IMS-STARTER – WEEK 5 EXERCISE

ELINA MCGLINCHEY – 22APRENABLE3

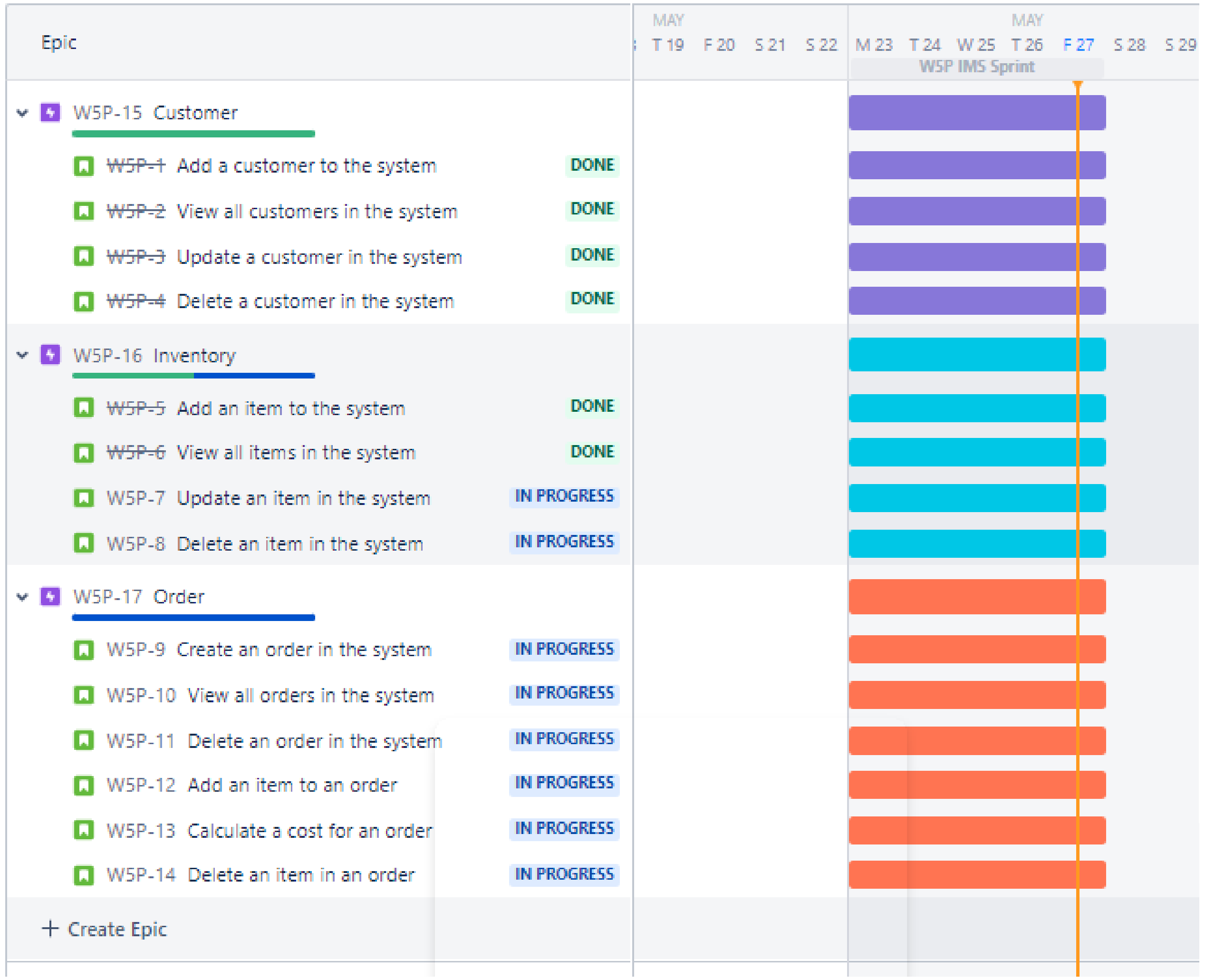
WEEK TIMETABLE

	DAY 1	DAY 2	DAY 3	DAY 4
STEP 1	Read through task and made notes (am)	Started reading through and assessing IMS-Starter code (am)	Copied and pasted missing files back from Github to Eclipse (am)	Copied IMS-Starter code in file explorer to keep safe before pushing to Github (am)
STEP 2	Forked project (am)	Began making changes to code – username and password (am)	Began DAO and Controller tests (am)	Pushed to Github but experienced issues – debugged connection issues (am)
STEP 3	Created Jira (am)	Added Item, Order and OrderItems (am)	Pushed to Github and files disappeared, needed to retrieve (am)	Created imstest database (am)
STEP 4	Created EER Diagram (am)	Continued to push up to Github – trouble (am)	Began testing Customer and Item DAOTests and ControllerTests – failures (am)	Ran further tests and debugged (am and pm)
STEP 5	Created Risk Assessment (am)	Debugging all afternoon and retrieving missing code after git push (pm)	Pushed to Github – lost files and needed to retrieve (pm)	Created Order DAOTest and ControllerTest – got failures 20/33 (pm)
STEP 6	Set up IMS database SQL (pm)		Debugging tests in code (pm)	Debugging for better connection – databases not found, app partially working (pm)
STEP 7	Create tables and set relationships (pm)			Merge to main (NEED TO DO)

