PES UNIVERSITY, BANGALORE

Department of Computer Science and Engineering

Software Engineering Project Report

Crowd Management at POS

Prepared by: **TEAM 09**

Brainiacs

Jeffrey S Varghese -PES1UG20CS651

Basava Prabhu-PES1UG20CS631

Pavan Kumar Nuthi-PES1UG20CS670

Ishita Bharadwaj-PES1UG20CS648

Software Engineering Project

Crowd Management at POS

Contents:

1a) SE concepts/principles/methods used (as explained below) to complete the project along with relevant artifacts created

Software Life Cycle Model: **Iterative**

We followed the Iterative model approach to implement/ develop our product.
Initial implementation starts from a skeleton of the product by simulating a prototype of our product using Arduino board and IR sensors.
We came up with the prototype for one terminal working independently, and later created more terminals and integrated them to work in cohesion and effectively manage the crowd at POS terminals. This makes use of successive refinements over the iterations. Using the Iterative SDLC model helped us identify requirements and visualize the solution. This also provided support for risk mitigation, incremental investment, reduced rework and feature creeps and lastly, increased customer engagement.
Despite the above advantages, each phase was rigid with overlaps in implementation.

Agile methodology: **SCRUM**

Scrum is a framework of rules, roles, events, and artifacts used to implement Agile projects. We developed our product using an iterative approach, consisting of sprints that typically lasted one to four weeks. This approach ensured that our team delivered a version of the product regularly. We prepared a burndown chart that shows the amount of work that has been completed in an epic or sprint, and the total work remaining. We used it to predict our team's likelihood of completing their work in the time available and being aware of any scope creep that occurs.

1b)SRS document preparation

i)Proposal of project

Retailers who are unable to mitigate challenges that come in the way to provide a seamless billing experience to their customers are more likely to suffer. They can lose customers and thus sales. The intended audience includes the customer waiting in line for their turn at the billing counters and also the retail store management to detect overcrowding and to simply fast track the billing processes so that the customer need not wait for long. Crowd monitoring systems (CMSs) provide a state-of-the-art solution to manage large crowds objectively. In recent years, researchers have discovered the potential of CMSs for crowd behavior research and have started to leverage CMSs to derive new insights regarding crowd movement behavior. The crowd state (i.e., walking velocity, density, and flow rate) at a retail outlet can be effectively determined using a comprehensive CMS.

ii)Feasibility

Economical Feasibility

This study is carried out to check the economic impact that the system will have on the retail domain as a whole. The developed system will be well within the budget and this can be achieved by using technologies that are freely available. Only some customized products such as Arduino UNO, IR sensor, jumper wires, LCD display(optional) may be purchased.

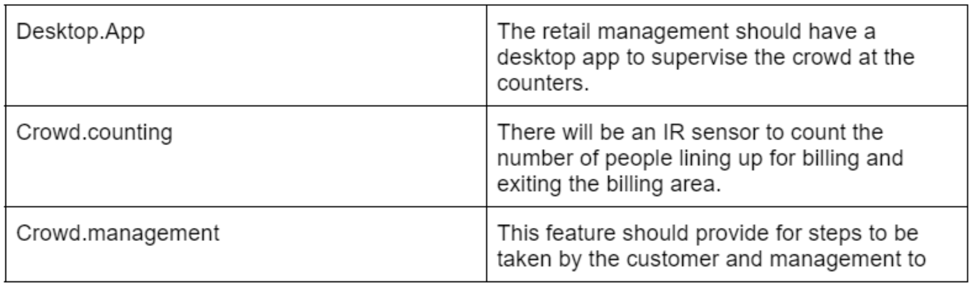
Technical Feasibility

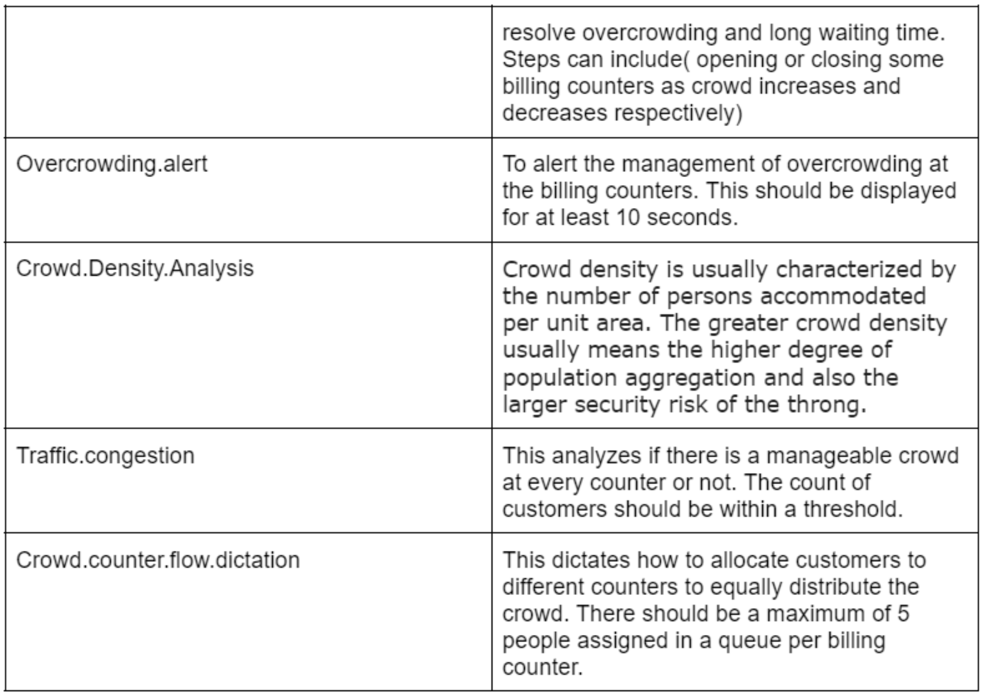
This study is carried out to check the technical feasibility, that is, the technical requirements of the system. The developed system has a modest requirement of Arduino UNO, IR sensors, jumper wires, LCD display(optional), as only minimal or null changes are required for implementing this system.

Social Feasibility

The aspect of study is to check the level of acceptance of the system by the customers and the management. The customer must not feel threatened by the system, instead must accept it as a necessity and be redirected to another counter/queue if the current counter is running on it’s maximum capacity .

iii)Functional Requirements





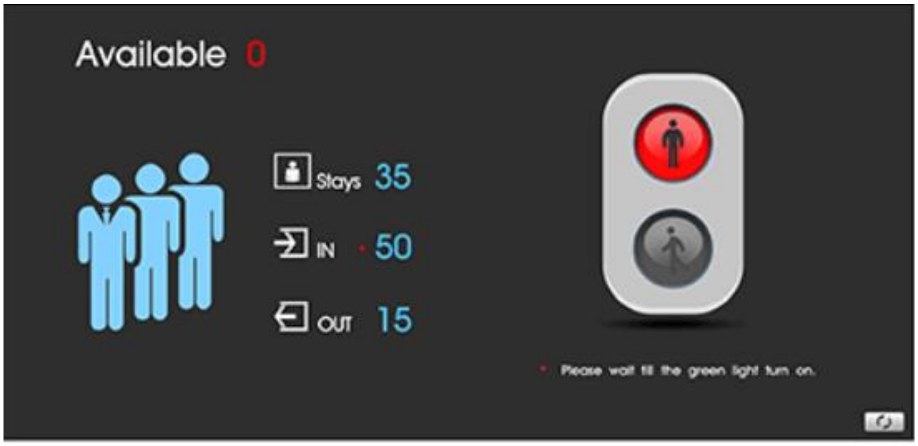
Desktop.App	The retail management should have a desktop app to supervise the crowd at the counters.
Crowd.counting	There will be an IR sensor to count the number of people lining up for billing and exiting the billing area.
Crowd.management	This feature should provide for steps to be taken by the customer and management to resolve overcrowding and long waiting time. Steps can include(opening or closing some billing counters as crowd increases and decreases respectively)
Overcrowding.alert	To alert the management of overcrowding at the billing counters. This should be displayed for at least 10 seconds.
Crowd.Density.Analysis	Crowd density is usually characterized by the number of persons accommodated per unit area. The greater crowd density usually means the higher degree of population aggregation and also the larger security risk of the throng.
Traffic.congestion	This analyzes if there is a manageable crowd at every counter or not. The count of customers should be within a threshold.
Crowd.counter.flow.dictation	This dictates how to allocate customers to different counters to equally distribute the crowd. There should be a maximum of 5 people assigned in a queue per billing counter.

