
DEVELOPMENT PLANNING DOCUMENT

for

Encost Smart Graph Project

Version 1.0

Prepared by: Joey Han
SoftFlux Engineer

SoftFlux

May 19, 2023

Contents

1	Introduction/Purpose	3
1.1	Purpose	3
1.2	Document Conventions	3
1.3	Intended Audience and Reading Suggestions	4
1.4	Project Scope	4
2	Specialized Requirements Specification	4
3	Product Backlog	5
3.1	Features	5
3.1.1	Categorising Users	5
3.1.2	ESGP Account Login	5
3.1.3	ESGP Feature Options	5
3.1.4	Loading the Encost Smart Homes Dataset	6
3.1.5	Categorising Smart Homes Devices	6
3.1.6	Building a Graph Data Type	7
3.1.7	Graph Visualisation	7
3.1.8	Calculating Device Distribution	8
4	Sprint Details	8
4.1	Sprint #1 <10/05/2023 7am – 12/05/2023 7am>	8
4.1.1	Product Backlog Items	8
4.1.2	Sprint Tasks	9
4.1.3	Software Design	10
4.1.4	Software Testing	13
4.1.5	Sprint Task Completion	15
4.2	Sprint #2 <12/05/2023 7am – 14/05/2023 7am>	16
4.2.1	Product Backlog Items	16
4.2.2	Sprint Tasks	17
4.2.3	Software Design	18
4.2.4	Software Testing	19
4.2.5	Sprint Task Completion	20
4.3	Sprint #3 <14/05/2023 7am – 16/05/2023 7am>	20
4.3.1	Product Backlog Items	20
4.3.2	Sprint Tasks	21
4.3.3	Software Design	21
4.3.4	Software Testing	22
4.3.5	Sprint Task Completion	25
5	Conclusion	26

Revision History

Name	Date	Reason for Changes	Version
SoftFlux	19/05/2023	Intro Purpose, Sprints	

1 Introduction/Purpose

1.1 Purpose

This document is Development Planning Document for the software Encost Smart Graph Project (ESGP) for the company Encost. The purpose of this document is to provide a way to compile all the tasks needed to do and track their progress over the entire software development process.

1.2 Document Conventions

This document uses the following conventions:

- ESGP: Encost Smart Graph Project
- ESHD: Encost Smart Home Dataset
- GSL: GraphStream Library
- CSV: Comma-Separated Values
- WIP: Work in progress

1.3 Intended Audience and Reading Suggestions

This document is intended for any developer, product owner, and project manager. Here are the potential uses for each of the reader types:

- Project Manager and Product Owner: Uses this document to track the progress of the software development.
- Developer: Uses this document to help implement software and track their progress.

1.4 Project Scope

The ESGP is a software system with aims to enable the Encost smart devices to be visualised in a graph data structure from the command line. With both accessibility to Encost verified users and community users but with restricted access to the software capability. There will be no hardware integration required for the ESGP nor for the testing purposed of the software. SoftFlux is responsible to the software development of the ESGP and the insurance that the software meets the requirements listed in the functional software test plan 5 and the SDS 1.

2 Specialized Requirements Specification

- Only high priorities features are made in the application. All low or medium priorities could be in the application, but they will not be available to use and will tell the user that it is not available or will only show the high priority part of the feature.
- Updated the software test plan 5 test codes due to various bugs. The updated code should have the same intention as the old codes. Updated the csv files so that the valid ones have the right type. E.g validTest.csv file had device type Washing Machine when it was supposed to be Washing Machine/Dryer.
- Added a parameter to the LoadFeaturesTest java file method as it is impossible to test for inputs without the parameter input and usertype as they have similar inputs, but they do different things.
- Added more methods to device class so that it can have all the device information filled up using the loading dataset csv file and helps assist the building a graph type more.
- For the building a graph data type, I will only made two attributes devices which is the linkedlist and the graph which is graph stream library in SDS 1. I also

only made `getDevices`, `addDevice`, `display`, `addEdge`, `getDevicesByCategory`, `getDevicesByType` and `getDevicesbyProduct` because the device distribution is the only high priority for summary statistics and the other methods are not needed.