JIRA Management

https://elinamcglinchey.atlassian.net/jira/software/projects/W5P/boards/1/roadmap?timeline=WEEKS

https://github.com/elinamcglinchey/IMS-Starter



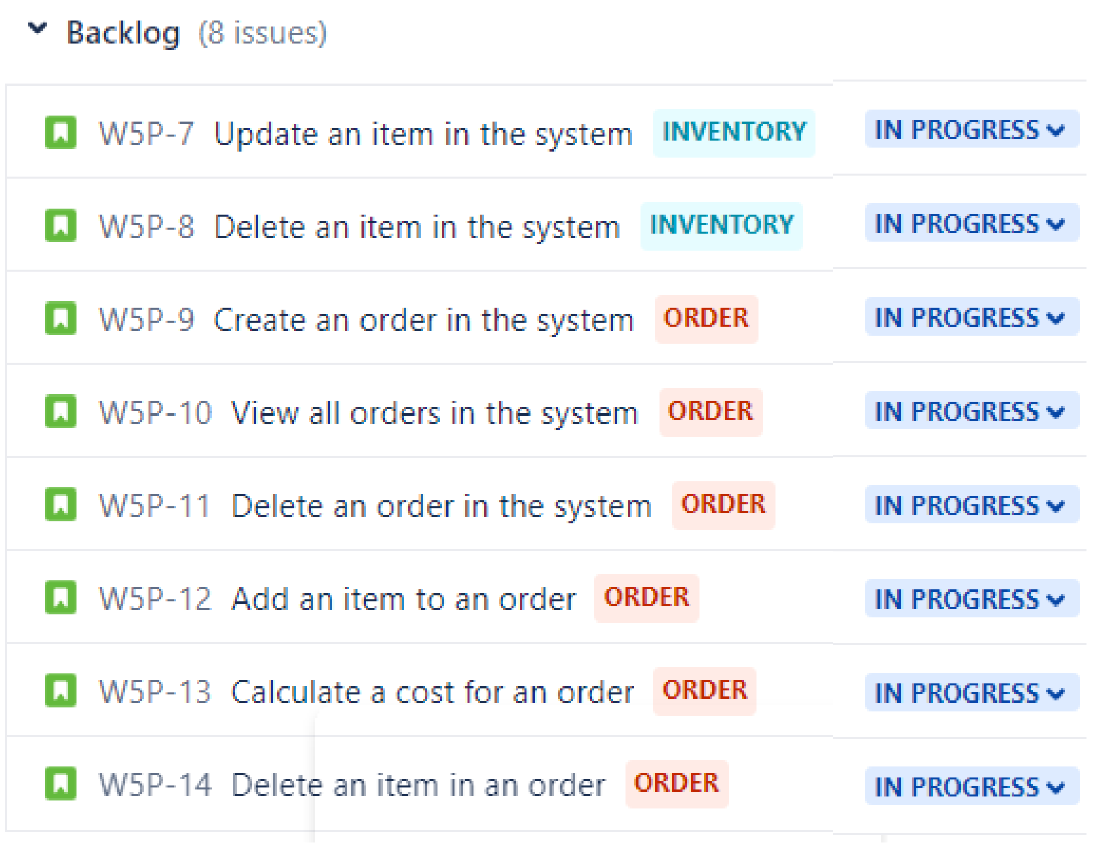
Epic

MAY T 19 F 20 S 21 S 22 | MAY M 23 T 24 W 25 T 26 F 27 S 28 S 29

W5P IMS Sprint

Epic	Status
W5P-15 Customer	
~~W5P-1~~ Add a customer to the system	DONE
~~W5P-2~~ View all customers in the system	DONE
~~W5P-3~~ Update a customer in the system	DONE
~~W5P-4~~ Delete a customer in the system	DONE
W5P-16 Inventory	
~~W5P-5~~ Add an item to the system	DONE
~~W5P-6~~ View all items in the system	DONE
W5P-7 Update an item in the system	IN PROGRESS
W5P-8 Delete an item in the system	IN PROGRESS
W5P-17 Order	
W5P-9 Create an order in the system	IN PROGRESS
W5P-10 View all orders in the system	IN PROGRESS
W5P-11 Delete an order in the system	IN PROGRESS
W5P-12 Add an item to an order	IN PROGRESS
W5P-13 Calculate a cost for an order	IN PROGRESS
W5P-14 Delete an item in an order	IN PROGRESS

+ Create Epic



Backlog (8 issues)

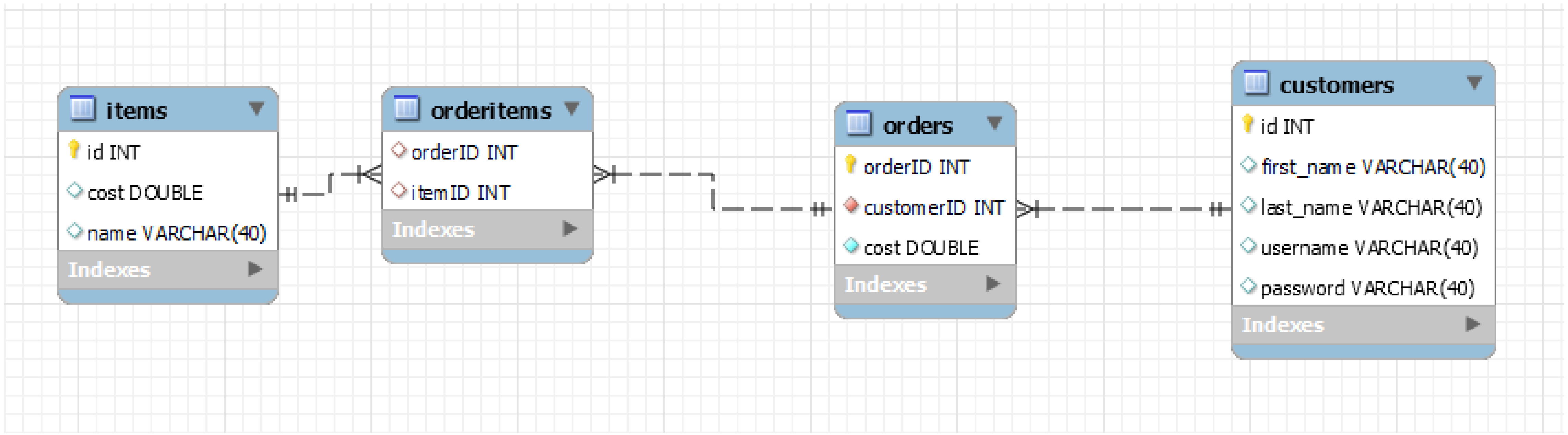
Issue	Summary	Epic	Status
W5P-7	Update an item in the system	INVENTORY	IN PROGRESS
W5P-8	Delete an item in the system	INVENTORY	IN PROGRESS
W5P-9	Create an order in the system	ORDER	IN PROGRESS
W5P-10	View all orders in the system	ORDER	IN PROGRESS
W5P-11	Delete an order in the system	ORDER	IN PROGRESS
W5P-12	Add an item to an order	ORDER	IN PROGRESS
W5P-13	Calculate a cost for an order	ORDER	IN PROGRESS
W5P-14	Delete an item in an order	ORDER	IN PROGRESS