iv)External interface Requirements

- **<u>User Interface</u>**

The software must display the following data to the user/retailer.

a. Number of customers waiting for billing outside the queue.
b. Number of customers who are present in the queue.
c. Threshold of maximum number of customers allowed .
d. Estimated waiting time for customers outside.
e. Stop or Go signal for customers outside.
f. Similar to the one shown below:



Please wait till the green light turn on.

- **Hardware Interface**

Works with any of the following operating systems:

- Windows 32/64 bit.
- MacOS
- Linux

Additional IoT based sensors:

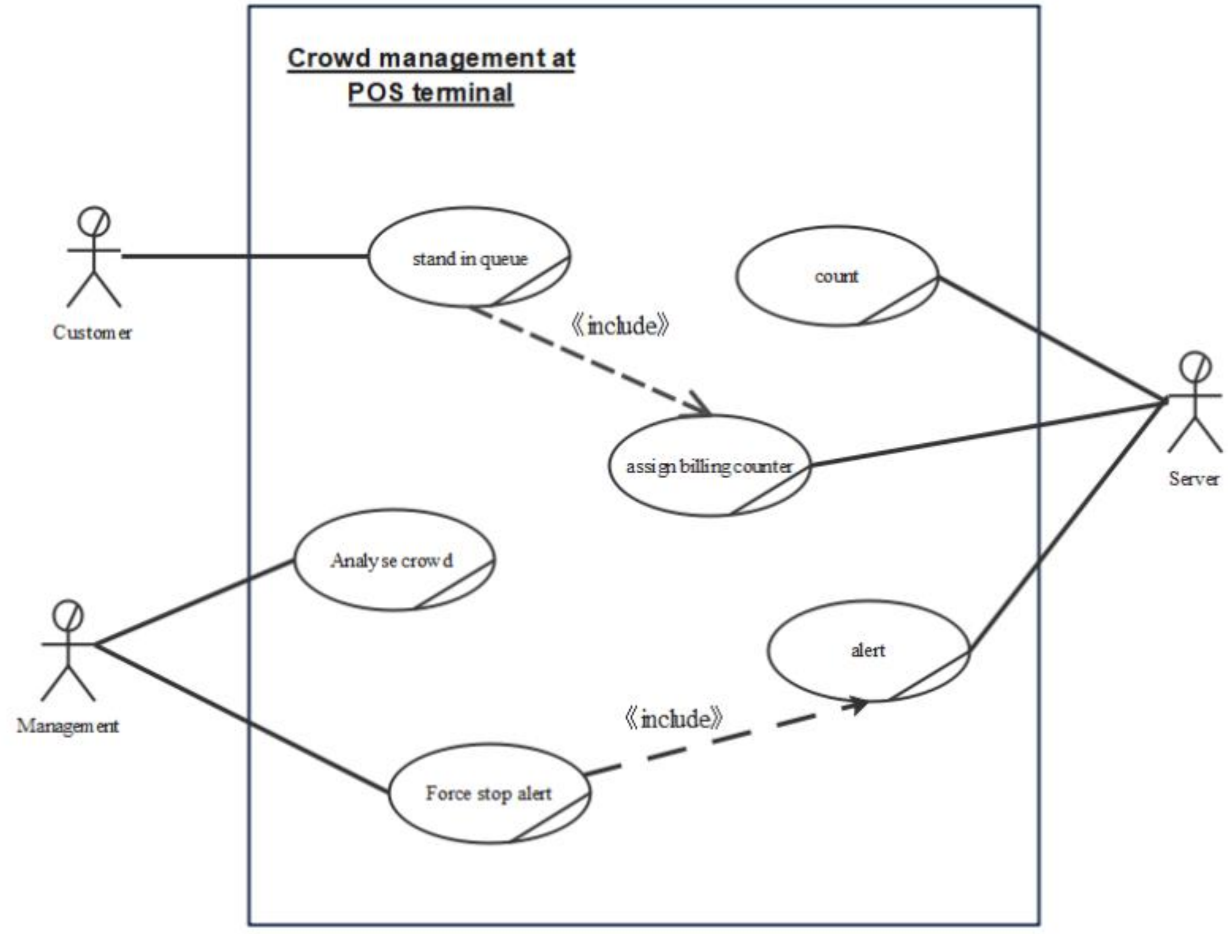
Depending upon the type of solution which is feasible to implement, we can use the following sensors to detect the presence of a crowd/customers.

Heat sensor:

Pressure/Force Sensor:

IC chips attached to shopping carts:

v)Analysis models



vi)Non-functional requirements

- **Performance requirements**
  PE-1:Application Response Time
  PE-2:Database Response Time
  PE-3:Scalability
- **Security requirements**

SE-1:The users/store managers shall be required to log in to the Crowd Management System to perform any operations of a given store.

SE-2:All network transactions that involve personally identifiable information shall be encrypted.

vii)Change management System

Changes to this system can be made only under the following conditions:

- Bugs reported
- Loophole is detected
- System fails new test cases

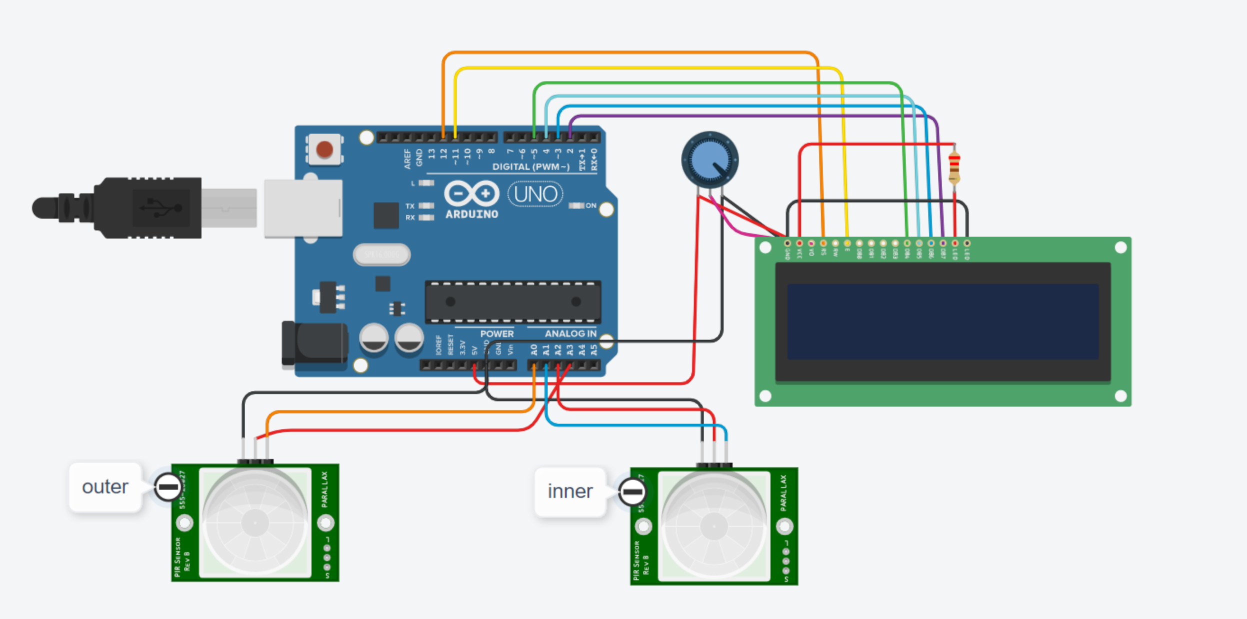
The sequential steps for the change control and management of the system is as follows:

