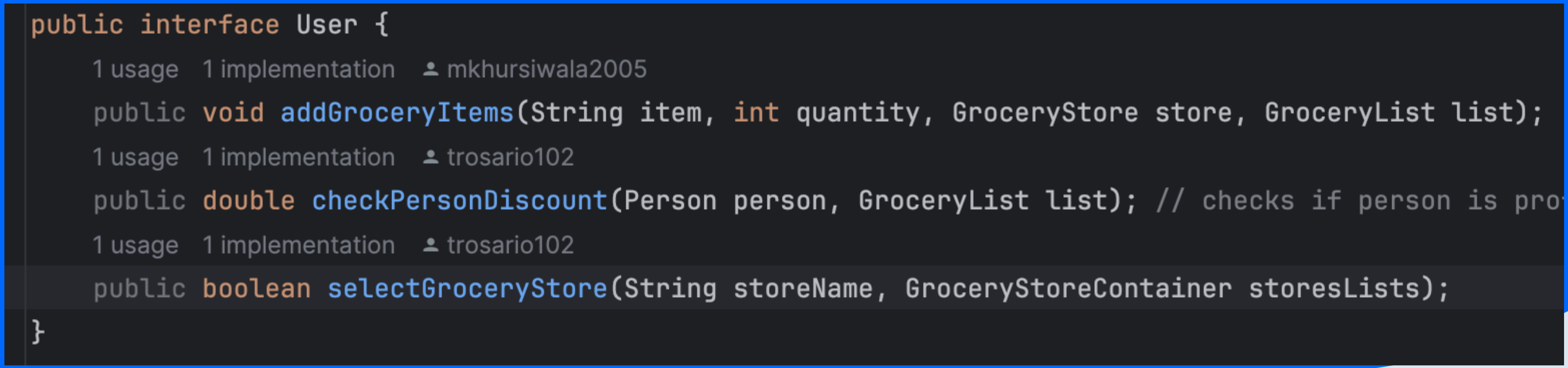
Overview UML



Shopper & User Interface



```
public interface Shopper {
    no usages   1 implementation   mkhursiwala2005
    public double adjustGroceryBudget(double amount);
    3 usages   3 implementations   mkhursiwala2005
    public double calculateDiscount(GroceryList list);
    no usages   1 implementation   trosario102
    public void setGroceryBudget(double amount);
}
```

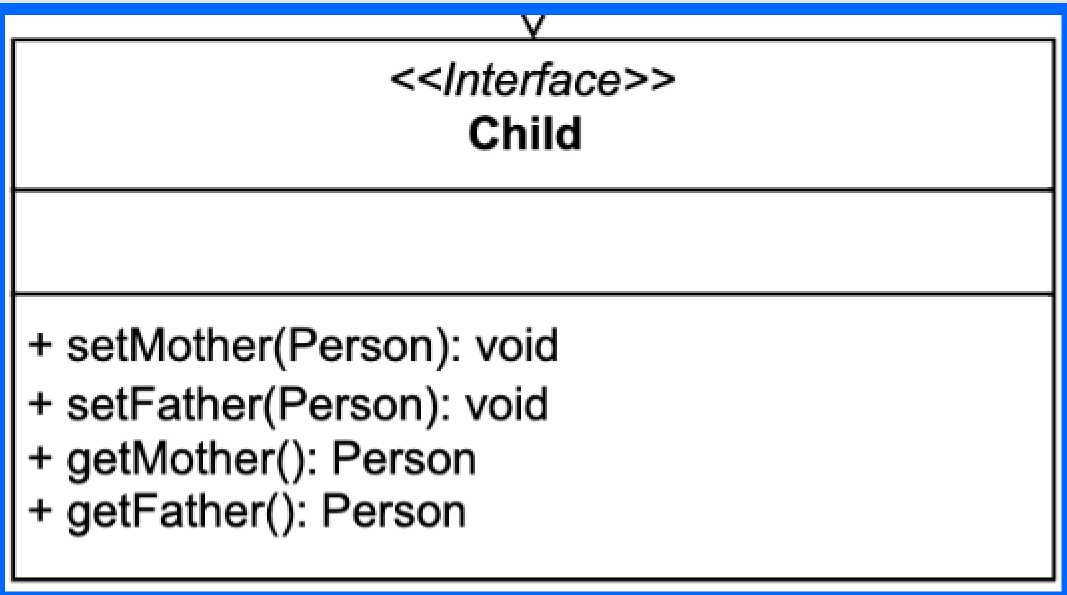


```
public interface User {
    1 usage   1 implementation   mkhursiwala2005
    public void addGroceryItems(String item, int quantity, GroceryStore store, GroceryList list);
    1 usage   1 implementation   trosario102
    public double checkPersonDiscount(Person person, GroceryList list); // checks if person is pro
    1 usage   1 implementation   trosario102
    public boolean selectGroceryStore(String storeName, GroceryStoreContainer storesLists);
}
```