What went wrong?

1.Time 2.Technical Issues 3. Pulling to early on Github branches

EER DIAGRAM

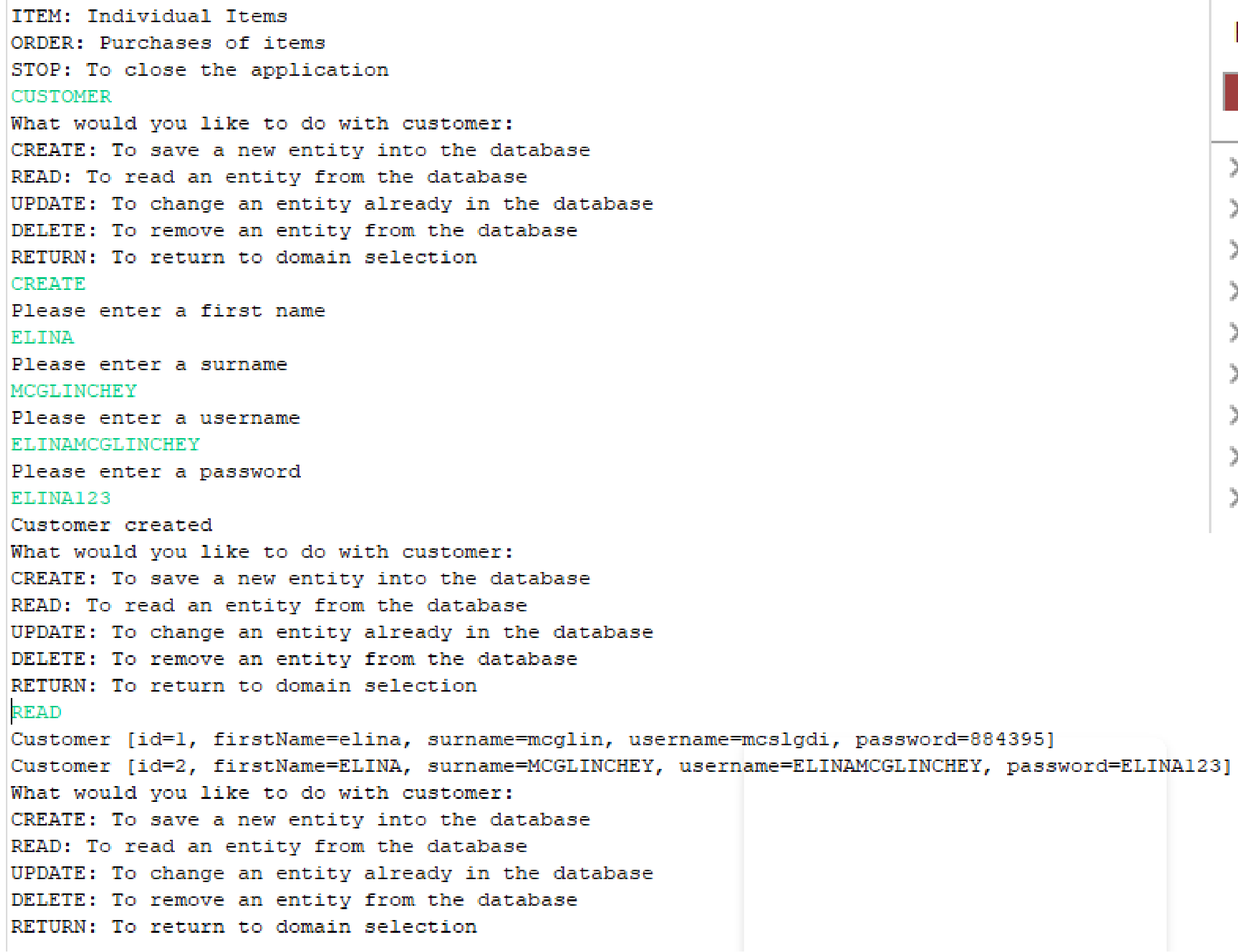
Diagram 3 – Day 5



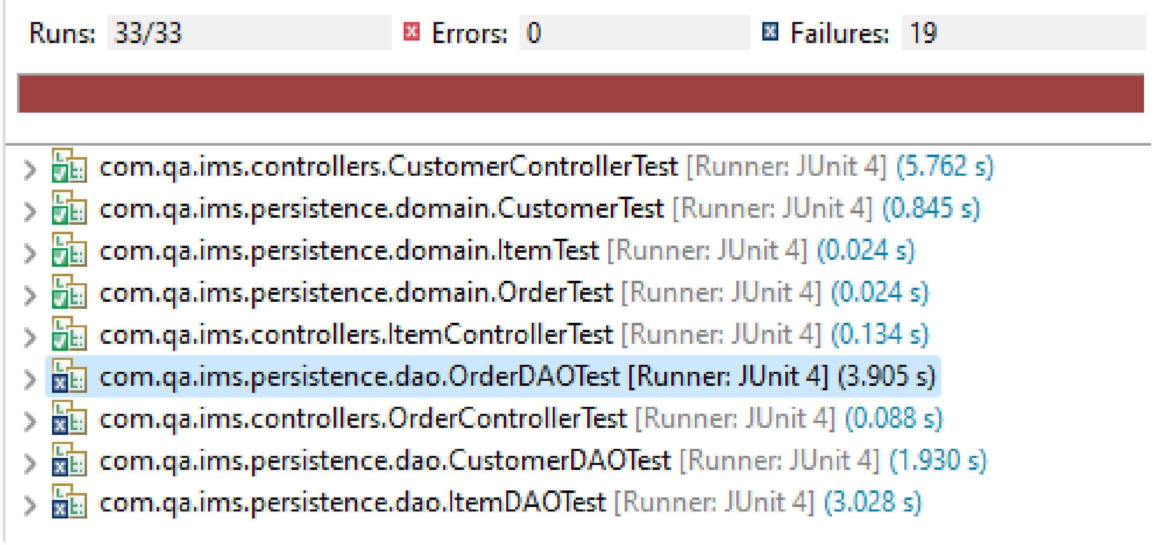
- Added OrderItems back in
- Organised database better
- Corrected errors in primary and foreign keys

CODEBASE

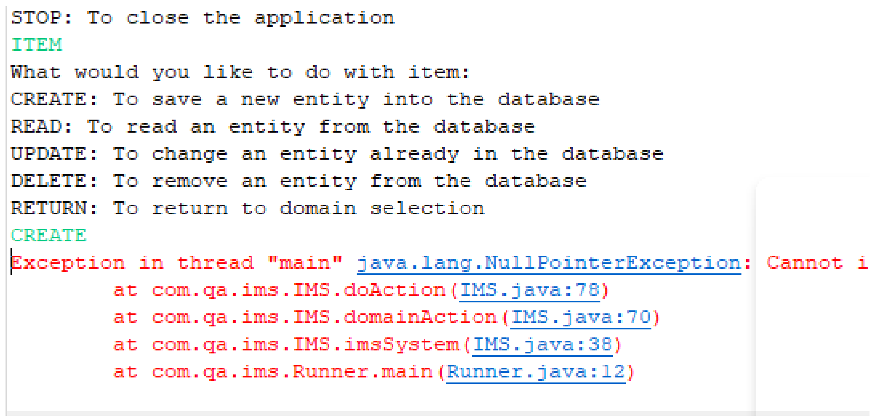
- Partially working app
- Zero errors in tests yet only partially passing
- Demo



```
ITEM: Individual Items
ORDER: Purchases of items
STOP: To close the application
CUSTOMER
What would you like to do with customer:
CREATE: To save a new entity into the database
READ: To read an entity from the database
UPDATE: To change an entity already in the database
DELETE: To remove an entity from the database
RETURN: To return to domain selection
CREATE
Please enter a first name
ELINA
Please enter a surname
MCGLINCHEY
Please enter a username
ELINAMCGLINCHEY
Please enter a password
ELINA123
Customer created
What would you like to do with customer:
CREATE: To save a new entity into the database
READ: To read an entity from the database
UPDATE: To change an entity already in the database
DELETE: To remove an entity from the database
RETURN: To return to domain selection
READ
Customer [id=1, firstName=elina, surname=mcglin, username=mcslgdi, password=884395]
Customer [id=2, firstName=ELINA, surname=MCGLINCHEY, username=ELINAMCGLINCHEY, password=ELINA123]
What would you like to do with customer:
CREATE: To save a new entity into the database
READ: To read an entity from the database
UPDATE: To change an entity already in the database
DELETE: To remove an entity from the database
RETURN: To return to domain selection
```



