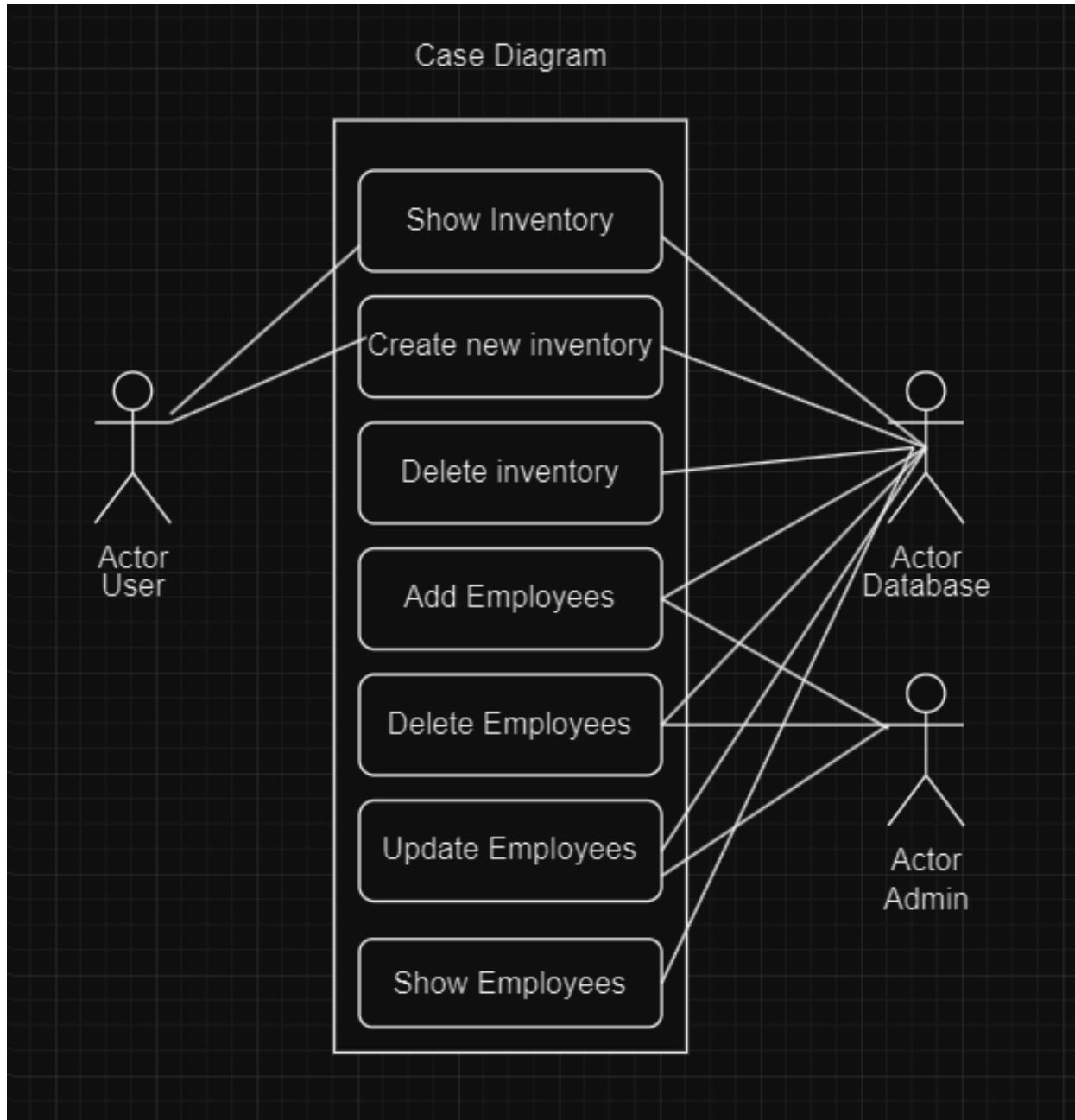


PAST ITERATION 1

Case Diagram/ Iteration 1 Case Diagram



Updated Use Case Scenarios for Iteration 3

Scenario 1

Use case: User Views Inventory

Actors: User, Database

Goal: User views the inventory

	Actor/User's Action		System's Response
		1	System prompts user with login screen
2	User inputs username and password and clicks Sign In	3	System checks if the credentials match/verify and takes user to the home page
4	User selects Navigate option at top left corner	5	System shows various options to the user
6	User selects Inventory	7	System shows Lego Inventory DBMS with Set Name, Item Number, Price, Quality and Last received date