- Changed `DeviceDistributionTest` and added more tests so that it tests categories, type and product name instead of just the type only.
- The graph visualisation created will not be like Figure 3.3 until our graph stream library upgrades to the 2.0 version. For now, it will have everything that is shown in the Figure 3.3 except for the formatting shown.

3 Product Backlog

3.1 Features

3.1.1 Categorising Users

User Stories:

- Sally Sallyson is one of the Encost Smart Device customers who opted to have their device information recorded (see 2.2 in SRS). As such, Encost would like for her to be able to view the Encost data. To view the data, Sally needs to be able to indicate that she is an Encost Community User and needs to be prompted by the ESGP feature options.
- James Wally is one of the Encost employees. He needs to be able to indicate that he is an encost-unverified user and needs to be prompted to ESGP account login.

3.1.2 ESGP Account Login

User Stories:

- James Wally is one of the Encost employees. He wants to be able to login with his username and password to to get access to his encost-verified user features.

3.1.3 ESGP Feature Options

User Stories:

- Sally Sallyson is one of the Encost Smart Device customers who opted to have their device information recorded (see 2.2 in SRS). As such, Sally needs to be able to access the displaying graph data and exit options.

- James Wally is one of the Encost employees. He needs to be able to access the loading custom dataset, displaying graph data, displaying summary statistics and exit options.

3.1.4 Loading the Encost Smart Homes Dataset

User Stories:

- As a product owner, I want my encost smart homes dataset to be in the system. I want my system to be able to find the dataset file, extract the relevant data and load it when the software application is opened. I want the dataset file to be a CSV formatted file.

Device ID	Date Connected	Device name	Device type	Household ID	Router Connection	Sends	Receives
EWR-1234	01/04/22	Encost Router 360	Router	WKO-1234	-	Yes	Yes
ELB-4567	01/04/22	Encost Smart Bulb B22 (multi colour)	Light bulb	WKO-1234	EWR-1234	No	Yes
EK-9876	07/05/22	Encost Smart Jug	Kettle	WKO-1234	EWR-1234	No	Yes
EHC-2468	01/04/22	Encost Smart Hub 2.0	Hub/Controller	WKO-1234	EWR-1234	Yes	Yes

Figure 3.1: Dataset file example with relevant data

3.1.5 Categorising Smart Homes Devices

User Stories:

- As a product owner, I want my system to be able to categorise each encost smart device into one of the five device categories shown in Figure 3.2 after it reads and processes the ESHD. I also want my system to create an object which it can be stored in, and the object should have all the information of the device.

Device Category	Device types	Device Names
Encost Wifi Routers	Router, Extender	Encost Router 360, Encost Router Plus, Encost Wifi Range Extender 1.0, Encost Wifi Range Extender 2.0
Encost Hubs/- Controllers	Hub/Controller	Encost Smart Hub, Encost Smart Hub 2.0, Encost Smart Hub Mini
Encost Smart Lighting	Light Bulb, Strip Lighting, Other Lighting	Encost Smart Bulb B22 (white), Encost Smart Bulb B22 (multi colour), Encost Smart Bulb E26 (white), Encost Smart Bulb E26 (multi colour), Encost Strip Lighting (white), Encost Strip Lighting (multi colour), Encost Novelty Light (giraffe), Encost Novelty Light (lion), Encost Novelty Light (bear)
Encost Smart Appliances	Kettle, Toaster, Coffee Maker	Encost Smart Jug, Encost Smart Whistling Kettle, Encost Smart Toaster (2 slice), Encost Smart Toaster (4 slice), Encost Smart Coffee Maker, Encost Smart Coffee Maker Mini, Encost Smart Coffee Maker Pro
Encost Smart Whiteware	Washing Machine/Dryer, Refrigerator/Freezer, Dishwasher	Encost Smart Washer, Encost Smart Washer Pro, Encost Smart Dryer, Encost Smart Dryer Pro, Encost Smart Refrigerator, Encost Smart Freezer, Encost Smart Refrigerator/Freezer Combo, Encost Dishwasher, Encost Dishwasher Pro