- Creating a request for change
- Reviewing and assessing a request for change
- Planning the change
- Testing the change
- Creating a change proposal
- Implementing changes
- Reviewing change performance
- Closing the process

1b) User Story

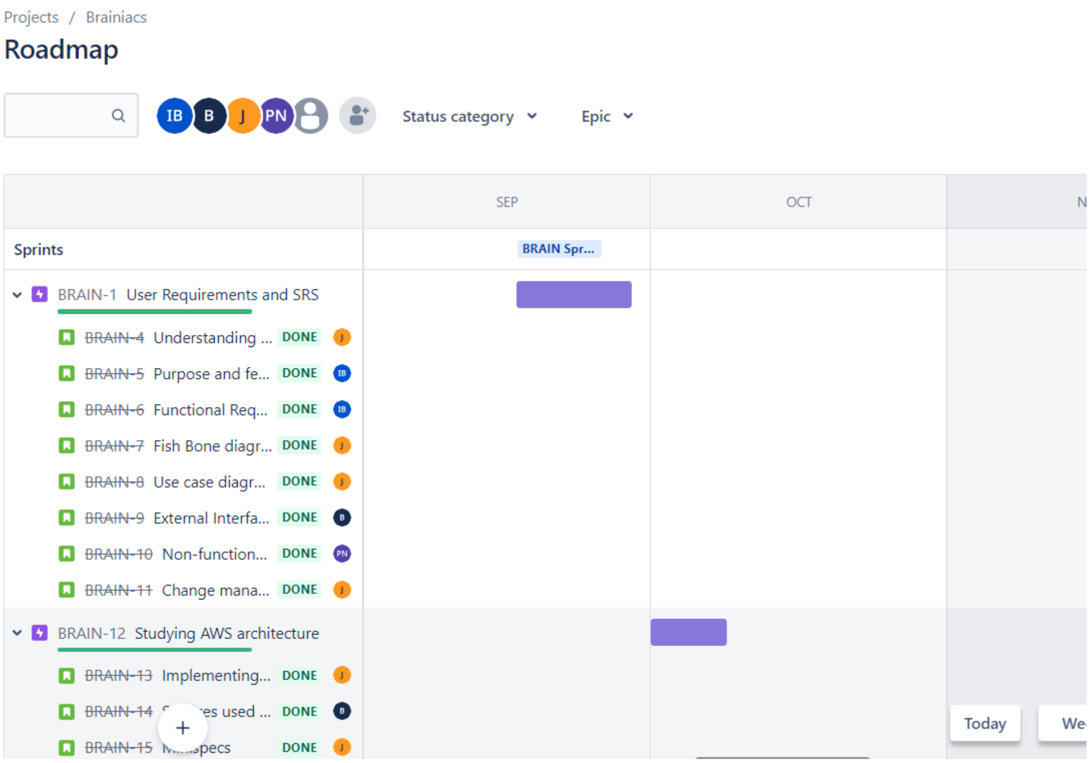
I was invited to a birthday party, so I thought I'd buy something nice for her 20th birthday. I went to Walmart to buy my friend a birthday gift. After spending some time selecting the gift, I was finally able to pick a good gift that my friend would like. I then proceeded to the POS to pay for the gift. Unfortunately the store was very crowded on that day as it was a weekend. I had to wait in the queue for an unreasonable amount of time, which made me miss the birthday party**.Only if the crowd at the POS was efficiently handled** i would have made it on time for the cake cutting.

1c) Prototype built



We built a prototype simulation consisting of an Arduino board, 2 IR sensors, an LCD display and connecting wires. There were 3 wires arising from each IR sensor - a power, a signal and a ground wire which were connected to the D ports(D0- D19) on the arduino board.
An LCD display was used for displaying the count of persons at one terminal at any given point of time. The outer IR sensor was used to keep track of customers entering the terminal and the inner IR sensor was used for tracking customers exiting the terminal. A delay of 500-1000 microseconds was given for proper updation of the counter value.

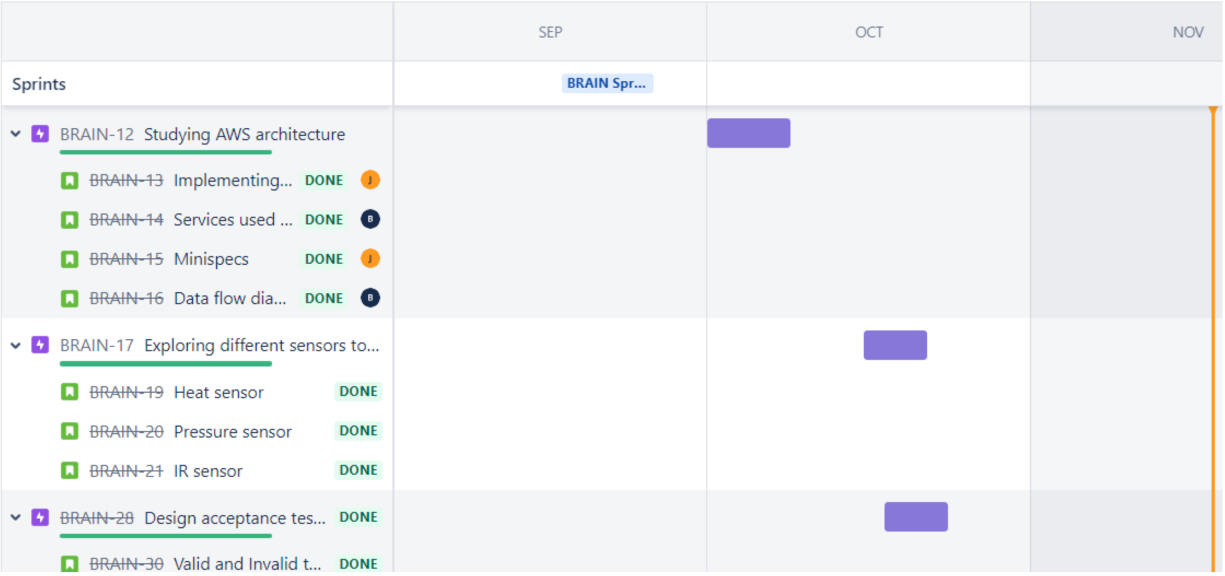
1d) Work Breakdown Structure (WBS) using JIRA