Scenario 2

Use case: Adding/Updating inventory

Actors: User, Database

Goal: Add or update new product to inventory

	Actor/User's Action		System's Response
		1	System prompts user with login screen

2	User inputs username and password and clicks Sign In	3	System checks if the credentials match/verify and takes user to the home page
4	User selects Navigate option at top left corner	5	System shows various options to the user
6	User selects Inventory	7	System shows inventory table with Set Name, Item Number, Price, Quantity and Last received date
8	User clicks the Add/Update option	9	System prompts user to fill in set information
10	User inputs set credentials and submits	11	System updates inventory table and adds the set to the database

Scenario 3

Use case: Deleting inventory

Actors: User, Database

Goal: Deleting a product from inventory

	Actor/User's Action		System's Response
		1	System prompts user with login screen
2	User inputs username and password and clicks Sign In	3	System checks if the credentials match/verify and takes user to home page

4	User selects Navigate option at top left corner	5	System shows various options to the user
6	User selects Inventory	7	System shows Lego Inventory DBMS with Set Name, Item Number, Price, Quantity and Last received date
8	User selects row they want to delete and click the Delete Row button	9	System deletes the row from the program and database

Scenario 4

Use case: Add Employees

Actors: Admin

Goal: To add an employee to the database

	Actor/Admin's Action		System's Response
		1	System prompts user with login screen
2	Admin inputs username and password and clicks Sign In	3	System checks if the credentials match/verify and takes user to home page
4	Admin selects Navigate option at top left corner	5	System shows various options to the user including Home, Employee, Inventory and Sign Out
6	Admin selects Employee	7	System shows the

	option		Employee table including their employee ID, First Name, Last Name and if they are Manager or not
8	Admin selects Add option	9	System prompts user to fill in employee details
10	Admin inputs employee credentials and submits	11	System updates employee table and adds the employee to the database

Scenario 5

Use case: Delete Employees

Actor: Admin

Goal: To delete employee information from the database

	Actor/Admin's Action		System's Response
		1	System prompts user with login screen
2	Admin inputs username and password and clicks Sign In	3	System checks if the credentials match/verify and takes user to Inventory Management page
4	Admin selects Navigate option at top left corner	5	System shows various options to the user including Home, Employee, Inventory

			and Sign Out
6	Admin selects Employee option	7	System shows the Employee table including their SSN, First Name, Last name and if they are managerial or not
8	Admin selects the row they want to delete and hits the delete button	9	System deletes the row from the program and from the database

Scenario 6

Use case: Sign up

Actor: User

Goal: Allows the user to make a username and password and have it registered and stored in the database

Precondition: Admin has added the employee with valid credentials into the database

	Actor/User's Action		System's Response
		1	System prompts user with login screen
2	User selects sign up button	3	System prompts user to enter login credentials
4	User inputs their login credentials and selects sign up button	5	System changes view to home page and adds login information to the database

Scenario 7

Use case: Filter selection

Actor: User (regular or admin)