Figure 3.2: Device table

3.1.6 Building a Graph Data Type

User Stories:

- As a product owner, I want my system to be able store each object in a graph data structure. I want my graph data structure to be able to use for summary statistics and graph visualisation.

3.1.7 Graph Visualisation

User Stories:

- Sally Sallyson is one of the Encost Smart Device customers who opted to have their device information recorded (see 2.2 in SRS). Sally needs to be able to view the visualisation of the graph which shows her devices, different colour device categories, the ability to send and receive commands from other devices and the connection of all her devices.

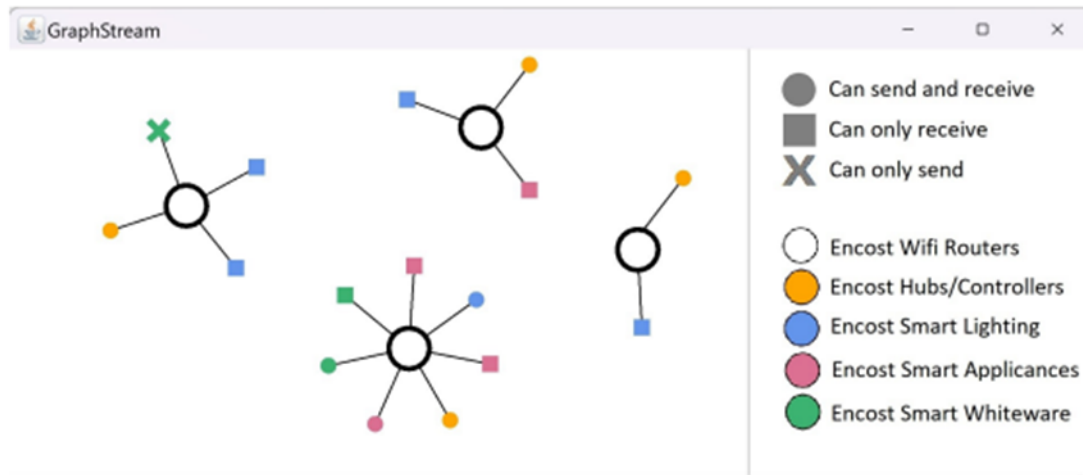


Figure 3.3: Graph visualisation

3.1.8 Calculating Device Distribution

User Stories:

- James Wally is one of the Encost employees. He needs to be able to view distribution of devices across category, type, and name, based on the information stored in the graph data structure using the summary statistics option in the ESGP feature options.

4 Sprint Details

4.1 Sprint #1 <10/05/2023 7am – 12/05/2023 7am>

4.1.1 Product Backlog Items

- Sally Sallyson is one of the Encost Smart Device customers who opted to have their device information recorded (see 2.2 in SRS). As such, Encost would like for her to be able to view the Encost data. To view the data, Sally needs to be able to indicate that she is an Encost Community User and needs to be prompted by the ESGP feature options.
- James Wally is one of the Encost employees. He needs to be able to indicate that he is an encost-unverified user and needs to be prompted to ESGP account login.