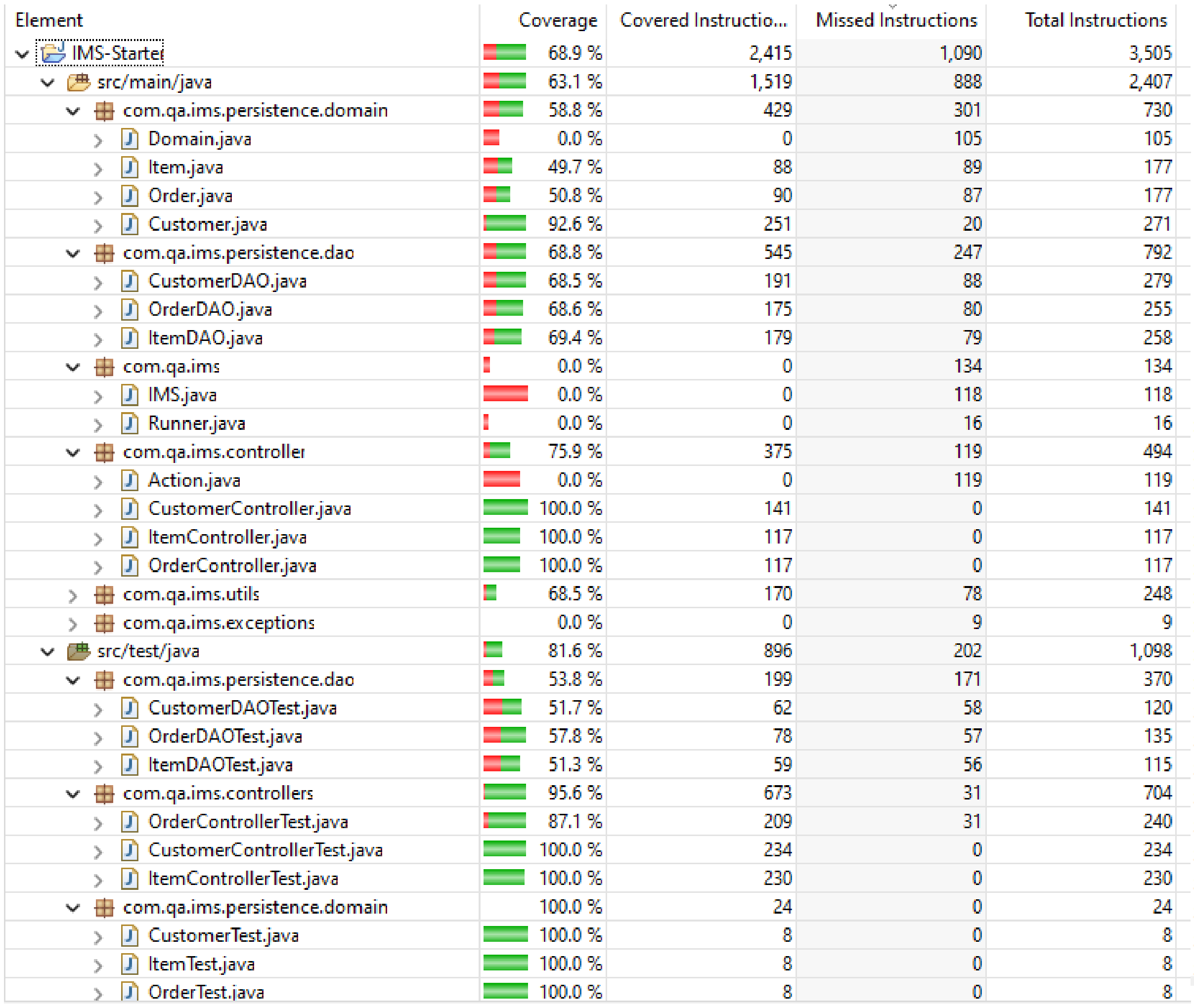
```
Runs: 33/33    Errors: 0    Failures: 19

com.qa.ims.controllers.CustomerControllerTest [Runner: JUnit 4] (5.762 s)
com.qa.ims.persistence.domain.CustomerTest [Runner: JUnit 4] (0.845 s)
com.qa.ims.persistence.domain.ItemTest [Runner: JUnit 4] (0.024 s)
com.qa.ims.persistence.domain.OrderTest [Runner: JUnit 4] (0.024 s)
com.qa.ims.controllers.ItemControllerTest [Runner: JUnit 4] (0.134 s)
com.qa.ims.persistence.dao.OrderDAOTest [Runner: JUnit 4] (3.905 s)
com.qa.ims.controllers.OrderControllerTest [Runner: JUnit 4] (0.088 s)
com.qa.ims.persistence.dao.CustomerDAOTest [Runner: JUnit 4] (1.930 s)