Key Part in Our Program

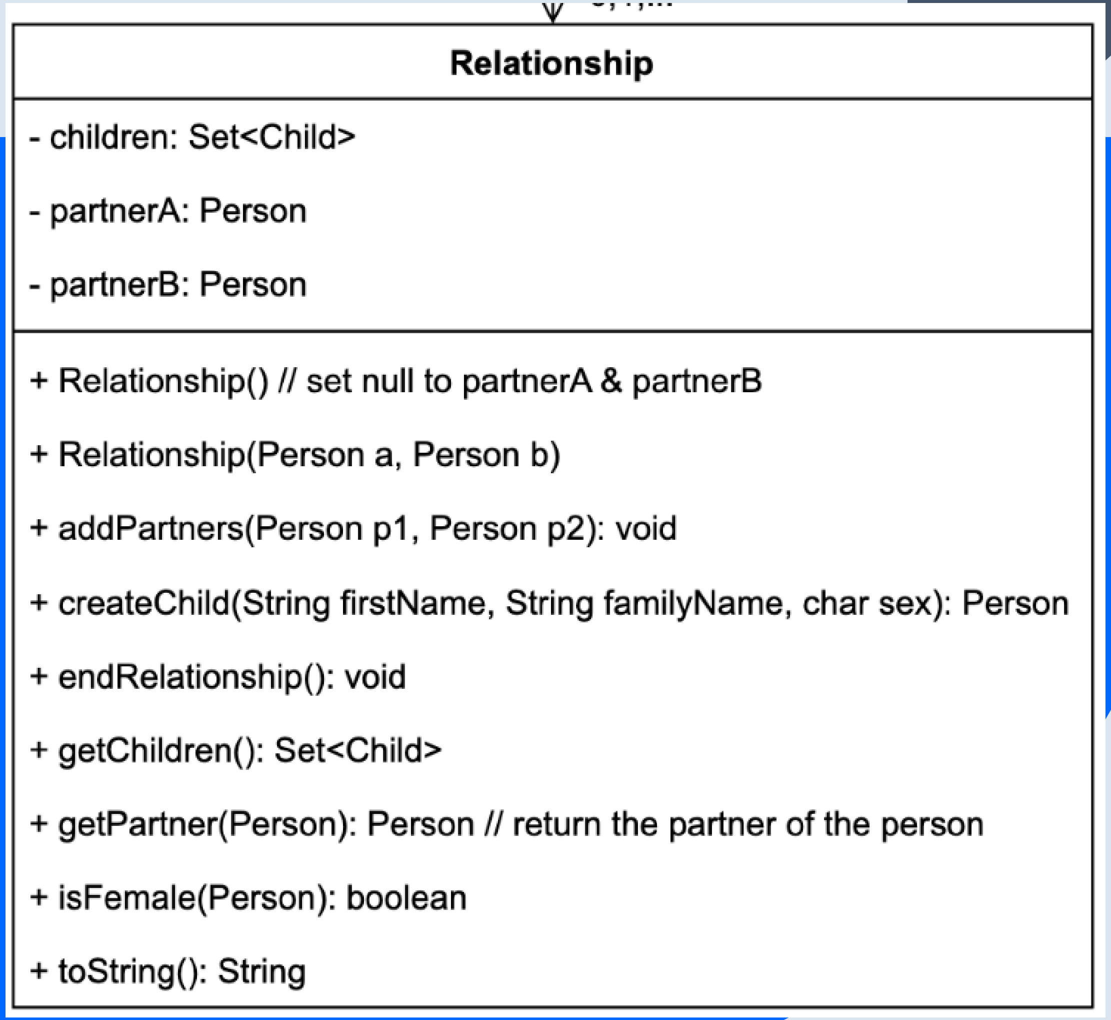
- Person
- Registry
- Course

Children Extension



<<Interface>>
Child

+ setMother(Person): void
+ setFather(Person): void
+ getMother(): Person
+ getFather(): Person



Relationship

- children: Set<Child>

- partnerA: Person

- partnerB: Person

+ Relationship() // set null to partnerA & partnerB

+ Relationship(Person a, Person b)

+ addPartners(Person p1, Person p2): void

+ createChild(String firstName, String familyName, char sex): Person

+ endRelationship(): void

+ getChildren(): Set<Child>

+ getPartner(Person): Person // return the partner of the person

+ isFemale(Person): boolean

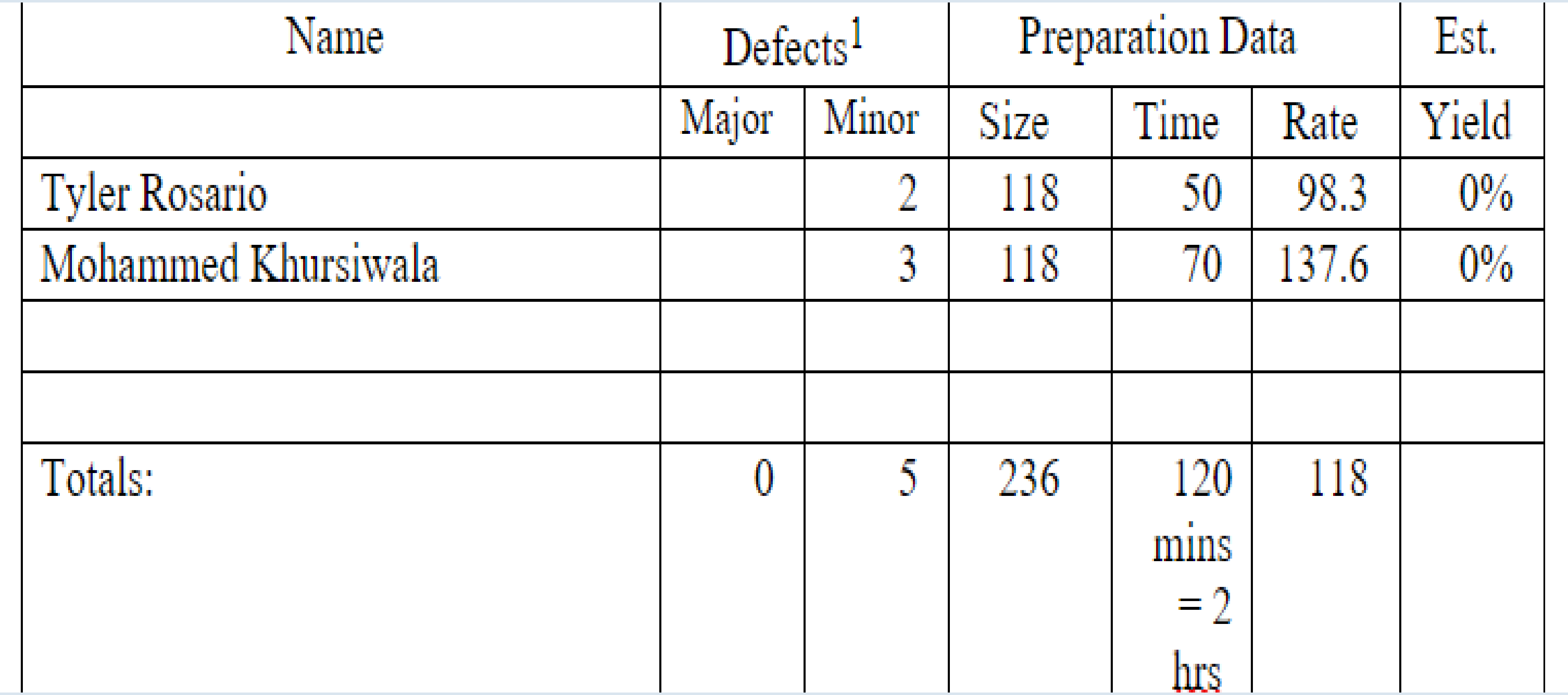
+ toString(): String

Software Engineering Metrics

- Number of classes: 14
- Number of interfaces: 3
- Number of methods: 115
- Lines of code: 1144
- Number of unresolved bugs: 22