- James Wally is one of the Encost employees. He wants to be able to login with his username and password to get access to his encost-verified user features.
- Sally Sallyson is one of the Encost Smart Device customers who opted to have their device information recorded (see 2.2 in SRS). As such, Sally needs to be able to access the displaying graph data and exit options.
- James Wally is one of the Encost employees. He needs to be able to access the loading custom dataset, displaying graph data, displaying summary statistics and exit options.
- (Added) As a product owner, I want my encost smart homes dataset to be in the system. I want my system to be able to find the dataset file, extract the relevant data and load it when the software application is opened. I want the dataset file to be a CSV formatted file.
- (Added) As a product owner, I want my system to be able to categorise each encost smart device into one of the five device categories in the device table after it reads and processes the ESHD. I also want my system to create an object which it can be stored in, and the object should have all the information of the device.

4.1.2 Sprint Tasks

Categorising users:

- User able to indicate community or encost-unverified user.
- Prompting user depending on their types.
- Input validation on back-end.

ESGP account login:

- Encost user login to be verified.
- Encost user login validation on back-end.
- Prompting user for what feature they want to use.

ESGP feature options:

- Input validation on back-end.
- Community user able to access their features.
- Encost user able to access their features.

Loading the ESHD:

- Error handling and checking.

- CSV formatted file.
- Dataset to be extracted and loaded when software application is opened.
- Adding device object to graph.

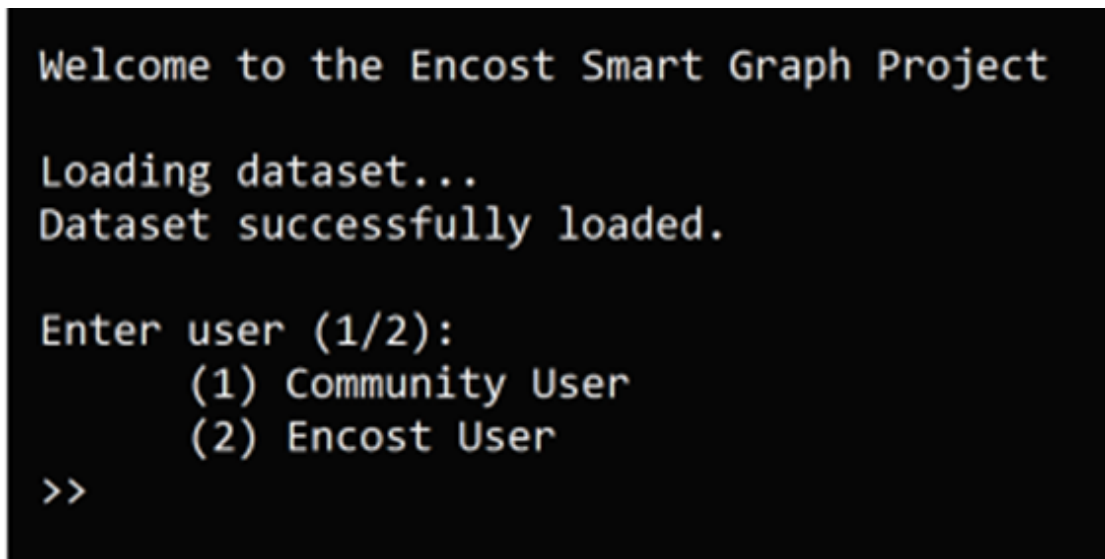
Categorising smart home devices:

- Sorting each encost smart device object into its own categories.
- Creating devices class.

4.1.3 Software Design

SDS for categorising users:

- Changing the Encost User output after the user inputs 2 to be Encost-unverified User.



```

Welcome to the Encost Smart Graph Project

Loading dataset...
Dataset successfully loaded.

Enter user (1/2):
    (1) Community User
    (2) Encost User
>>

```

Figure 4.1: Categorising users prompt design

SDS for ESGP account login:

- Adding a line before the welcome message which says Successful Login when user have valid credentials.
- Adding a space and brackets besides user_1 which says (Encost-verified User).

```
Enter user (1/2):  
    (1) Community User  
    (2) Encost User  
>> 2  
  
Encost User  
  
Enter username: user_1  
Enter password:  
  
Welcome user_1
```