com.qa.ims.persistence.dao.ItemDAOTest [Runner: JUnit 4] (3.028 s)
```



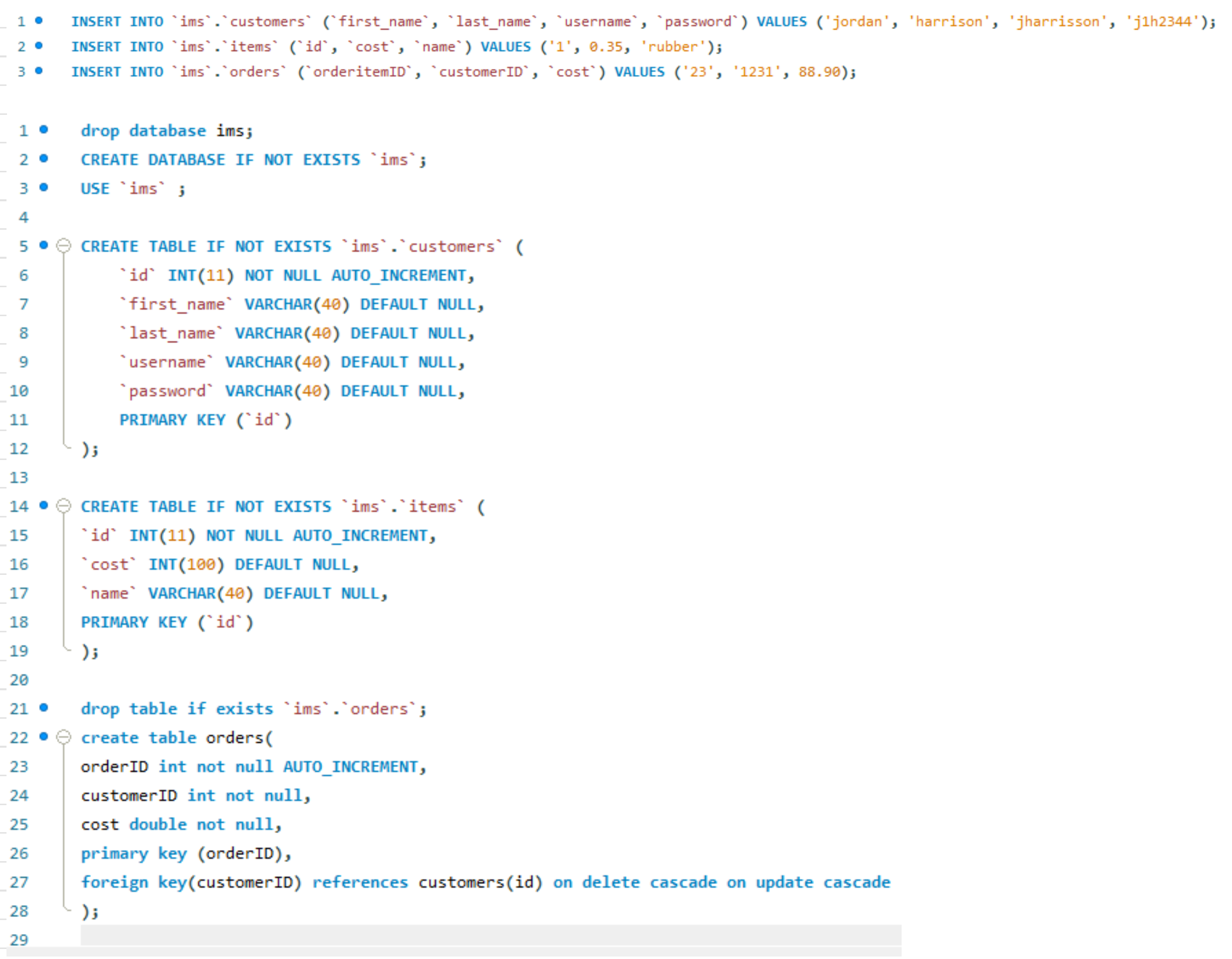
```
STOP: To close the application
ITEM
What would you like to do with item:
CREATE: To save a new entity into the database
READ: To read an entity from the database
UPDATE: To change an entity already in the database
DELETE: To remove an entity from the database
RETURN: To return to domain selection
CREATE
Exception in thread "main" java.lang.NullPointerException: Cannot i
        at com.qa.ims.IMS.doAction(IMS.java:78)
        at com.qa.ims.IMS.domainAction(IMS.java:70)
        at com.qa.ims.IMS.imsSystem(IMS.java:38)
        at com.qa.ims.Runner.main(Runner.java:12)
```