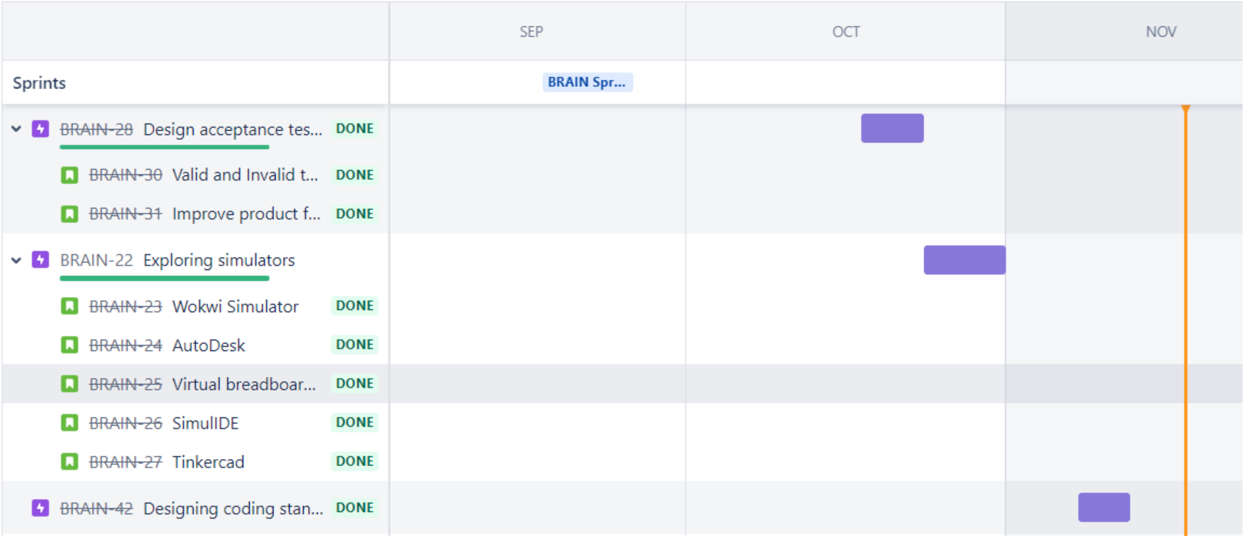
Projects / Brainiacs

Roadmap

Sprints		
BRAIN-1 User Requirements and SRS		
~~BRAIN-4~~ Understanding ...	DONE	J
~~BRAIN-5~~ Purpose and fe...	DONE	IB
~~BRAIN-6~~ Functional Req...	DONE	IB
~~BRAIN-7~~ Fish Bone diagr...	DONE	J
~~BRAIN-8~~ Use case diagr...	DONE	J
~~BRAIN-9~~ External Interfa...	DONE	B
~~BRAIN-10~~ Non-function...	DONE	PN
~~BRAIN-11~~ Change mana...	DONE	J
BRAIN-12 Studying AWS architecture		
~~BRAIN-13~~ Implementing...	DONE	J
~~BRAIN-14~~ es used ...	DONE	B
~~BRAIN-15~~ M...specs	DONE	J



Sprints		
BRAIN-12 Studying AWS architecture		
~~BRAIN-13~~ Implementing...	DONE	J
~~BRAIN-14~~ Services used ...	DONE	B
~~BRAIN-15~~ Minispecs	DONE	J
~~BRAIN-16~~ Data flow dia...	DONE	B
BRAIN-17 Exploring different sensors to...		
~~BRAIN-19~~ Heat sensor	DONE	
~~BRAIN-20~~ Pressure sensor	DONE	
~~BRAIN-21~~ IR sensor	DONE	
~~BRAIN-28~~ Design acceptance tes...	DONE	
~~BRAIN-30~~ Valid and Invalid t...	DONE	