Figure 4.2: Account login prompt design

SDS for ESGP feature options:

```
Community User

ESGP Features:
    (1) Graph representation of data
    (X) Exit
Select option >>
```

Figure 4.3: Feature options for community user design

```
ESGP Features:
    (1) Load custom dataset
    (2) Graph representation of data
    (3) View summary statistics
    (X) Exit
Select option >>
```

Figure 4.4: Feature options for encost user design

4.1.4 Software Testing

```
.
+-- JUnit Jupiter [OK]
'-- JUnit Vintage [OK]
    '-- CategorizingUsersTest [OK]
        +-- [0] [OK]
        | '-- evaluatesUser[0] [OK]
        +-- [1] [OK]
        | '-- evaluatesUser[1] [OK]
        +-- [2] [OK]
        | '-- evaluatesUser[2] [OK]
        +-- [3] [OK]
        | '-- evaluatesUser[3] [OK]
        +-- [4] [OK]
        | '-- evaluatesUser[4] [OK]
        +-- [5] [OK]
        | '-- evaluatesUser[5] [OK]
        +-- [6] [OK]
        | '-- evaluatesUser[6] [OK]
        +-- [7] [OK]
        | '-- evaluatesUser[7] [OK]
        +-- [8] [OK]
        | '-- evaluatesUser[8] [OK]
        +-- [9] [OK]
        | '-- evaluatesUser[9] [OK]
        +-- [10] [OK]
        | '-- evaluatesUser[10] [OK]
        +-- [11] [OK]
        | '-- evaluatesUser[11] [OK]
        '-- [12] [OK]
        '-- evaluatesUser[12] [OK]

Test run finished after 68 ms
[      16 containers found      ]
[       0 containers skipped    ]
[      16 containers started    ]
[       0 containers aborted    ]
[      16 containers successful ]
[       0 containers failed     ]
[      13 tests found           ]
[       0 tests skipped         ]
[      13 tests started         ]
[       0 tests aborted         ]
[      13 tests successful      ]
[       0 tests failed          ]
```

Figure 4.5: JUnit test for categorising users

```

.
+-- JUnit Jupiter [OK]
'-- JUnit Vintage [OK]
    '-- AccountLoginTest [OK]
        +-- [0] [OK]
            | '-- testLogin[0] [OK]
        +-- [1] [OK]
            | '-- testLogin[1] [OK]
        +-- [2] [OK]
            | '-- testLogin[2] [OK]
        +-- [3] [OK]
            | '-- testLogin[3] [OK]
        +-- [4] [OK]
            | '-- testLogin[4] [OK]
        +-- [5] [OK]
            | '-- testLogin[5] [OK]
        +-- [6] [OK]
            | '-- testLogin[6] [OK]
        +-- [7] [OK]
            | '-- testLogin[7] [OK]
        +-- [8] [OK]
            | '-- testLogin[8] [OK]
        +-- [9] [OK]
            | '-- testLogin[9] [OK]
        +-- [10] [OK]
            | '-- testLogin[10] [OK]
        +-- [11] [OK]
            | '-- testLogin[11] [OK]
        +-- [12] [OK]
            | '-- testLogin[12] [OK]
        +-- [13] [OK]
            | '-- testLogin[13] [OK]
        '-- [14] [OK]
            '-- testLogin[14] [OK]

Test run finished after 77 ms
[      18 containers found      ]
[       0 containers skipped    ]
[      18 containers started    ]
[       0 containers aborted    ]
[      18 containers successful ]
[       0 containers failed     ]
[      15 tests found           ]
[       0 tests skipped         ]
[      15 tests started         ]
[       0 tests aborted         ]
[      15 tests successful      ]