CODEBASE

- Partially working app
- https://github.com/elinamcglinchey/IMS-Starter
- Summary



Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
IMS-Starter	68.9 %	2,415	1,090	3,505
src/main/java	63.1 %	1,519	888	2,407
com.qa.ims.persistence.domain	58.8 %	429	301	730
Domain.java	0.0 %	0	105	105
Item.java	49.7 %	88	89	177
Order.java	50.8 %	90	87	177
Customer.java	92.6 %	251	20	271
com.qa.ims.persistence.dao	68.8 %	545	247	792
CustomerDAO.java	68.5 %	191	88	279
OrderDAO.java	68.6 %	175	80	255
ItemDAO.java	69.4 %	179	79	258
com.qa.ims	0.0 %	0	134	134
IMS.java	0.0 %	0	118	118
Runner.java	0.0 %	0	16	16
com.qa.ims.controller	75.9 %	375	119	494
Action.java	0.0 %	0	119	119
CustomerController.java	100.0 %	141	0	141
ItemController.java	100.0 %	117	0	117
OrderController.java	100.0 %	117	0	117
com.qa.ims.utils	68.5 %	170	78	248
com.qa.ims.exceptions	0.0 %	0	9	9
src/test/java	81.6 %	896	202	1,098
com.qa.ims.persistence.dao	53.8 %	199	171	370
CustomerDAOTest.java	51.7 %	62	58	120
OrderDAOTest.java	57.8 %	78	57	135
ItemDAOTest.java	51.3 %	59	56	115
com.qa.ims.controllers	95.6 %	673	31	704
OrderControllerTest.java	87.1 %	209	31	240
CustomerControllerTest.java	100.0 %	234	0	234
ItemControllerTest.java	100.0 %	230	0	230
com.qa.ims.persistence.domain	100.0 %	24	0	24
CustomerTest.java	100.0 %	8	0	8
ItemTest.java	100.0 %	8	0	8
OrderTest.java	100.0 %	8	0	8

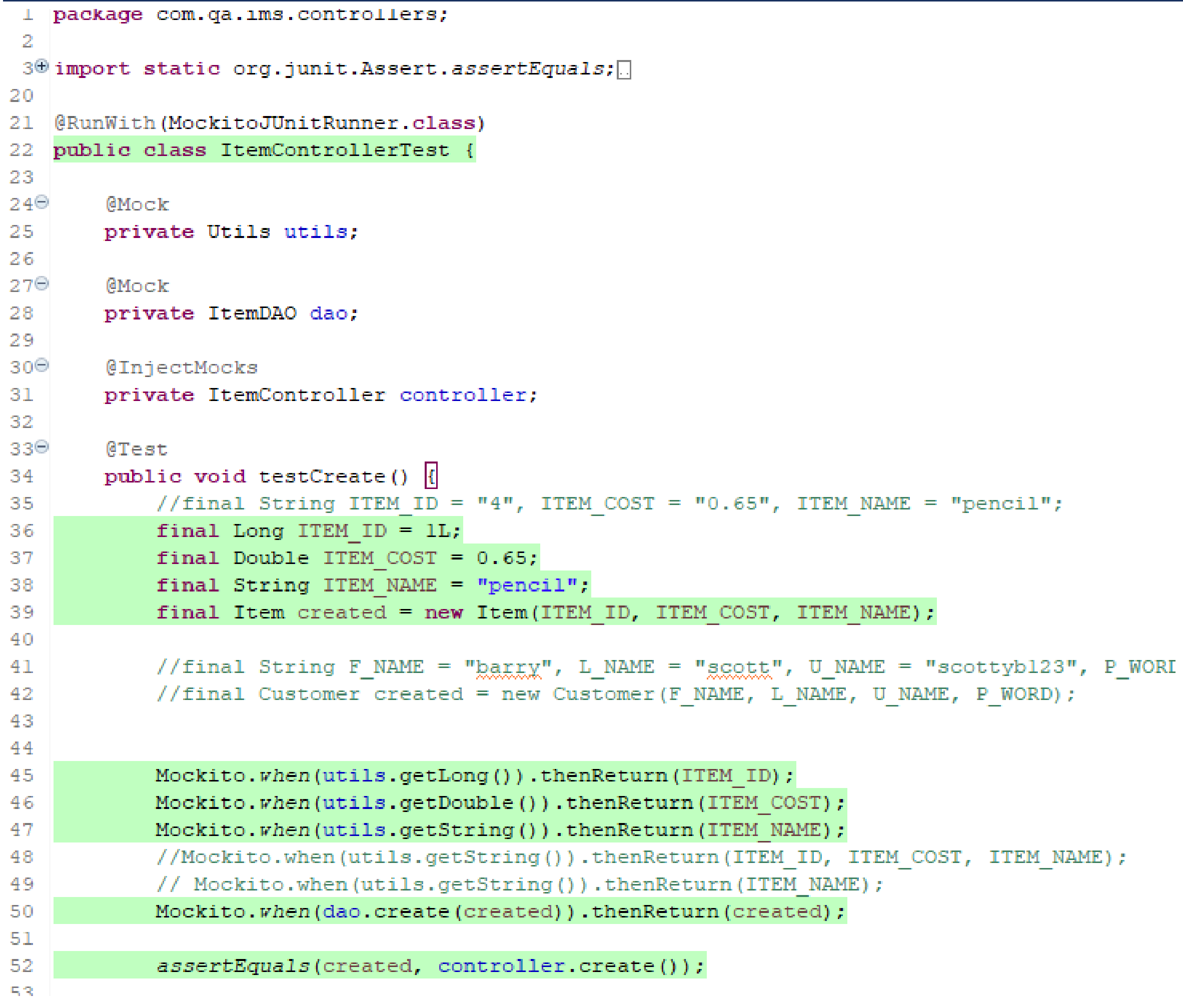


```sql
INSERT INTO `ims`.`customers` (`first_name`, `last_name`, `username`, `password`) VALUES ('jordan', 'harrison', 'jharrisson', 'j1h2344');
INSERT INTO `ims`.`items` (`id`, `cost`, `name`) VALUES ('1', 0.35, 'rubber');
INSERT INTO `ims`.`orders` (`orderitemID`, `customerID`, `cost`) VALUES ('23', '1231', 88.90);
```