Code Review Results (Jeremiah)

GroceryItem, GroceryStore, GroceryList



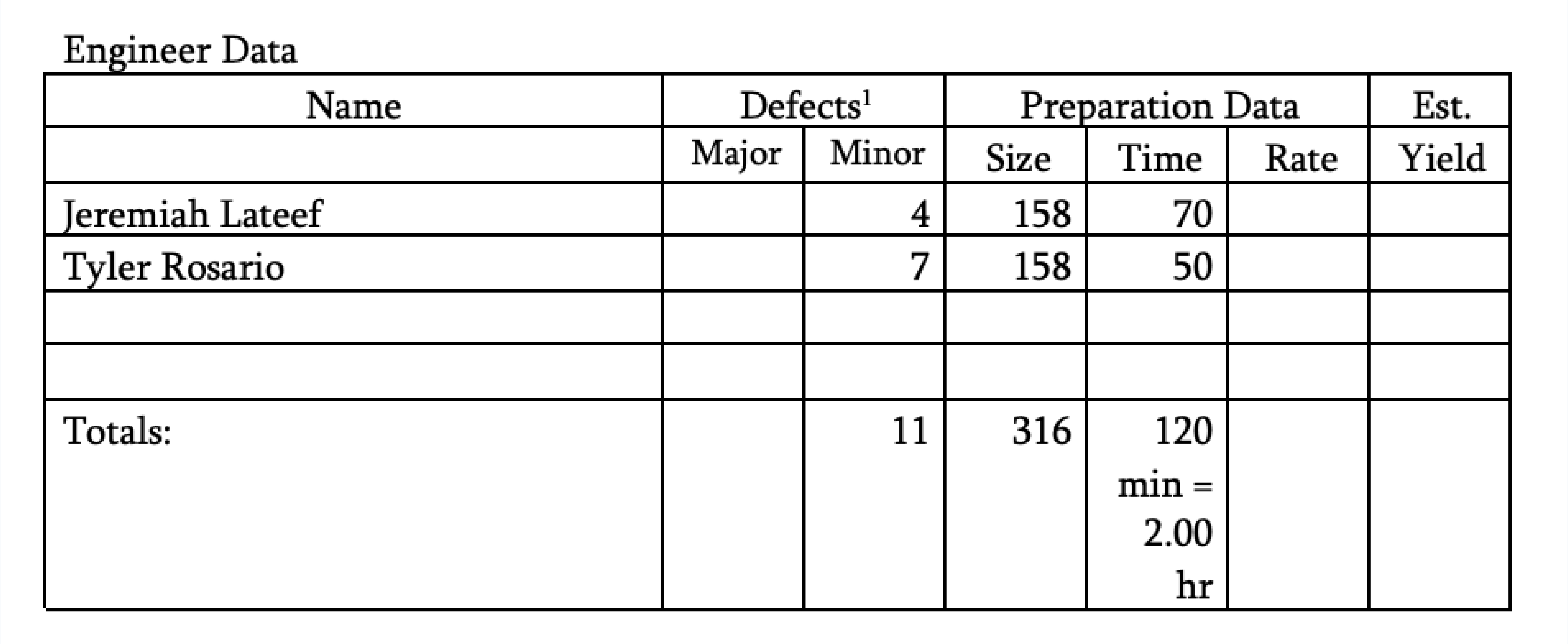
Name	Defects[1]		Preparation Data			Est.
	Major	Minor	Size	Time	Rate	Yield
Tyler Rosario		2	118	50	98.3	0%
Mohammed Khursiwala		3	118	70	137.6	0%
Totals:	0	5	236	120 mins = 2 hrs	118	

Code Review Results (Jeremiah)

- Some of the complex methods did not have any comment explaining what it does.
- There were redundant java.util package in some of the files.
- The code had some stylistic issues that had some minor impact on the beauty of the code, i.e. extra spaces between methods.
- A few of the comments were reductive, and did not explain the code well.

Code Review Results (Mohammed)