```

Figure 4.6: JUnit test for account login

```

+-- JUnit Jupiter [OK]
'-- JUnit Vintage [OK]
  '-- LoadFeaturesTest [OK]
    +-- [0] [OK]
    | '-- testFeature[0] [OK]
    +-- [1] [OK]
    | '-- testFeature[1] [OK]
    +-- [2] [OK]
    | '-- testFeature[2] [OK]
    +-- [3] [OK]
    | '-- testFeature[3] [OK]
    +-- [4] [OK]
    | '-- testFeature[4] [OK]
    +-- [5] [OK]
    | '-- testFeature[5] [OK]
    +-- [6] [OK]
    | '-- testFeature[6] [OK]
    +-- [7] [OK]
    | '-- testFeature[7] [OK]
    +-- [8] [OK]
    | '-- testFeature[8] [OK]
    +-- [9] [OK]
    | '-- testFeature[9] [OK]
    +-- [10] [OK]
    | '-- testFeature[10] [OK]
    +-- [11] [OK]
    | '-- testFeature[11] [OK]
    +-- [12] [OK]
    | '-- testFeature[12] [OK]
    '-- [13] [OK]
    '-- testFeature[13] [OK]

Test run finished after 74 ms
[      17 containers found      ]
[      0 containers skipped     ]
[      17 containers started     ]
[      0 containers aborted     ]
[      17 containers successful  ]
[      0 containers failed      ]
[      14 tests found           ]
[      0 tests skipped          ]
[      14 tests started          ]
[      0 tests aborted           ]
[      14 tests successful       ]
[      0 tests failed            ]

```

Figure 4.7: JUnit test for ESGP feature options

4.1.5 Sprint Task Completion

- Categorising users: Done
- ESGP account login: Done
- ESGP feature options: Done

- Categorising smart home devices: WIP
- Loading the ESHD: WIP
- Loading the ESHD and Categorising smart home devices is rolled over to next sprint as it needs the building a graph data type to work before it is fully done.

4.2 Sprint #2 <12/05/2023 7am – 14/05/2023 7am>

4.2.1 Product Backlog Items

- As a product owner, I want my encost smart homes dataset to be in the system. I want my system to be able to find the dataset file, extract the relevant data and load it when the software application is opened. I want the dataset file to be a CSV formatted file.
- As a product owner, I want my system to be able to categorise each encost smart device into one of the five device categories in the device table after it reads and processes the ESHD. I also want my system to create an object which it can be stored in, and the object should have all the information of the device.
- As a product owner, I want my system to be able store each object in a graph data structure. I want my graph data structure to be able to use for summary statistics and graph visualisation.
- Sally Sallyson is one of the Encost Smart Device customers who opted to have their device information recorded (see 2.2 in SRS). Sally needs to be able to view the visualisation of the graph which shows her devices, different colour device categories, the ability to send and receive commands from other devices and the connection of all her devices.
- James Wally is one of the Encost employees. He needs to be able to view distribution of devices across category, type, and name, based on the information stored in the graph data structure using the summary statistics option in the ESGP feature options.

4.2.2 Sprint Tasks

Loading the ESHD:

- Adding device object to graph.

Categorising smart homes devices:

- Sorting each encost smart device object into its own categories.

Building a graph data type:

- Adding device object to graph.
- Making methods to aid in the graph visualisation and summary statistics.

Graph visualisation:

- Show the graph to the user in the way of Figure 3.3 of the graph visualisation example without formatting as stated in the specialized requirements specification.

Calculating device distribution:

- Calculating the device distribution by product category, product type, product name.
- Displaying it on the console to the user in the format of the device distribution output example.