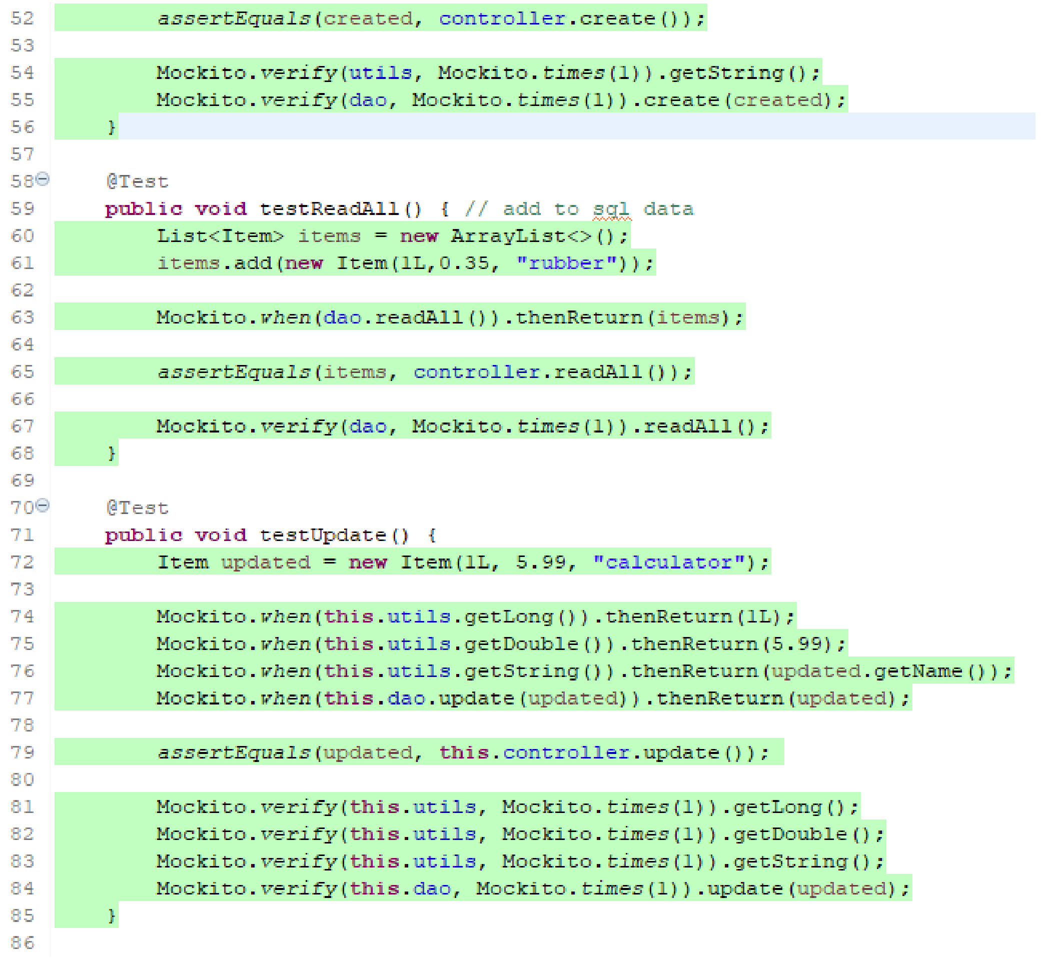
```sql
drop database ims;
CREATE DATABASE IF NOT EXISTS `ims`;
USE `ims` ;

CREATE TABLE IF NOT EXISTS `ims`.`customers` (
    `id` INT(11) NOT NULL AUTO_INCREMENT,
    `first_name` VARCHAR(40) DEFAULT NULL,
    `last_name` VARCHAR(40) DEFAULT NULL,
    `username` VARCHAR(40) DEFAULT NULL,
    `password` VARCHAR(40) DEFAULT NULL,
    PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `ims`.`items` (
`id` INT(11) NOT NULL AUTO_INCREMENT,
`cost` INT(100) DEFAULT NULL,
`name` VARCHAR(40) DEFAULT NULL,
PRIMARY KEY (`id`)
);

drop table if exists `ims`.`orders`;
create table orders(
orderID int not null AUTO_INCREMENT,
customerID int not null,
cost double not null,
primary key (orderID),
foreign key(customerID) references customers(id) on delete cascade on update cascade
);
```