Person.java



Engineer Data

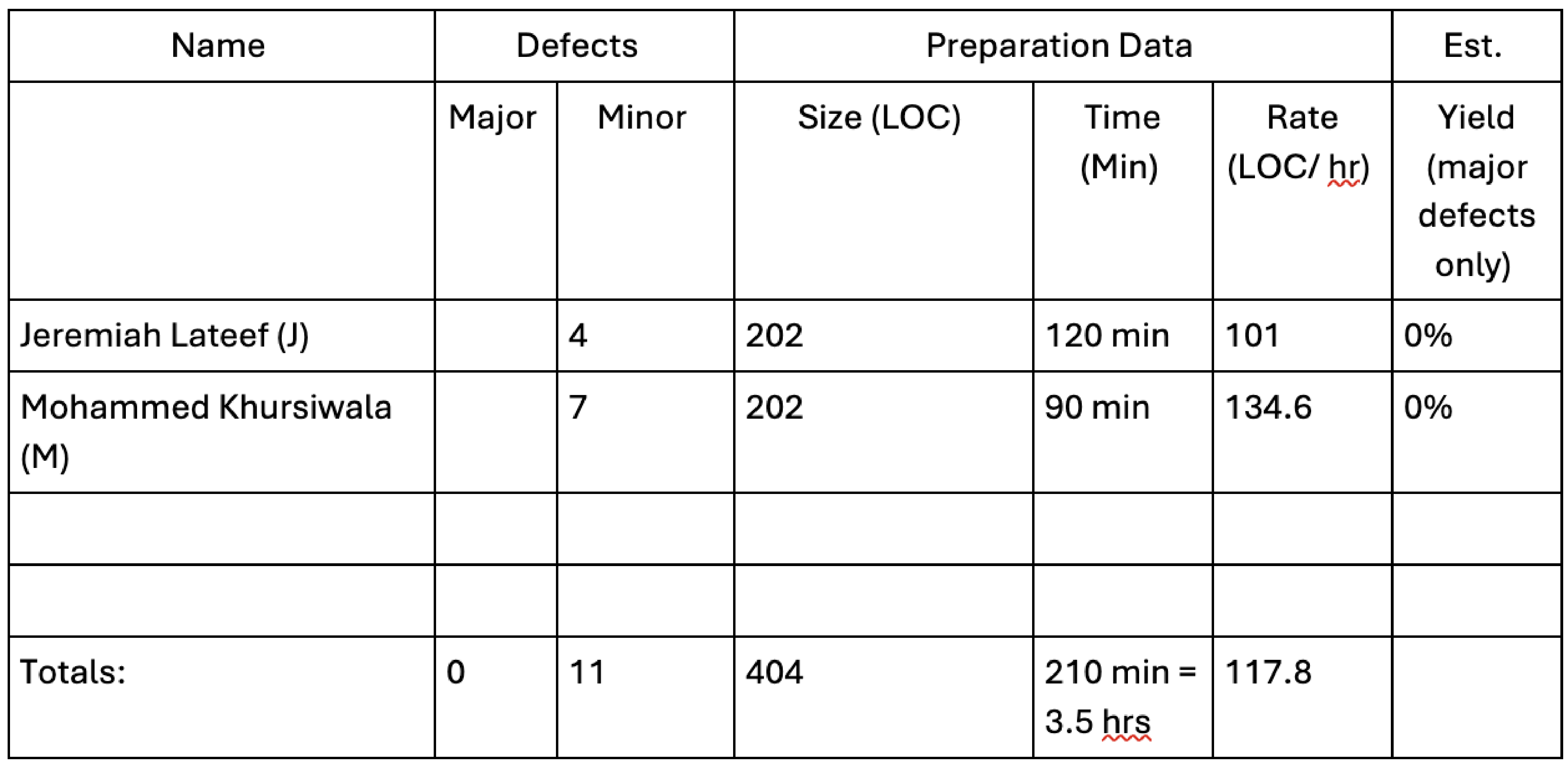
Name	Defects[1]		Preparation Data			Est.
	Major	Minor	Size	Time	Rate	Yield
Jeremiah Lateef		4	158	70		
Tyler Rosario		7	158	50		
Totals:		11	316	120 min = 2.00 hr		

Code Review Results (Mohammed)

- Comments are not written in full, i. e they're abbreviated
- Over importing with java.util.*
- Extra spaces before some of the methods
- Unused some of methods in Person class
- Not many comments to explain code

Code Review Results (Tyler)

Course, Relationship



Name	Defects		Preparation Data			Est.
	Major	Minor	Size (LOC)	Time (Min)	Rate (LOC/ hr)	Yield (major defects only)
Jeremiah Lateef (J)		4	202	120 min	101	0%
Mohammed Khursiwala (M)		7	202	90 min	134.6	0%
Totals:	0	11	404	210 min = 3.5 hrs	117.8	

Code Review Results (Tyler)

- Some methods had no usages
- Some constructor parameters did not have descriptive names
- Not enough comments
- Unused imports

Code Review Takeaways

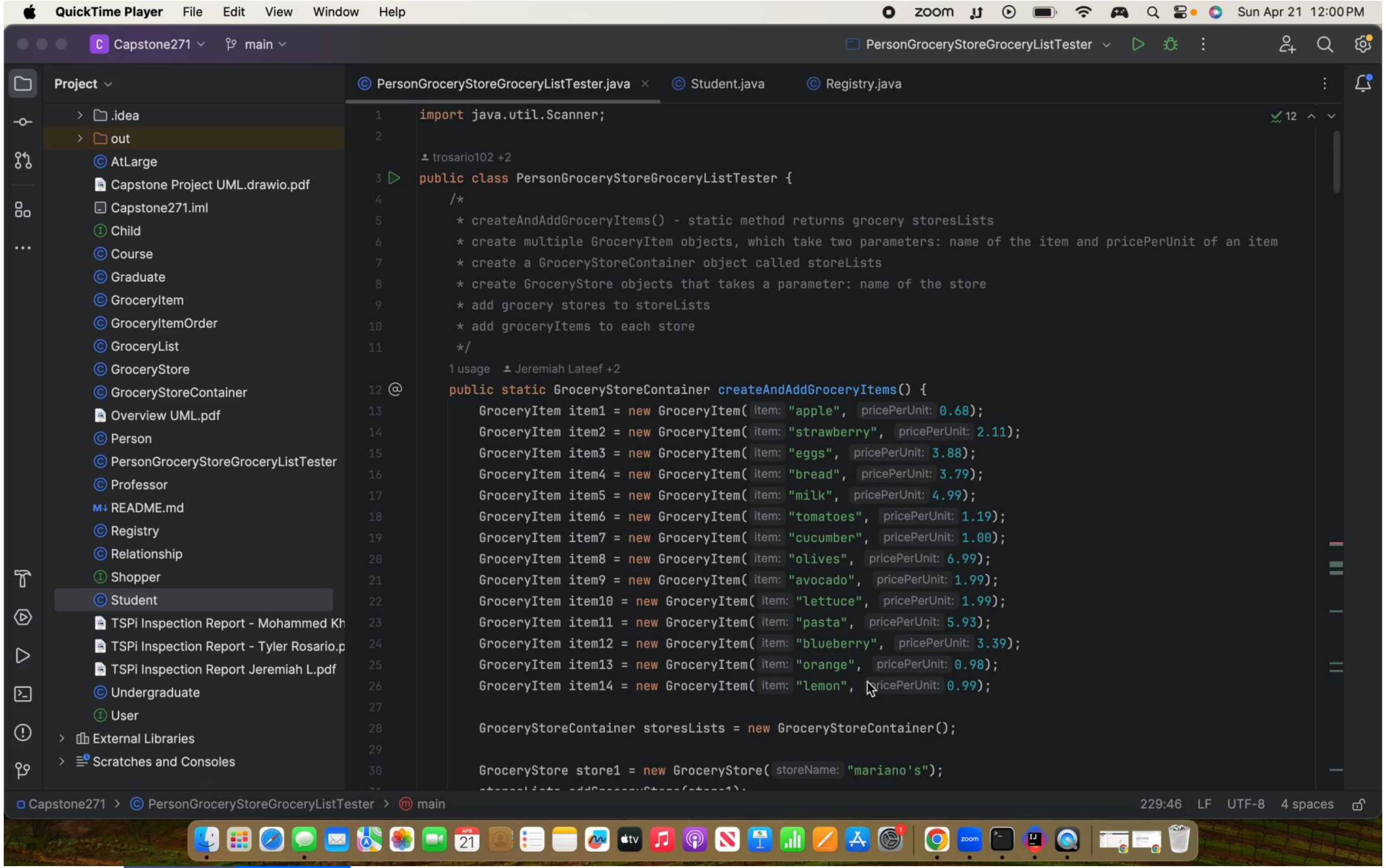
- See each other's coding style
- Different ways to implement code
- Learn exactly how different classes/methods work
- Allows for bug detection leading to higher quality code
- Standardize coding practices
- Team collaboration and cohesion