4.2.3 Software Design

SDS for device distribution:

Device Distribution	
=====	
Encost Wifi Routers	20
Router	15
Encost Router 360	2
Encost Router Plus	13
Extender	5
Encost Wifi Range Extender 1.0	0
Encost Wifi Range Extender 2.0	5
Encost Hubs/Controllers	6
Hub/Controller	6
Encost Smart Hub	1
Encost Smart Hub 2.0	3
Encost Smart Hub Mini	2
Encost Smart Lighting	34
Light Bulb	12
Encost Smart Bulb B22 (white)	4
Encost Smart Bulb B22 (multi colour)	3
Encost Smart Bulb E26 (white)	2
Encost Smart Bulb E26 (multi colour)	3
Strip Lighting	4
Encost Strip Lighting (white)	1
Encost Strip Lighting (multi colour)	3
Other Lighting	18
Encost Novelty Light (giraffe)	10
Encost Novelty Light (lion)	6
Encost Novelty Light (bear)	2

Figure 4.8: Device distribution output design

4.2.4 Software Testing

Device Distribution		
=====		
Encost Wifi Routers	1	
Router	1	
Encost Router 360	1	
Encost Router Plus	0	
Extender	0	
Encost Wifi Range Extender 1.0	0	
Encost Wifi Range Extender 2.0	0	
Encost Hubs/Controllers	1	
Hub/Controller	1	
Encost Smart Hub	0	
Encost Smart Hub 2.0	1	
Encost Smart Hub Mini	0	
Encost Smart Lighting	1	
Light Bulb	1	
Encost Smart Bulb B22 (white)	0	
Encost Smart Bulb B22 (multi colour)	1	
Encost Smart Bulb E26 (white)	0	
Encost Smart Bulb E26 (multi colour)	0	
Strip Lighting	0	
Encost Strip Lighting (white)	0	
Encost Strip Lighting (multi colour)	0	
Other Lighting	0	
Encost Novelty Light (giraffe)	0	
Encost Novelty Light (lion)	0	
Encost Novelty Light (bear)	0	
Encost Smart Appliances	1	
Kettle	1	
Encost Smart Jug	1	
Encost Smart Whistling Kettle	0	
Toaster	0	
Encost Smart Toaster (2 slice)	0	
Encost Smart Toaster (4 slice)	0	
Coffee Maker	0	
Encost Smart Coffee Maker	0	
Encost Smart Coffee Maker Mini	0	
Encost Smart Coffee Maker Pro	0	

Figure 4.9: Device distribution output testing part 1

Encost Smart Whiteware	0
Washing Machine/Dryer	0
Encost Smart Washer	0
Encost Smart Washer Pro	0
Encost Smart Dryer	0
Encost Smart Dryer Pro	0
Refrigerator/Freezer	0
Encost Smart Refrigerator	0
Encost Smart Freezer	0
Encost Smart Refrigerator/Freezer Combo	0
Dishwasher	0
Encost Dishwasher	0
Encost Dishwasher Pro	0

Figure 4.10: Device distribution output testing part 2

4.2.5 Sprint Task Completion

- Loading the ESHD: WIP
- Building a graph data type: WIP
- Graph visualisation: WIP
- Calculating device distribution: WIP
- All WIP because the testing for all of them weren't passing the tests.

4.3 Sprint #3 <14/05/2023 7am – 16/05/2023 7am>

4.3.1 Product Backlog Items

- As a product owner, I want my encost smart homes dataset to be in the system. I want my system to be able to find the dataset file, extract the relevant data and load it when the software application is opened. I want the dataset file to be a CSV formatted file.
- As a product owner, I want my system to be able to categorise each encost smart device into one of the five device categories in the device table after it reads and processes the ESHD. I also want my system to create an object which it can be stored in, and the object should have all the information of the device.
- As a product owner, I want my system to be able store each object in a graph data structure. I want my graph data structure to be able to use for summary statistics and graph visualisation.

- Sally Sallyson is one of the Encost Smart Device customers who opted to have their device information recorded (see 2.2 in SRS). Sally needs to be able to view the visualisation of the graph which shows her devices, different colour device categories, the ability to send and receive commands from other devices and the connection of all her devices.
- James Wally is one of the Encost employees. He needs to be able to view distribution of devices across category, type, and name, based on the information stored in the graph data structure using the summary statistics option in the ESGP feature options.