Sprints

~~BRAIN-28~~ Design acceptance tes... DONE

~~BRAIN-30~~ Valid and Invalid t... DONE

~~BRAIN-31~~ Improve product f... DONE

BRAIN-22 Exploring simulators

~~BRAIN-23~~ Wokwi Simulator DONE

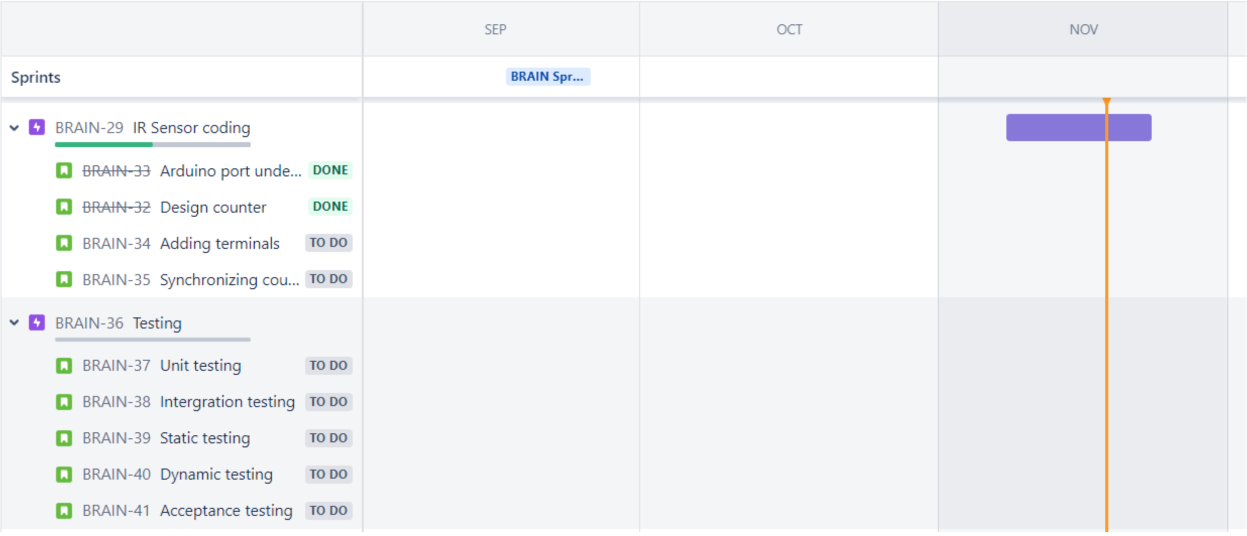
~~BRAIN-24~~ AutoDesk DONE

~~BRAIN-25~~ Virtual breadboar... DONE

~~BRAIN-26~~ SimulIDE DONE

~~BRAIN-27~~ Tinkercad DONE

~~BRAIN-42~~ Designing coding stan... DONE



Sprints

BRAIN-29 IR Sensor coding

~~BRAIN-33~~ Arduino port unde... DONE

~~BRAIN-32~~ Design counter DONE

BRAIN-34 Adding terminals TO DO

BRAIN-35 Synchronizing cou... TO DO

BRAIN-36 Testing

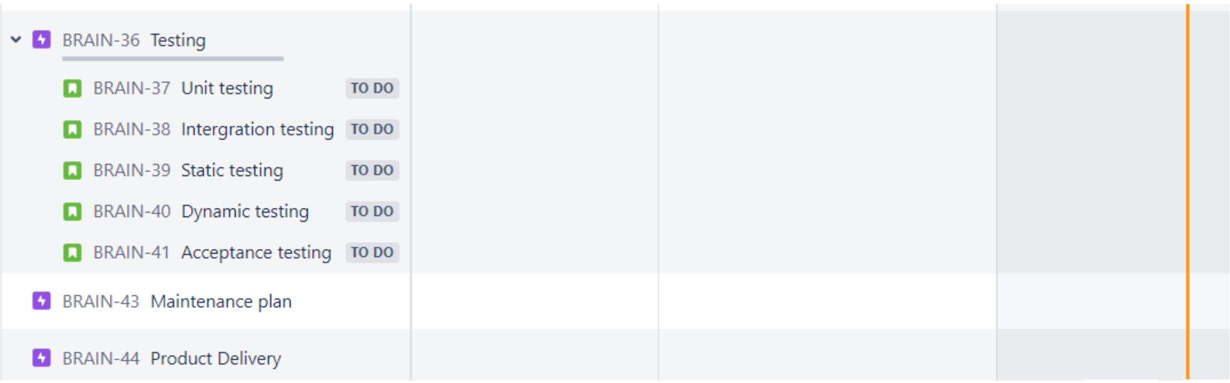
BRAIN-37 Unit testing TO DO

BRAIN-38 Intergration testing TO DO

BRAIN-39 Static testing TO DO

BRAIN-40 Dynamic testing TO DO

BRAIN-41 Acceptance testing TO DO



BRAIN-36 Testing

BRAIN-37 Unit testing TO DO

BRAIN-38 Intergration testing TO DO

BRAIN-39 Static testing TO DO

BRAIN-40 Dynamic testing TO DO

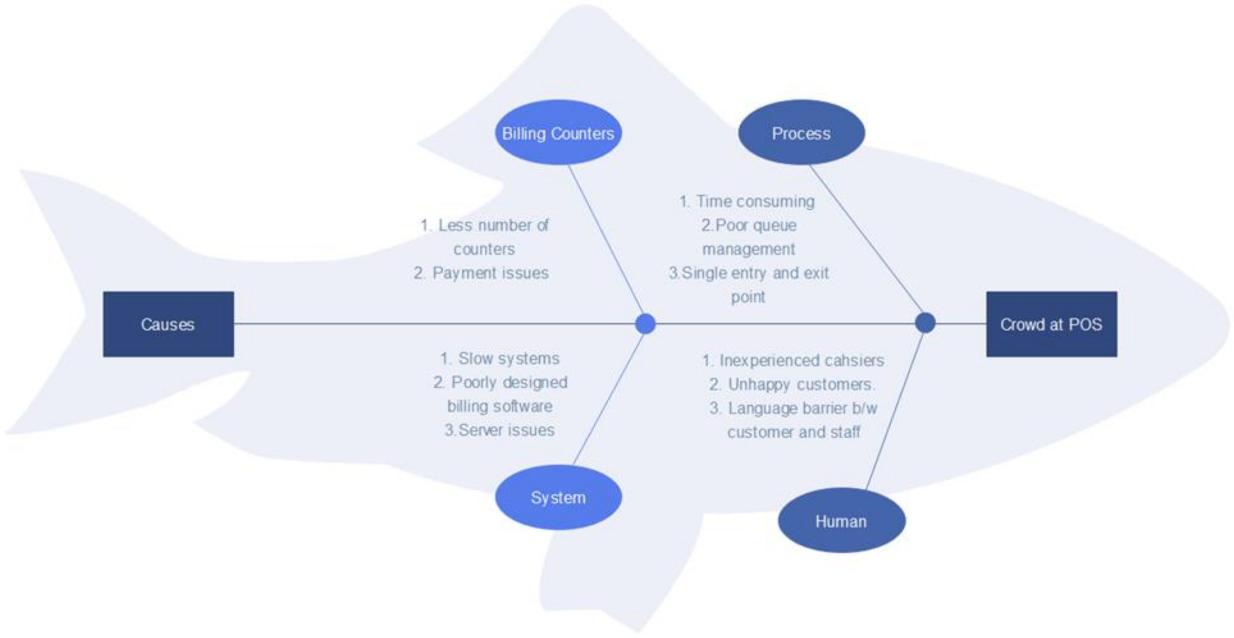
BRAIN-41 Acceptance testing TO DO

BRAIN-43 Maintenance plan

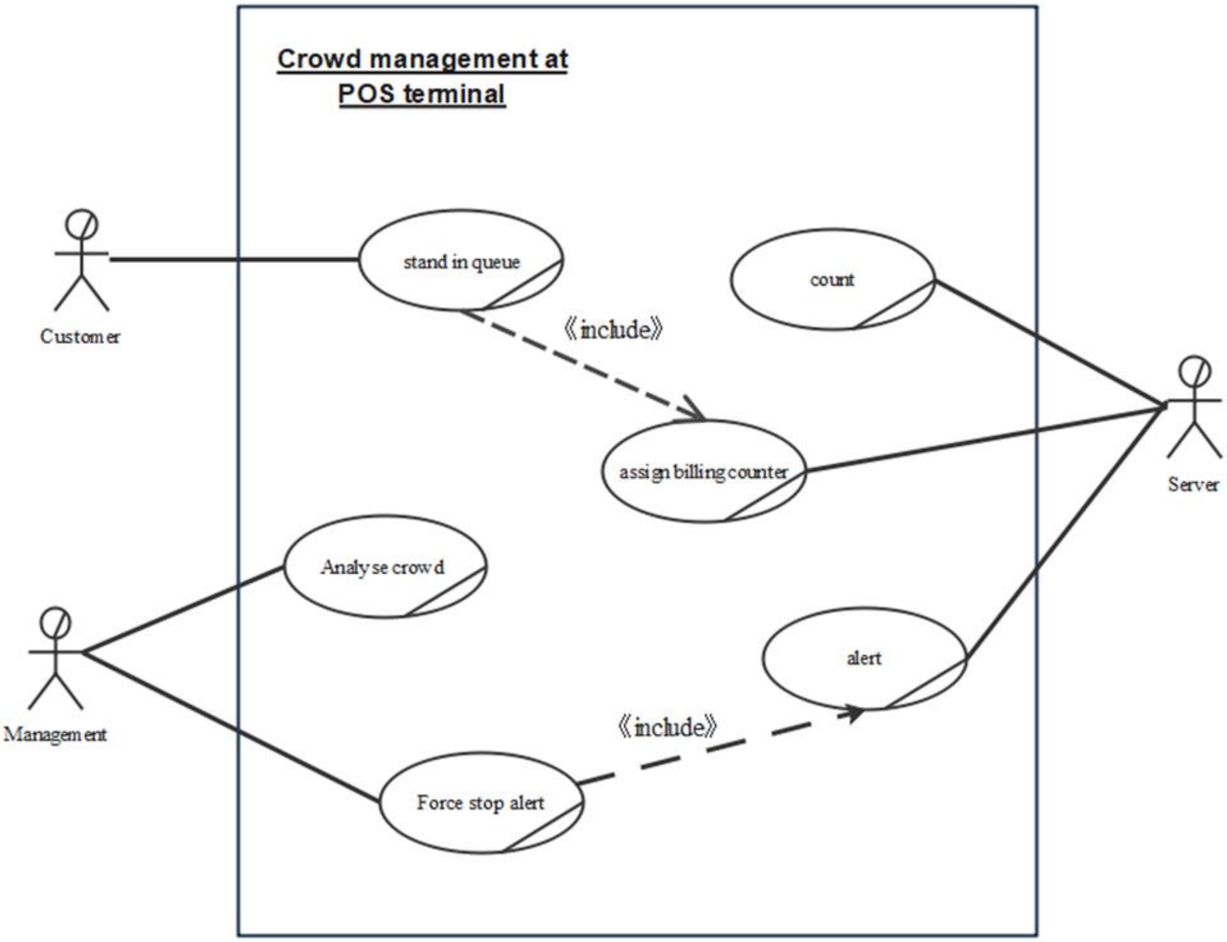
BRAIN-44 Product Delivery

1e) Design for the project selected – Show Architectural diagram, Design Diagram, UML diagrams, DFD, etc