Takeaways

- Communicate early and often
- Establish naming conventions
- Design a unified UML to simplify coding process
- Use feature branches and review code before merging
- Test code individually and together early in the process

App Demo

Ancestry Demo 1



- .idea
- out
- AtLarge
- Capstone Project UML.drawio.pdf
- Capstone271.iml
- Child
- Course
- Graduate
- GroceryItem
- GroceryItemOrder
- GroceryList
- GroceryStore
- GroceryStoreContainer
- Overview UML.pdf
- Person
- PersonGroceryStoreGroceryListTester
- Professor
- README.md
- Registry
- Relationship
- Shopper
- Student
- TSPi Inspection Report - Mohammed Kh
- TSPi Inspection Report - Tyler Rosario.p
- TSPi Inspection Report Jeremiah L.pdf
- Undergraduate
- User
- External Libraries
- Scratches and Consoles

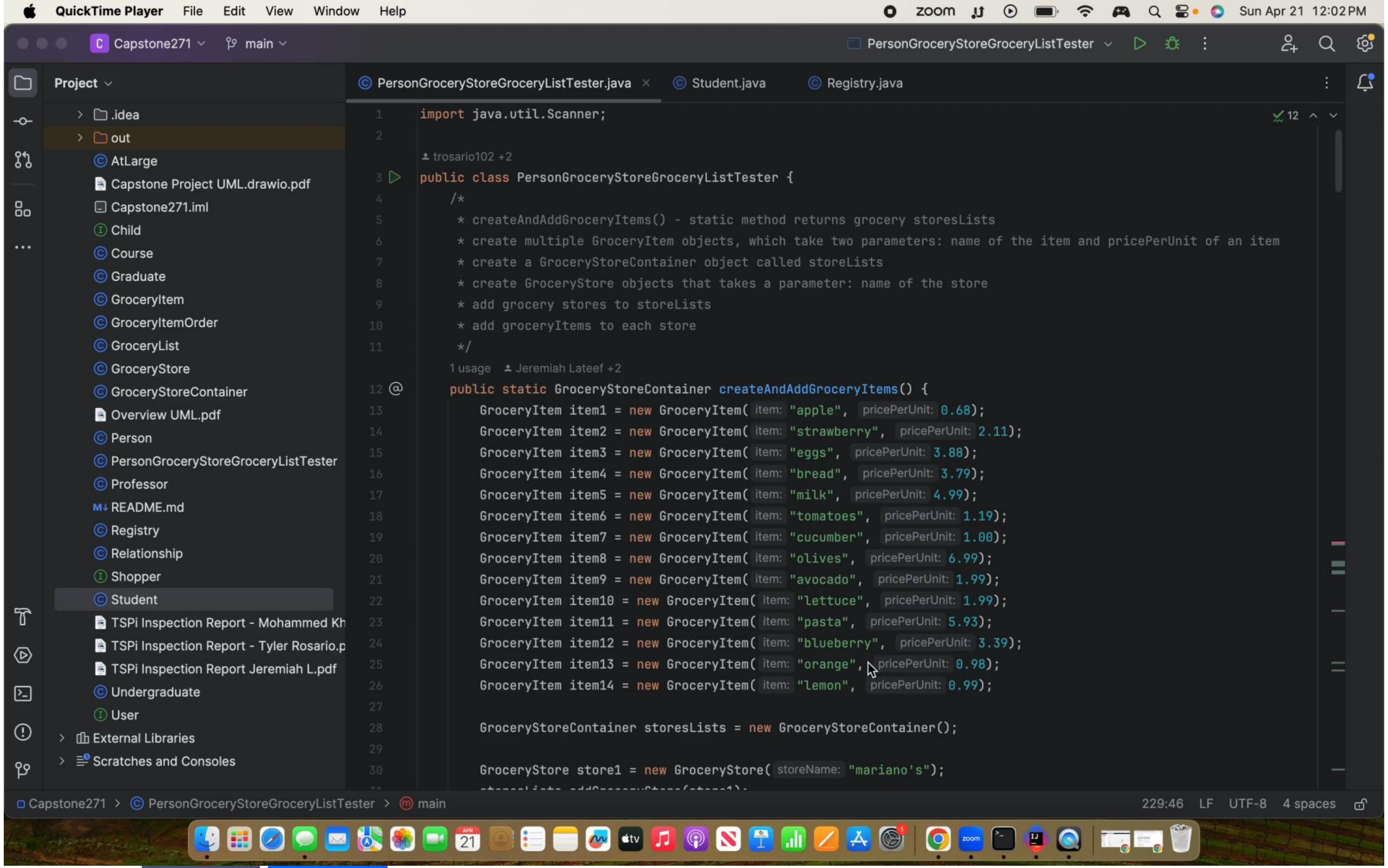
```
import java.util.Scanner;

trosario102 +2
public class PersonGroceryStoreGroceryListTester {
    /*
     * createAndAddGroceryItems() - static method returns grocery storesLists
     * create multiple GroceryItem objects, which take two parameters: name of the item and pricePerUnit of an item
     * create a GroceryStoreContainer object called storeLists
     * create GroceryStore objects that takes a parameter: name of the store
     * add grocery stores to storeLists
     * add groceryItems to each store
     */
    1 usage  Jeremiah Lateef +2
    public static GroceryStoreContainer createAndAddGroceryItems() {
        GroceryItem item1 = new GroceryItem( item: "apple",  pricePerUnit: 0.68);
        GroceryItem item2 = new GroceryItem( item: "strawberry",  pricePerUnit: 2.11);
        GroceryItem item3 = new GroceryItem( item: "eggs",  pricePerUnit: 3.88);
        GroceryItem item4 = new GroceryItem( item: "bread",  pricePerUnit: 3.79);
        GroceryItem item5 = new GroceryItem( item: "milk",  pricePerUnit: 4.99);
        GroceryItem item6 = new GroceryItem( item: "tomatoes",  pricePerUnit: 1.19);
        GroceryItem item7 = new GroceryItem( item: "cucumber",  pricePerUnit: 1.00);
        GroceryItem item8 = new GroceryItem( item: "olives",  pricePerUnit: 6.99);
        GroceryItem item9 = new GroceryItem( item: "avocado",  pricePerUnit: 1.99);
        GroceryItem item10 = new GroceryItem( item: "lettuce",  pricePerUnit: 1.99);
        GroceryItem item11 = new GroceryItem( item: "pasta",  pricePerUnit: 5.93);
        GroceryItem item12 = new GroceryItem( item: "blueberry",  pricePerUnit: 3.39);
        GroceryItem item13 = new GroceryItem( item: "orange",  pricePerUnit: 0.98);
        GroceryItem item14 = new GroceryItem( item: "lemon",  pricePerUnit: 0.99);

        GroceryStoreContainer storesLists = new GroceryStoreContainer();

        GroceryStore store1 = new GroceryStore( storeName: "mariano's");
```