Goal: Allows either a regular or admin user to filter the inventory table to

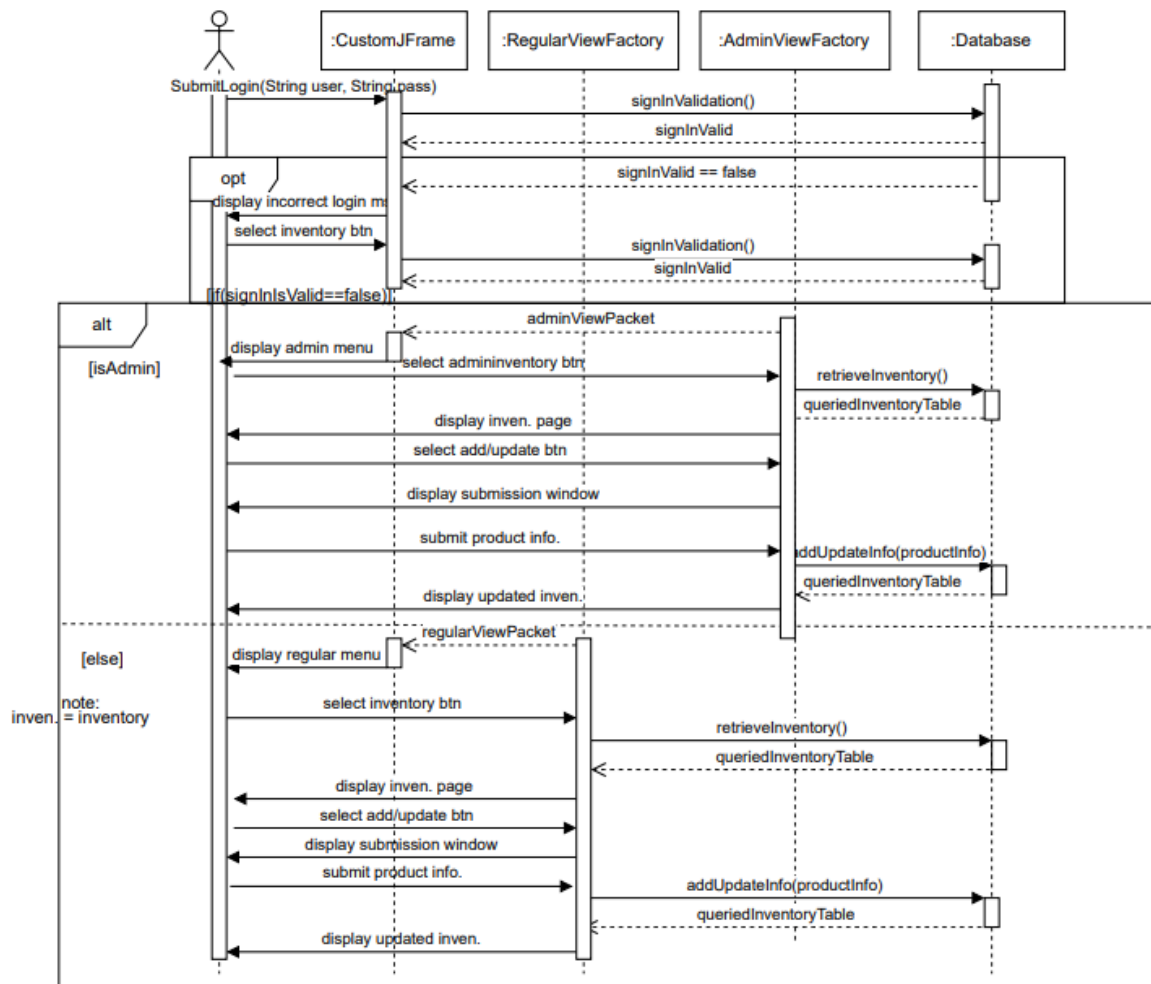
Better search for a certain item's status.