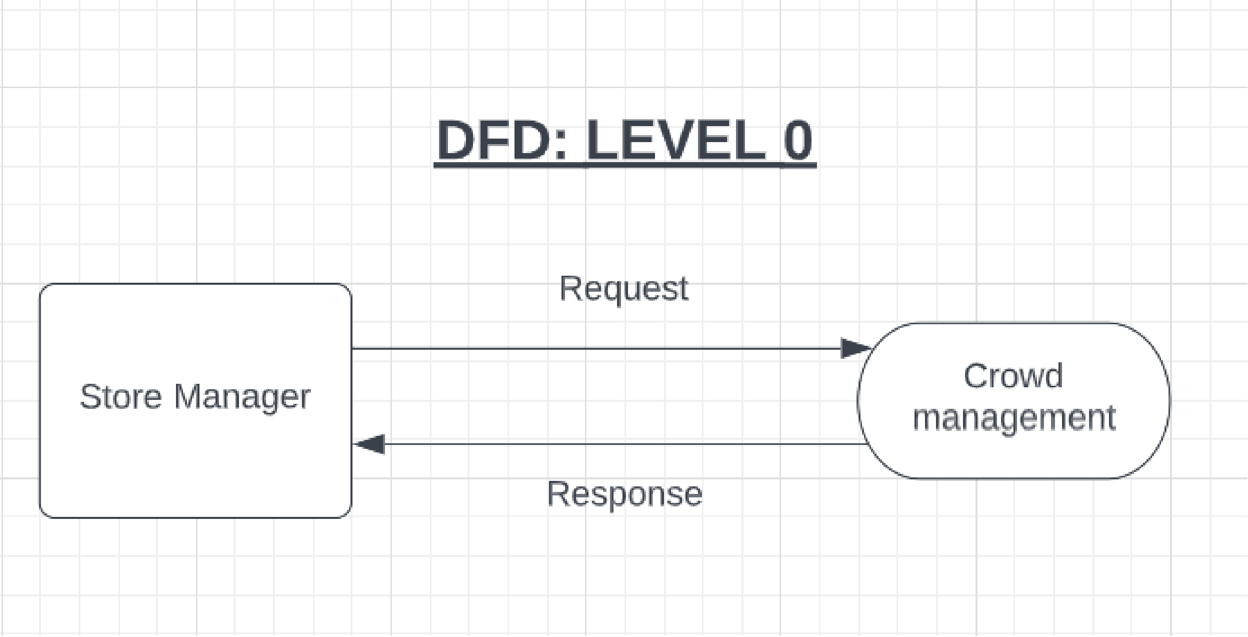
Fish Bone Diagram:

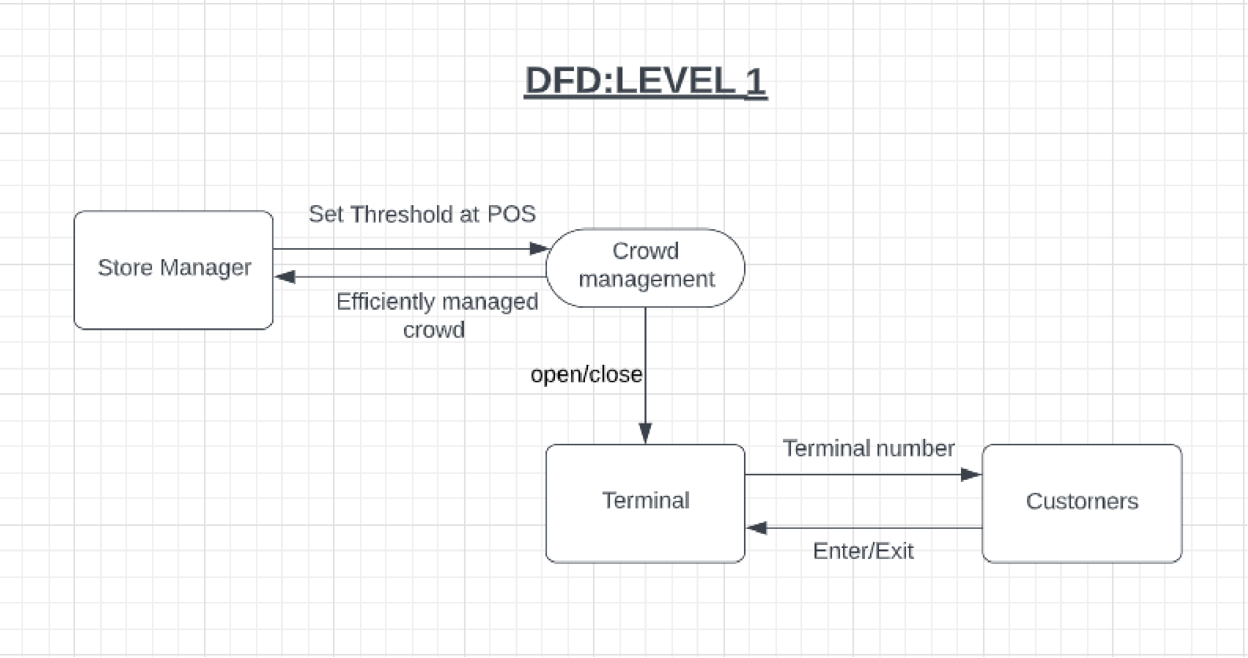


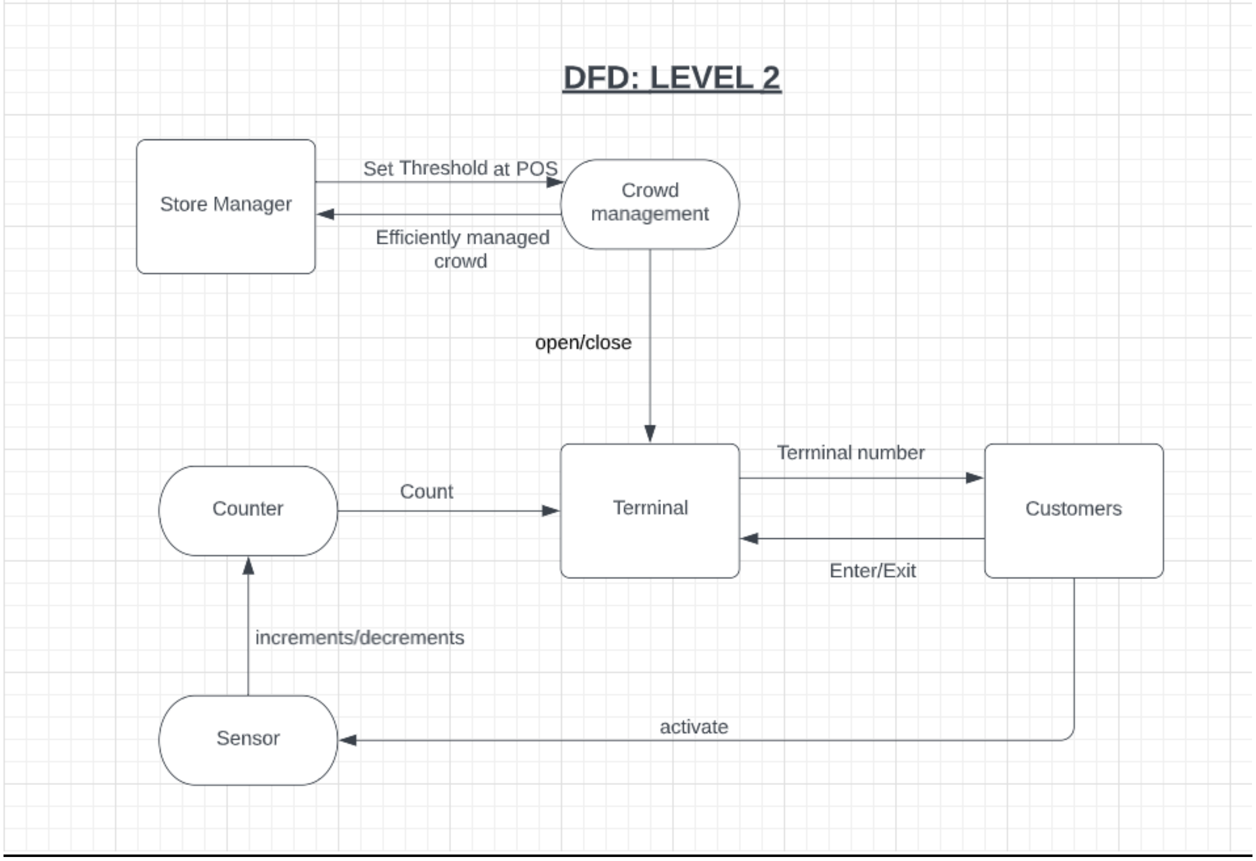
Use Case Diagram:



Data Flow Diagram:






DFD: LEVEL 2
Store Manager
Set Threshold at POS
Crowd management
Efficiently managed crowd
open/close
Counter
Count
Terminal
Terminal number
Customers
Enter/Exit
increments/decrements
Sensor
activate

1f) Explain Coding practices/standards used

- Focus on code readability and maintainability.
    - Write as few lines as possible
    - Use appropriate naming conventions
    - Segments blocks
- Reduce complexity us much as possible
- Divide the code into components to maximize reusability
- Documentation is maintained and codes are well commented
- Don’t use a single identifier for multiple purposes.
- Turn daily backup’s into an instinct
- Formalize Exception handling
    - Use try - catch

1g) SCM environment used like GitHub and any SCM concepts such as Branch Management and Versioning used for the project.

Software Configuration Management (SCM) is a software engineering discipline consisting of standard processes and techniques often used by organizations to manage the changes introduced into its software products. SCM helps in identifying individual elements and configurations,tracking changes, and version selection, control, and baselining.

SCM is a Process to systematically organize, manage and control changes in documents, code and other entities that constitute a software product.

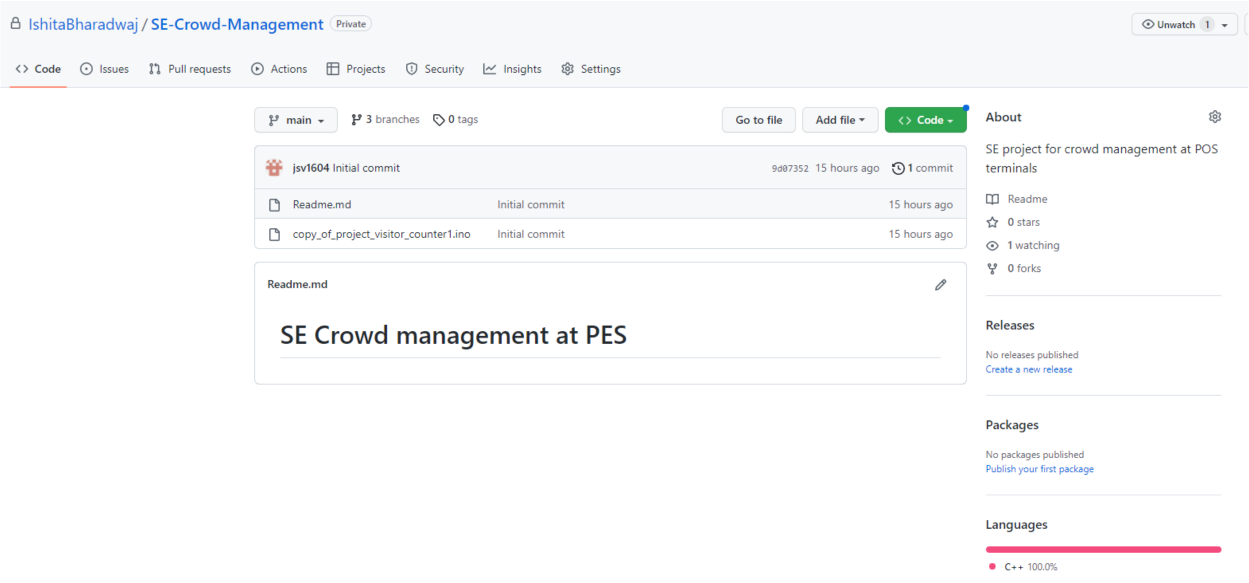
Github- We decided to use Github for managing, organizing our project for the following reasons:

- Git is free and open source software for distributed version control.
- Tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development.
- Access control
- Bug tracking
- Software feature requests
- Task management

Using Github- The following pictures show how we leveraged the features of github to create versions for our project.

Github Link to Project:

Link to github project



IshitaBharadwaj / SE-Crowd-Management Private

3 branches 0 tags

jsv1604 Initial commit 9d07352 15 hours ago 1 commit

Readme.md	Initial commit	15 hours ago
copy_of_project_visitor_counter1.ino	Initial commit	15 hours ago

Readme.md

SE Crowd management at PES

About

SE project for crowd management at POS terminals

Readme
0 stars
1 watching
0 forks

Releases

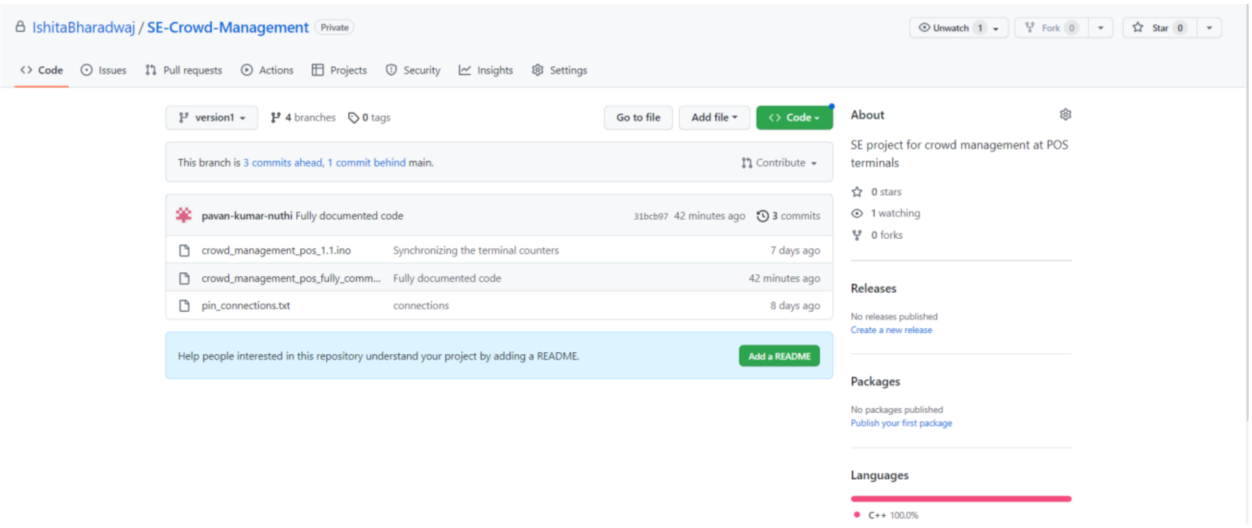
No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

● C++ 100.0%



IshitaBharadwaj / SE-Crowd-Management Private

4 branches 0 tags

This branch is 3 commits ahead, 1 commit behind main.

pavan-kumar-nuthi Fully documented code 31bcb97 42 minutes ago 3 commits

crowd_management_pos_1.1.ino	Synchronizing the terminal counters	7 days ago
crowd_management_pos_fully_comm...	Fully documented code	42 minutes ago
pin_connections.txt	connections	8 days ago

Help people interested in this repository understand your project by adding a README.