4.3.2 Sprint Tasks

Loading the ESHD:

- Adding device object to graph.

Categorising smart homes devices:

- Sorting each encost smart device object into its own categories.

Building a graph data type:

- Adding device object to graph.
- Making methods to aid in the graph visualisation and summary statistics.

Graph visualisation:

- Show the graph to the user in the way of Figure 3.3 of the graph visualisation example without formatting as stated in the specialized requirements specification.

Calculating device distribution:

- Calculating the device distribution by product category, product type, product name.
- Displaying it on the console to the user in the format of the device distribution output example.

4.3.3 Software Design

The software design is in sprint #2.

4.3.4 Software Testing

```
MD5 hash of test.csv:  
aad38becaee5e197c804702f32aaca1e  
CertUtil: -hashfile command completed successfully.
```

Figure 4.11: MD5 Testing for loading the ESHD Part 1

```
MD5 hash of Encost_Smart_Homes_Dataset_Small.csv:  
aad38becaee5e197c804702f32aaca1e  
CertUtil: -hashfile command completed successfully.
```

Figure 4.12: MD5 Testing for loading the ESHD Part 2

```
.
+-- JUnit Jupiter [OK]
'-- JUnit Vintage [OK]
    '-- CategorisingDevicesTest [OK]
        +-- validCSVTest [OK]
        '-- invalidCSVTest [OK]

Test run finished after 128 ms
[      3 containers found      ]
[      0 containers skipped    ]
[      3 containers started    ]
[      0 containers aborted    ]
[      3 containers successful  ]
[      0 containers failed     ]
[      2 tests found           ]
[      0 tests skipped         ]
[      2 tests started         ]
[      0 tests aborted         ]
[      2 tests successful      ]
[      0 tests failed          ]
```

Figure 4.13: JUnit test for categorising smart home devices

```
.
+-- JUnit Jupiter [OK]
'-- JUnit Vintage [OK]
    '-- BuildGraphTest [OK]
        +-- invalidGraphTest [OK]
        '-- validGraphTest [OK]

Test run finished after 146 ms
[      3 containers found      ]
[      0 containers skipped    ]
[      3 containers started    ]
[      0 containers aborted    ]
[      3 containers successful  ]
[      0 containers failed     ]
[      2 tests found           ]
[      0 tests skipped         ]
[      2 tests started         ]
[      0 tests aborted         ]
[      2 tests successful      ]
[      0 tests failed          ]
```

Figure 4.14: JUnit test for building a graph data type


```

.
+-- JUnit Jupiter [OK]
'-- JUnit Vintage [OK]
    '-- DeviceDistributionTest [OK]
        +-- validDeviceCategoryDistributionTest [OK]
        +-- invalidDeviceCategoryDistributionTest [OK]
        +-- validDeviceNameDistributionTest [OK]
        +-- validDeviceTypeDistributionTest [OK]
        +-- invalidDeviceNameDistributionTest [OK]
        '-- invalidDeviceTypeDistributionTest [OK]

Test run finished after 155 ms
[      3 containers found      ]
[      0 containers skipped    ]
[      3 containers started    ]
[      0 containers aborted    ]
[      3 containers successful  ]
[      0 containers failed     ]
[      6 tests found           ]
[      0 tests skipped         ]
[      6 tests started         ]
[      0 tests aborted         ]
[      6 tests successful      ]
[      0 tests failed          ]

```

Figure 4.15: JUnit test for calculating device distribution

4.3.5 Sprint Task Completion

- Loading the ESHD: Done
- Building a graph data type: Done
- Graph visualisation: Done
- Calculating device distribution: Done
- All tasks have now been completed.

5 Conclusion

The system is now able to:

- categorise users
- account login for encost user
- have feature options for community and encost-verified users
- load the encost smart homes dataset
- categorise smart home devices
- use graph data type for device distribution and graph visualisation
- graph visualisation without formatting
- calculating device distribution and display it as output