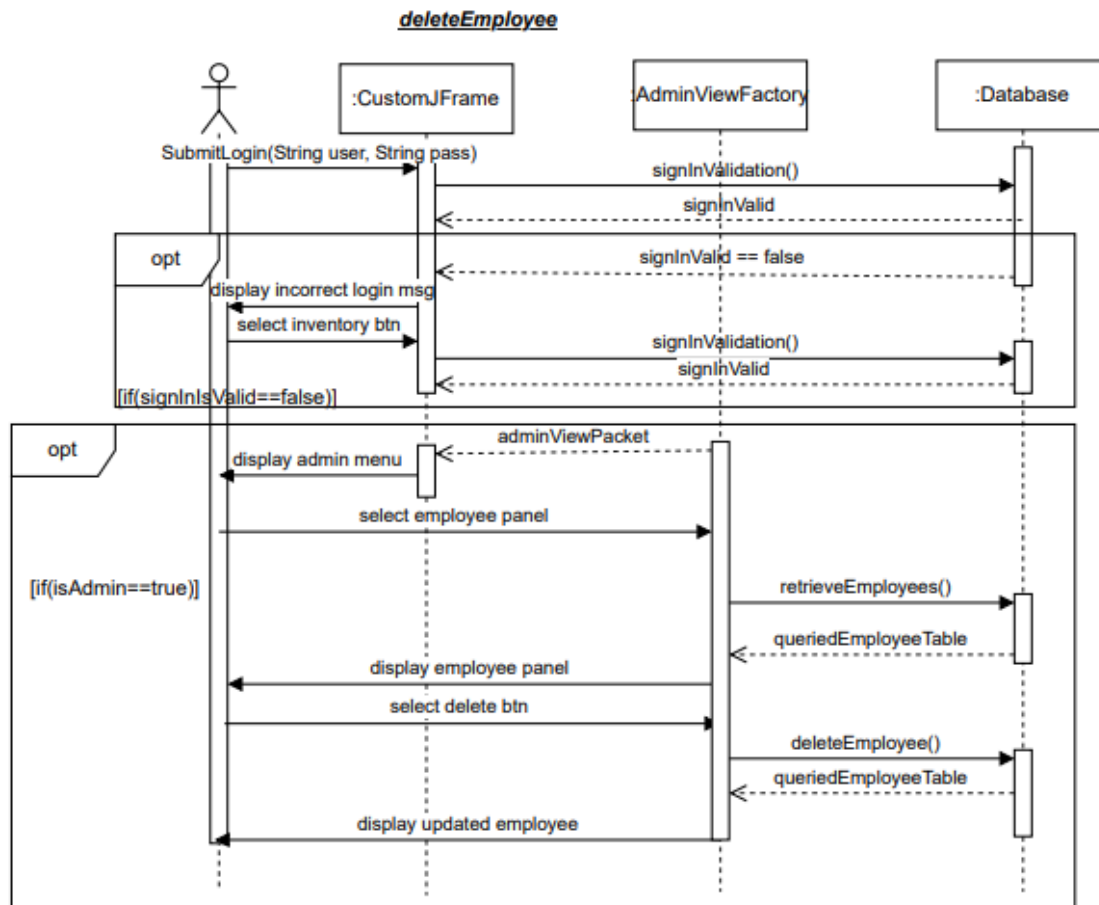
Precondition: Sets are already added to the database

	Actor/Admin's Action		System's Response
		1	System prompts user with login screen
2	User inputs username and password and clicks Sign In	3	System checks if the credentials match/verify and takes user to the home page
4	User selects the inventory menu option	5	System shows the inventory panel
6	User selects the filter text field and enters their filter word	7	System filters the inventory table to fit the user's passed in filter
8	User empties the text field and hits enter	9	System resets filter and restores table to show all products