Ancestry Demo 2



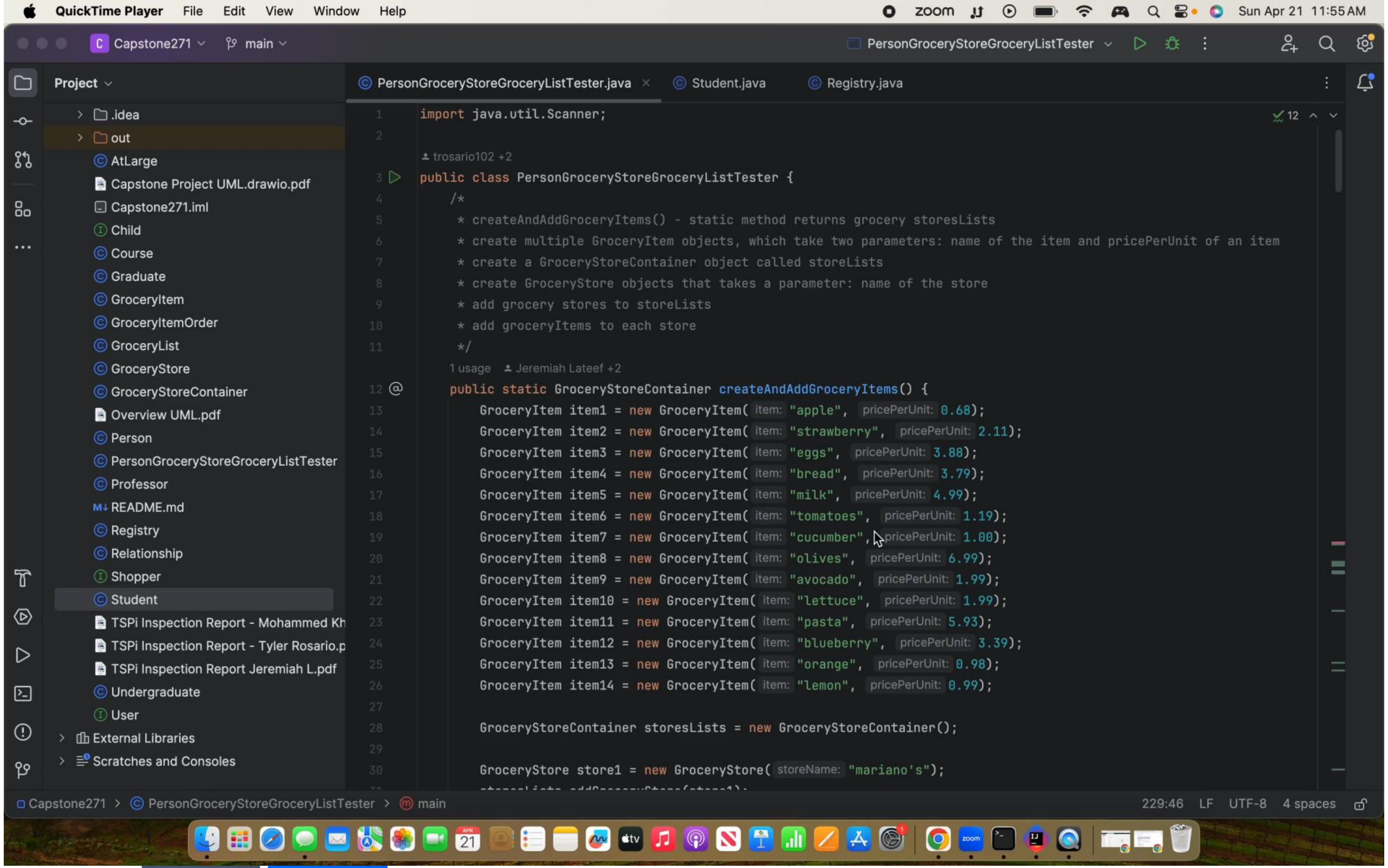
```java
import java.util.Scanner;

trosario102 +2
public class PersonGroceryStoreGroceryListTester {
    /*
     * createAndAddGroceryItems() - static method returns grocery storesLists
     * create multiple GroceryItem objects, which take two parameters: name of the item and pricePerUnit of an item
     * create a GroceryStoreContainer object called storeLists
     * create GroceryStore objects that takes a parameter: name of the store
     * add grocery stores to storeLists
     * add groceryItems to each store
     */
    1 usage  Jeremiah Lateef +2
    public static GroceryStoreContainer createAndAddGroceryItems() {
        GroceryItem item1 = new GroceryItem( item: "apple",  pricePerUnit: 0.68);
        GroceryItem item2 = new GroceryItem( item: "strawberry",  pricePerUnit: 2.11);
        GroceryItem item3 = new GroceryItem( item: "eggs",  pricePerUnit: 3.88);
        GroceryItem item4 = new GroceryItem( item: "bread",  pricePerUnit: 3.79);
        GroceryItem item5 = new GroceryItem( item: "milk",  pricePerUnit: 4.99);
        GroceryItem item6 = new GroceryItem( item: "tomatoes",  pricePerUnit: 1.19);
        GroceryItem item7 = new GroceryItem( item: "cucumber",  pricePerUnit: 1.00);
        GroceryItem item8 = new GroceryItem( item: "olives",  pricePerUnit: 6.99);
        GroceryItem item9 = new GroceryItem( item: "avocado",  pricePerUnit: 1.99);
        GroceryItem item10 = new GroceryItem( item: "lettuce",  pricePerUnit: 1.99);
        GroceryItem item11 = new GroceryItem( item: "pasta",  pricePerUnit: 5.93);
        GroceryItem item12 = new GroceryItem( item: "blueberry",  pricePerUnit: 3.39);
        GroceryItem item13 = new GroceryItem( item: "orange",  pricePerUnit: 0.98);
        GroceryItem item14 = new GroceryItem( item: "lemon",  pricePerUnit: 0.99);

        GroceryStoreContainer storesLists = new GroceryStoreContainer();

        GroceryStore store1 = new GroceryStore( storeName: "mariano's");
```
)