CODEBASE

- Partially working app
- Zero errors in tests yet only partially passing
- Summary



```java
package com.qa.ims.controllers;

import static org.junit.Assert.assertEquals;

@RunWith(MockitoJUnitRunner.class)
public class ItemControllerTest {

    @Mock
    private Utils utils;

    @Mock
    private ItemDAO dao;

    @InjectMocks
    private ItemController controller;

    @Test
    public void testCreate() {
        //final String ITEM_ID = "4", ITEM_COST = "0.65", ITEM_NAME = "pencil";
        final Long ITEM_ID = 1L;
        final Double ITEM_COST = 0.65;
        final String ITEM_NAME = "pencil";
        final Item created = new Item(ITEM_ID, ITEM_COST, ITEM_NAME);

        //final String F_NAME = "barry", L_NAME = "scott", U_NAME = "scottyb123", P_WOR
        //final Customer created = new Customer(F_NAME, L_NAME, U_NAME, P_WORD);


        Mockito.when(utils.getLong()).thenReturn(ITEM_ID);
        Mockito.when(utils.getDouble()).thenReturn(ITEM_COST);
        Mockito.when(utils.getString()).thenReturn(ITEM_NAME);
        //Mockito.when(utils.getString()).thenReturn(ITEM_ID, ITEM_COST, ITEM_NAME);
        // Mockito.when(utils.getString()).thenReturn(ITEM_NAME);
        Mockito.when(dao.create(created)).thenReturn(created);

        assertEquals(created, controller.create());
```



```java
        assertEquals(created, controller.create());

        Mockito.verify(utils, Mockito.times(1)).getString();
        Mockito.verify(dao, Mockito.times(1)).create(created);
    }

    @Test
    public void testReadAll() { // add to sql data
        List<Item> items = new ArrayList<>();
        items.add(new Item(1L,0.35, "rubber"));

        Mockito.when(dao.readAll()).thenReturn(items);

        assertEquals(items, controller.readAll());

        Mockito.verify(dao, Mockito.times(1)).readAll();
    }

    @Test
    public void testUpdate() {
        Item updated = new Item(1L, 5.99, "calculator");

        Mockito.when(this.utils.getLong()).thenReturn(1L);
        Mockito.when(this.utils.getDouble()).thenReturn(5.99);
        Mockito.when(this.utils.getString()).thenReturn(updated.getName());
        Mockito.when(this.dao.update(updated)).thenReturn(updated);

        assertEquals(updated, this.controller.update());

        Mockito.verify(this.utils, Mockito.times(1)).getLong();
        Mockito.verify(this.utils, Mockito.times(1)).getDouble();
        Mockito.verify(this.utils, Mockito.times(1)).getString();
        Mockito.verify(this.dao, Mockito.times(1)).update(updated);
    }

```

RISK ASSESSMENT

- Risk Matrix

Risks	Negligible	Minor	Major	Critical	Catastrophic
Time	Completed all tasks	Unable to do stretch goals	No time to do any successful application functions	No time to debug errors in tests	No time to do any necessary classes
Injury/Illness	Completely healthy	Cold, slightly anxious or stress	Flu/Stomach bug/Food Poisoning, anxiety	Miss majority of week due to injury or illness	Injury or illness – missing assessed week
Theft	No theft	Theft of other possessions related to laptop/pc	Theft of appliances necessary for task (mouse/keyboard)	Theft of laptop or pc	Theft of laptop or pc with no external drive or cloud upload
Software Corruption	No Corruption	Github/Gitbash failure	Complex version history	Files unable to load or lost	Files completed deleted
Github pushes (i.e pushed too early)	No github issues	Pull request on Github too early	Didn't separate branches	Merged to main too early	Corrupted main
Missed commit	No missed commits	Some missed commits	Big commits missed	Most commits missed	All commits missed
Poor implementation (lack of experience)	Complete understanding	Basic debugging issues	Lack of eclipse understanding	Barely have knowledge of any softwares	No understanding of any softwares

REFLECTION

- What went well?
- What did not go well?
- What can I improve for future projects?

Did go well	Did not go well
Worked independently to debug	Needed help on simple syntax errors missed ({ ;)) etc
Collaborated with peers in a team – helping each other	Failures in many tests
Extended Java, SQL and Git understanding	Occasionally needed help in debugging
Got some tests working with no prior Java knowledge	Git uploads

What can I improve for future projects?

- Dissect complete code and make pen and paper notes before beginning
- Revise Github, SQL and Gitbash skills (had forgotten many since not using since Week 1 or 2)
- Work more into the evenings
- Take small interval breaks to remove self from the screen during intense projects
- Practice making branches on Git
- Ask for more help in the beginning to avoid unnecessary loopholes
- Practice SQL, Git, Github and Java skills – look back on previous videos to remind and revise skills learned
- Make more notes

NEXT STEPS

What is left do to in this Sprint/Project?

- Finish off testing (attempt to get more successful tests)
- Debug DAO issues in SQL
- Run MVN clean package if I finish tests
- Ensure jar file with dependencies is in main repo NOT target
- Push to Github and merge dev to main
- Create a README

THANK YOU

ELINAMCGLINCHEY1@GMAIL.COM