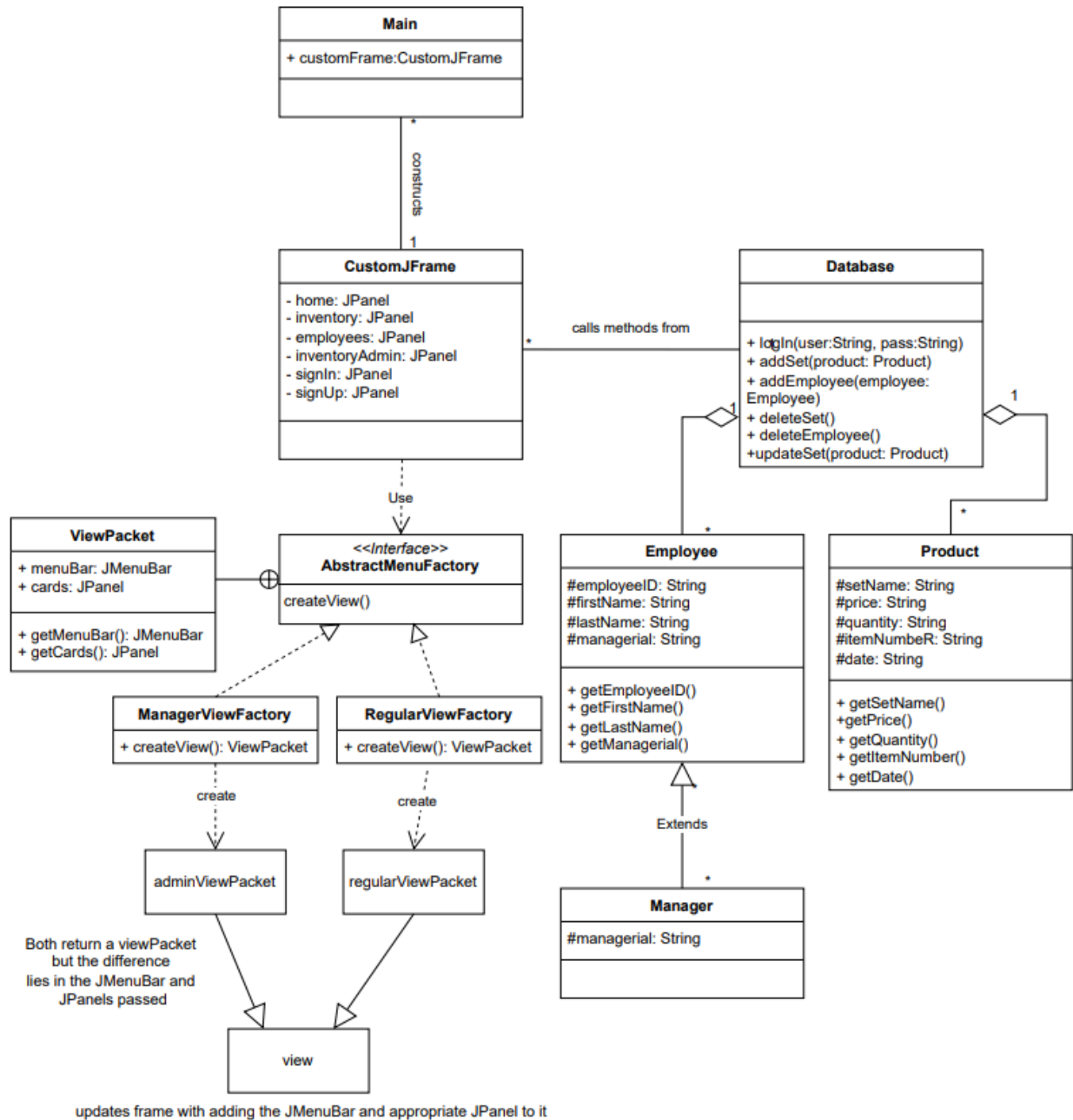
Sequence Diagrams

regular user add/update product information





Updated Class Diagram



System Test Cases

Test Case 1

Purpose:

Verify that admin's can properly log in

Requirement Traceability:

This should be a functional requirement but it was not listed.

Setup:

Setup the project to run and be connected to the UNF VPN.
Follow the directions below.

Test Data:

Action	Input	Expected Output
Run the application		The application's sign-in GUI appears with a username field and password field with a sign-in button and sign-up button below the fields.
Enter admin credentials	Enter brian for the username. Enter pass for the password.	The application shows the home page with the admin menu bar in the top left.

NOTE: What is meant by the admin menu bar is the option to select employees which by default, is not accessible to regular privileged users.

Test Case 2

Purpose:

Verify that regular users can properly log in and that the viewer
Can view the inventory.

Requirement Traceability:

Functional requirements 2 and 3. (However, the login portion was never a functional requirement but it should have been).

Setup:

Setup the project to run and be connected to the UNF VPN.
Follow the directions below.

Test Data:

Action	Input	Expected Output
Run the application		The application's sign-in GUI appears with a username field and password field with a sign-in button and sign-up button below the fields.
Enter regular user credentials	Enter guest for the username. Enter pass for the password. Select sign-in.	The application shows the home page with the regular menu bar in the top left.
Select inventory	Select inventory from menubar.	The application shows the inventory page and only has a button to add inventory and a text field to filter inventory.

NOTE: What is meant by the regular menu bar is only the option to select inventory and the inventory page should only allow adding inventory.

Test Case 3

Purpose:

Verify that admin users can delete inventory.

Requirement Traceability:

Functional requirements 3,4, and primarily 1.

Setup:

Setup the project to run and be connected to the UNF VPN.
Follow the directions below.

Test Data:

Action	Input	Expected Output
Run the application		The application's sign-in GUI appears with a username field and password field with a sign-in button and sign-up button below the fields.
Enter admin user credentials	Enter brian for the username. Enter pass for the password. Select sign-in.	The application shows the home page with the admin menu bar in the top left.
Select inventory	Select inventory from menu bar.	The application shows the inventory page with the inventory table and it has the option to add inventory, delete inventory, and filter inventory.
Select a row to be deleted	Select the desired inventory item to be deleted.	The application reloads the table, showing the selected item has disappeared from the table.

Test Case 4

Purpose:

Verify that admin users can add employees to the database as well as delete them.

Requirement Traceability:

Functional requirement 3,4, but primarily 5 and 6.

Setup:

Setup the project to run and be connected to the UNF VPN.

Follow the directions below.