Add/Drop Course



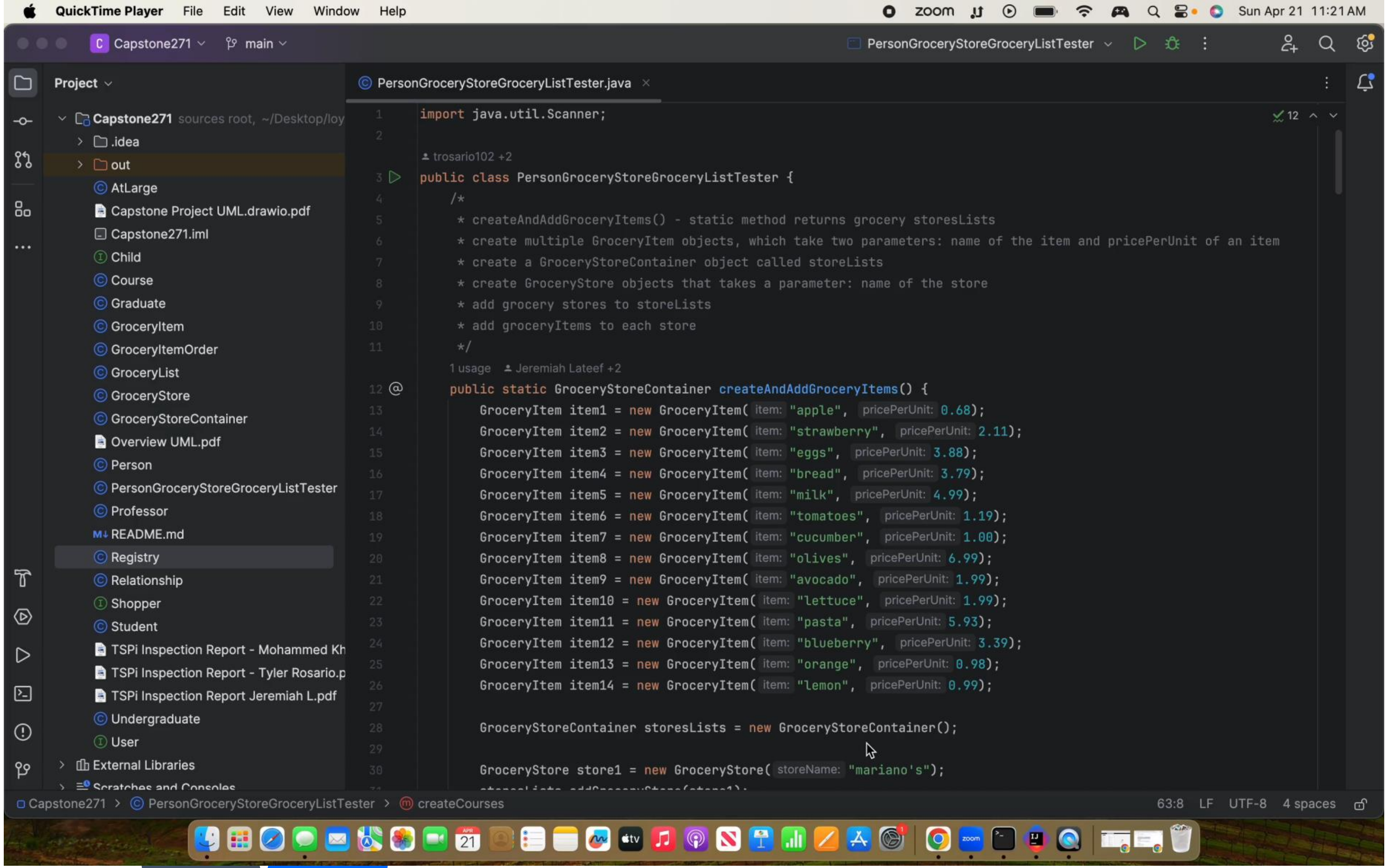
```java
import java.util.Scanner;

trosario102 +2
public class PersonGroceryStoreGroceryListTester {
    /*
     * createAndAddGroceryItems() - static method returns grocery storesLists
     * create multiple GroceryItem objects, which take two parameters: name of the item and pricePerUnit of an item
     * create a GroceryStoreContainer object called storeLists
     * create GroceryStore objects that takes a parameter: name of the store
     * add grocery stores to storeLists
     * add groceryItems to each store
     */
    1 usage  Jeremiah Lateef +2
    public static GroceryStoreContainer createAndAddGroceryItems() {
        GroceryItem item1 = new GroceryItem( item: "apple",  pricePerUnit: 0.68);
        GroceryItem item2 = new GroceryItem( item: "strawberry",  pricePerUnit: 2.11);
        GroceryItem item3 = new GroceryItem( item: "eggs",  pricePerUnit: 3.88);
        GroceryItem item4 = new GroceryItem( item: "bread",  pricePerUnit: 3.79);
        GroceryItem item5 = new GroceryItem( item: "milk",  pricePerUnit: 4.99);
        GroceryItem item6 = new GroceryItem( item: "tomatoes",  pricePerUnit: 1.19);
        GroceryItem item7 = new GroceryItem( item: "cucumber",  pricePerUnit: 1.00);
        GroceryItem item8 = new GroceryItem( item: "olives",  pricePerUnit: 6.99);
        GroceryItem item9 = new GroceryItem( item: "avocado",  pricePerUnit: 1.99);
        GroceryItem item10 = new GroceryItem( item: "lettuce",  pricePerUnit: 1.99);
        GroceryItem item11 = new GroceryItem( item: "pasta",  pricePerUnit: 5.93);
        GroceryItem item12 = new GroceryItem( item: "blueberry",  pricePerUnit: 3.39);
        GroceryItem item13 = new GroceryItem( item: "orange",  pricePerUnit: 0.98);
        GroceryItem item14 = new GroceryItem( item: "lemon",  pricePerUnit: 0.99);

        GroceryStoreContainer storesLists = new GroceryStoreContainer();

        GroceryStore store1 = new GroceryStore( storeName: "mariano's");
```

Grocery Demo



QuickTime Player File Edit View Window Help
Sun Apr 21 11:21 AM
Capstone271
main
PersonGroceryStoreGroceryListTester
Project
Capstone271 sources root, ~/Desktop/loy
.idea
out
AtLarge
Capstone Project UML.drawio.pdf
Capstone271.iml
Child
Course
Graduate
GroceryItem
GroceryItemOrder
GroceryList
GroceryStore
GroceryStoreContainer
Overview UML.pdf
Person
PersonGroceryStoreGroceryListTester
Professor
README.md
Registry
Relationship
Shopper
Student
TSPi Inspection Report - Mohammed Kh
TSPi Inspection Report - Tyler Rosario.p
TSPi Inspection Report Jeremiah L.pdf
Undergraduate
User
External Libraries
Scratches and Consoles
PersonGroceryStoreGroceryListTester.java
trosario102 +2
1 usage
Jeremiah Lateef +2
Capstone271 > PersonGroceryStoreGroceryListTester > createCourses
63:8 LF UTF-8 4 spaces

```java
import java.util.Scanner;

public class PersonGroceryStoreGroceryListTester {
    /*
     * createAndAddGroceryItems() - static method returns grocery storesLists
     * create multiple GroceryItem objects, which take two parameters: name of the item and pricePerUnit of an item
     * create a GroceryStoreContainer object called storeLists
     * create GroceryStore objects that takes a parameter: name of the store
     * add grocery stores to storeLists
     * add groceryItems to each store
     */
    public static GroceryStoreContainer createAndAddGroceryItems() {
        GroceryItem item1 = new GroceryItem( item: "apple",  pricePerUnit: 0.68);
        GroceryItem item2 = new GroceryItem( item: "strawberry",  pricePerUnit: 2.11);
        GroceryItem item3 = new GroceryItem( item: "eggs",  pricePerUnit: 3.88);
        GroceryItem item4 = new GroceryItem( item: "bread",  pricePerUnit: 3.79);
        GroceryItem item5 = new GroceryItem( item: "milk",  pricePerUnit: 4.99);
        GroceryItem item6 = new GroceryItem( item: "tomatoes",  pricePerUnit: 1.19);
        GroceryItem item7 = new GroceryItem( item: "cucumber",  pricePerUnit: 1.00);
        GroceryItem item8 = new GroceryItem( item: "olives",  pricePerUnit: 6.99);
        GroceryItem item9 = new GroceryItem( item: "avocado",  pricePerUnit: 1.99);
        GroceryItem item10 = new GroceryItem( item: "lettuce",  pricePerUnit: 1.99);
        GroceryItem item11 = new GroceryItem( item: "pasta",  pricePerUnit: 5.93);
        GroceryItem item12 = new GroceryItem( item: "blueberry",  pricePerUnit: 3.39);
        GroceryItem item13 = new GroceryItem( item: "orange",  pricePerUnit: 0.98);
        GroceryItem item14 = new GroceryItem( item: "lemon",  pricePerUnit: 0.99);

        GroceryStoreContainer storesLists = new GroceryStoreContainer();

        GroceryStore store1 = new GroceryStore( storeName: "mariano's");
```

Thank you for listening!

Any questions?