Test Data:

Action	Input	Expected Output
Run the application		The application's sign-in GUI appears with a username field and password field with a sign-in button and sign-up button below the fields.
Enter regular user credentials	Enter guest for the username. Enter pass for the password. Select sign-in.	The application shows the home page with the regular menu bar in the top left.
Select employee	Select employee from menu bar.	The application shows the employee page which has buttons to add and delete employees and the employee table itself.
Select add	Select add button and input employee credentials as: First name as test Last name as test Employee ID as 112233 Manager as Y Then hit ok.	The application updates the table to show the test employee.

Select test employee row and select delete	Select the test employee and then hit the delete button.	The application updates the table to show the test employee has been removed.
--	--	---

Test Case 5

Purpose:

Verify that any user can sign in after signing up.

Requirement Traceability:

Functional requirement 8.

Setup:

Setup the project to run and be connected to the UNF VPN.

Follow the directions below.













Test Data:

NOTE REPEAT ADDING A TEST USER FROM THE STEPS IN TEST CASE 4'S RELEVANT ADDING EMPLOYEE SECTION

Action	Input	Expected Output
Run the application		The application's sign-in GUI appears with a username field and password field with a sign-in button and sign-up button below the fields.
Select sign-up	Select sign-up	The application shows the sign-up page with the employee ID field, username field, and password field.
Enter the credentials you desire and enter employee ID	Enter the test employee ID you chose for it and then a desired	The application shows the same page.

	username and password.	
Select cancel to return to the previous page and sign-in.		The application shows the sign-in page.
Enter login credentials.	Enter the login credentials previously chosen and hit sign-in.	The application shows the home page and the appropriate menu bar (in this case it was admin because we chose admin employee)

Unit Test Coverage and vi Test

▼ CEN4010		17.8 %	532	2,454	2,986
> ManagerViewFactory.java		0.0 %	0	920	920
> Database.java		0.0 %	0	613	613
> RegularViewFactory.java		0.0 %	0	557	557
> CustomJFrame.java		71.8 %	532	209	741
> Employee.java		0.0 %	0	53	53
> Manager.java		0.0 %	0	36	36
> Product.java		0.0 %	0	33	33
> AbstractMenuFactory.java		0.0 %	0	25	25
> Main.java		0.0 %	0	8	8
▼ test		96.4 %	212	8	220
> CEN4010		96.4 %	212	8	220

Commit Logs

main		All users	All time
Commits on Apr 20, 2024			
Added final iteration 0 version SiivaHalation committed now		Verified fba3040	
Delete CEN4010GRP3newITERATION0.docx SiivaHalation committed now		Verified 058069b	
Add read me SiivaHalation committed 35 minutes ago		Verified 62601c1	
Add final version of the UML diagram SiivaHalation committed 35 minutes ago		Verified 87ad75d	
Delete ClassDiagramCEN4010.jpg SiivaHalation committed 36 minutes ago		Verified 762ca8b	
Adding final sequence diagram SiivaHalation committed 36 minutes ago		Verified 02f4254	
Delete SequenceDiagramCEN4010.drawio.pdf SiivaHalation committed 37 minutes ago		Verified fedb67b	
Added regular view factory SiivaHalation committed 38 minutes ago		Verified 295a971	
Update ManagerMenuBarFactory.java		Verified f451cbf	
Update CustomJFrame.java		Verified ce834c0	
Commits on Apr 16, 2024			

```
Last login: Tue Mar 26 18:43:50 on ttys000
[briandinh@Brians-MacBook-Air-2 ~ % cd G3
[briandinh@Brians-MacBook-Air-2 G3 % git init
Reinitialized existing Git repository in /Users/briandinh/G3/.git/
[briandinh@Brians-MacBook-Air-2 G3 % git log
commit e342e02de8051cc270bd917eed31516043a17158 (HEAD -> main, origin/main, origin/HEAD)
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:36:56 2024 -0400
```

Add files via upload

Messed up

```
commit f951ecf46b51e221fc48cf6fc0b9b6e73023cd05
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:35:54 2024 -0400
```

Delete InventoryManagerDBMS directory

```
commit 4ac282f897146d3748d1ae5e3892d47effa01674
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:33:47 2024 -0400
```

Delete InventoryManagerDBMS/src/.DS_Store

```
commit d83ed5b194ac851243b18a001e582a3ed68e01e9
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
:...skipping...
commit e342e02de8051cc270bd917eed31516043a17158 (HEAD -> main, origin/main, origin/HEAD)
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:36:56 2024 -0400
```

Add files via upload

Messed up

```
commit f951ecf46b51e221fc48cf6fc0b9b6e73023cd05
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:35:54 2024 -0400
```

Delete InventoryManagerDBMS directory

```
commit 4ac282f897146d3748d1ae5e3892d47effa01674
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:33:47 2024 -0400
```

Delete InventoryManagerDBMS/src/.DS_Store

```
commit d83ed5b194ac851243b18a001e582a3ed68e01e9
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:33:31 2024 -0400

:...skipping...
commit e342e02de8051cc270bd917eed31516043a17158 (HEAD -> main, origin/main, origin/HEAD)
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:36:56 2024 -0400
```

Add files via upload

Messed up

```
commit f951ecf46b51e221fc48cf6fc0b9b6e73023cd05
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:35:54 2024 -0400
```

```

commit 4ac282f897146d3748d1ae5e3892d47effa01674
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:33:47 2024 -0400

Delete InventoryManagerDBMS/src/.DS_Store

commit d83ed5b194ac851243b18a001e582a3ed68e01e9
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:33:31 2024 -0400

Delete InventoryManagerDBMS/test/.DS_Store

commit 11fe5d768ad324feaa633f0b806b3551781c4e79
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:32:17 2024 -0400

Delete InventoryManagerDBMS/.DS_Store

commit 2aa3f2f42e1d73cf2e5bdeef2efbc560f3a49558
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Tue Mar 26 17:32:03 2024 -0400

Delete .DS_Store

commit 9e9557889a8861b847899e832a7b05d1755df75c
Author: SiivaHalation <briandinh160@gmail.com>
Date: Tue Mar 26 17:31:16 2024 -0400

t p:wq# modified: InventoryManagerDBMS/src/CEN4010/CustomJFrame.java

commit 07be9c3a5c82f4a72213af867220f7506e420f87
Author: SiivaHalation <briandinh160@gmail.com>
Date: Tue Mar 26 17:13:49 2024 -0400

adding UML

commit fad786cc053f70401cf6b6152ad810235c98d082
Author: SiivaHalation <briandinh160@gmail.com>
Date: Sun Mar 24 16:19:18 2024 -0400

adding sequence diagram as pdf

commit 92fbded7d2aac707b1039228bf1b02870b5fcd0f
Author: Trung Dinh <157513352+SiivaHalation@users.noreply.github.com>
Date: Sat Mar 23 16:10:51 2024 -0400

Readding with correct name

commit 0d53c8970829edfdd45218c842ebd2d4f2563295
:

```

READ ME

LEGO Inventory DBMS | CEN 4010

#####

#Summary#

#####

This program serves a light weight inventory database management system that keeps track of the employees that can access the program and the LEGO sets that are within the store's inventory. The program allows for administrator accounts to create, read, update, and delete inventory entries but only allows create, read, and delete for employee credentials. The program allows regular users to only create and read inventory entries; Regular employees have no access to employee information.

#####

#Running the Program initially#

#####

**To run the program you can clone it:
<https://github.com/UNF-CEN4010/G3.git>**

**Additionally, you will need ojdbc11 which can be downloaded here:
<https://www.oracle.com/database/technologies/appdev/jdbc-downloads.html>**

Or download it as a zip and then open eclipse IDE. Once you have opened it select JavaProject from the top left button marked as new. Here you will uncheck the default location check box and then browse for the InventoryDBMSManager folder. You will select it when you find it. Afterwards, you will need to add the ojdbc11.jar and junit5 library to the build path under

the class path section in the library tab. For ojdbc11, you will select add external JARs and select it wherever you chose to have it when you downloaded it. Afterwards, you will select add library and select junit then make sure it is junit 5 after selecting junit. Apply and close and the project should be able to be ran.

#####

#Deviations from the original iteration#

#####

The sign-out feature no longer takes the user back to the sign-in page. This was done due to restrictions of working with different views and the code behind it.

There will no longer be a message board due to time constraints.

#####

#Design Pattern(s) used#

#####

The only pattern used was abstractfactory. This was used to achieve the admin and regular view by having a ViewPacket passed back to the CustomJFrame class that would have the respective admin or regular JMenuBar that would access the admin or regular JPanel using a cardLayout manager.