About

SE project for crowd management at POS terminals

0 stars
1 watching
0 forks

Releases

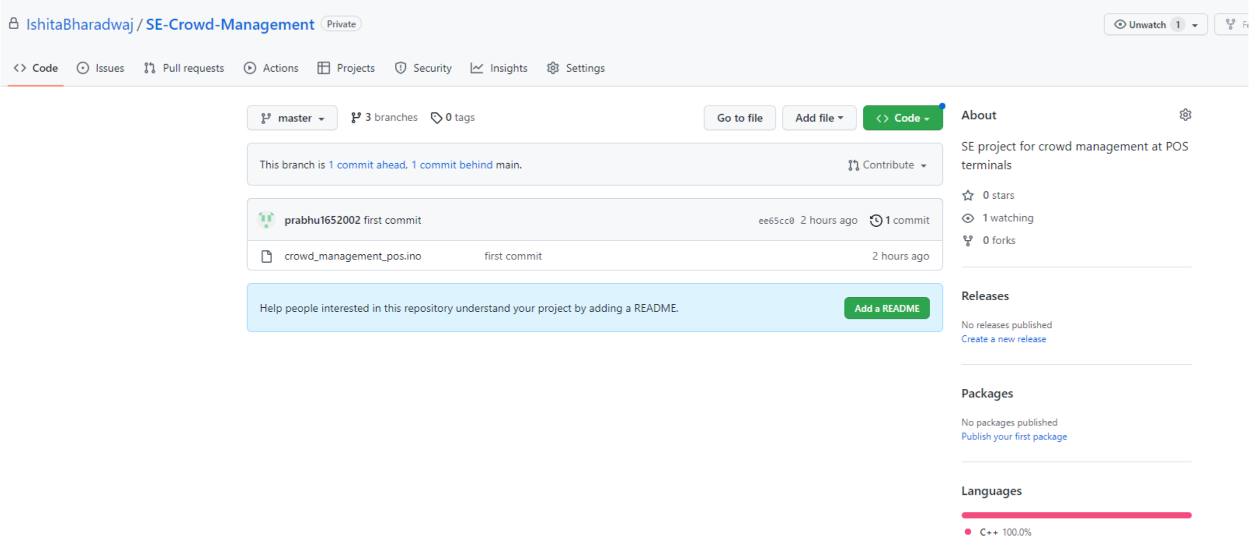
No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

● C++ 100.0%



IshitaBharadwaj / SE-Crowd-Management Private

3 branches 0 tags

This branch is 1 commit ahead, 1 commit behind main.

prabhu1652002 first commit ee65cc0 2 hours ago 1 commit

crowd_management_pos.ino	first commit	2 hours ago

Help people interested in this repository understand your project by adding a README.

About

SE project for crowd management at POS terminals

0 stars
1 watching
0 forks

Releases

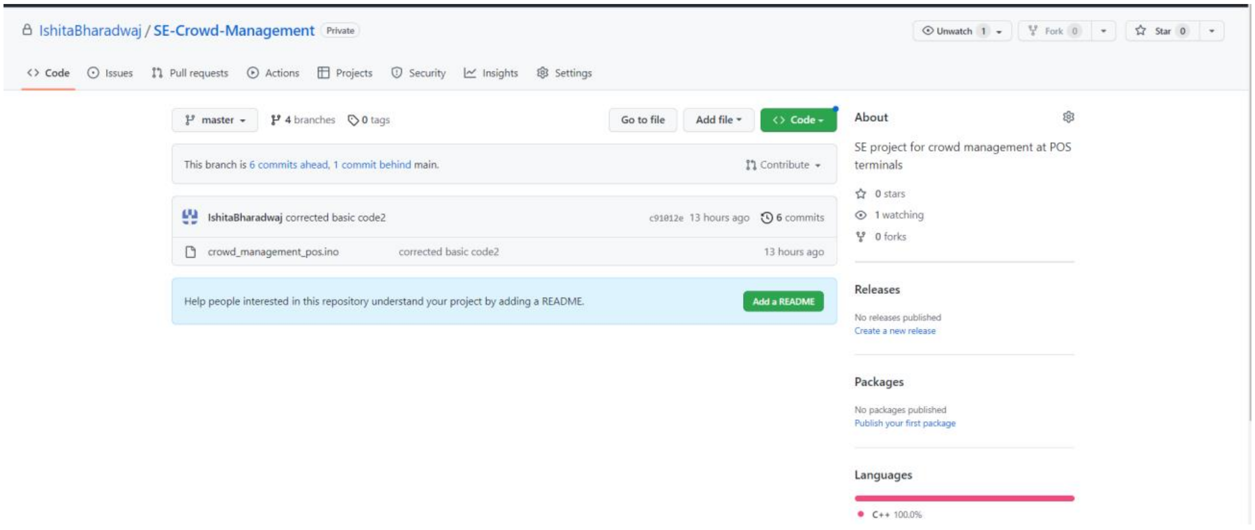
No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

● C++ 100.0%



This branch is 6 commits ahead, 1 commit behind main.

IshitaBharadwaj corrected basic code2 c91012e 13 hours ago 6 commits

crowd_management_pos.ino	corrected basic code2	13 hours ago

Help people interested in this repository understand your project by adding a README.

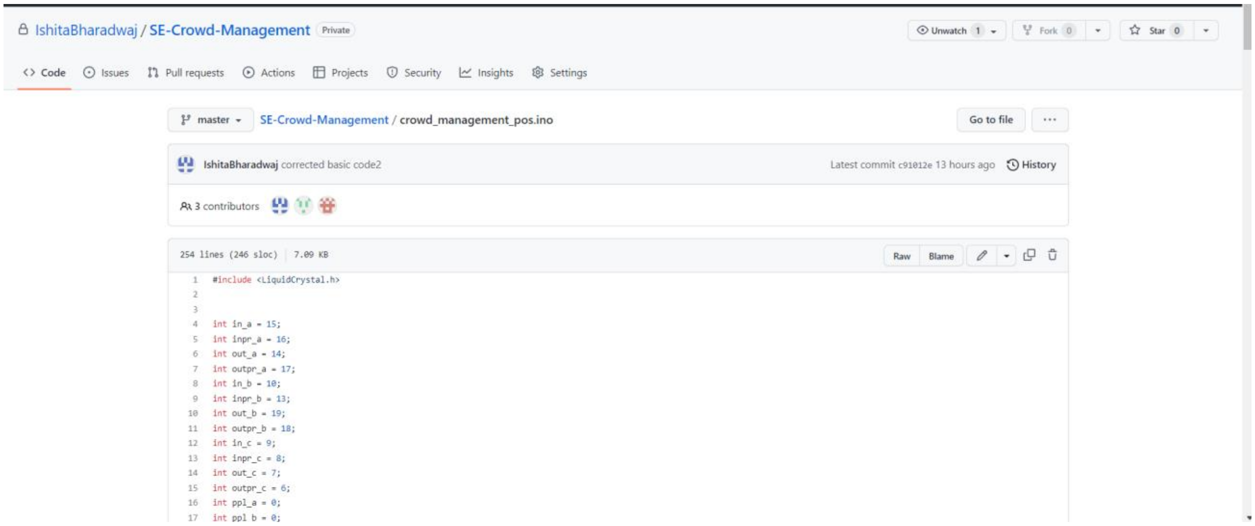
SE project for crowd management at POS terminals

0 stars
1 watching
0 forks

No releases published
Create a new release

No packages published
Publish your first package

C++ 100.0%



SE-Crowd-Management / crowd_management_pos.ino

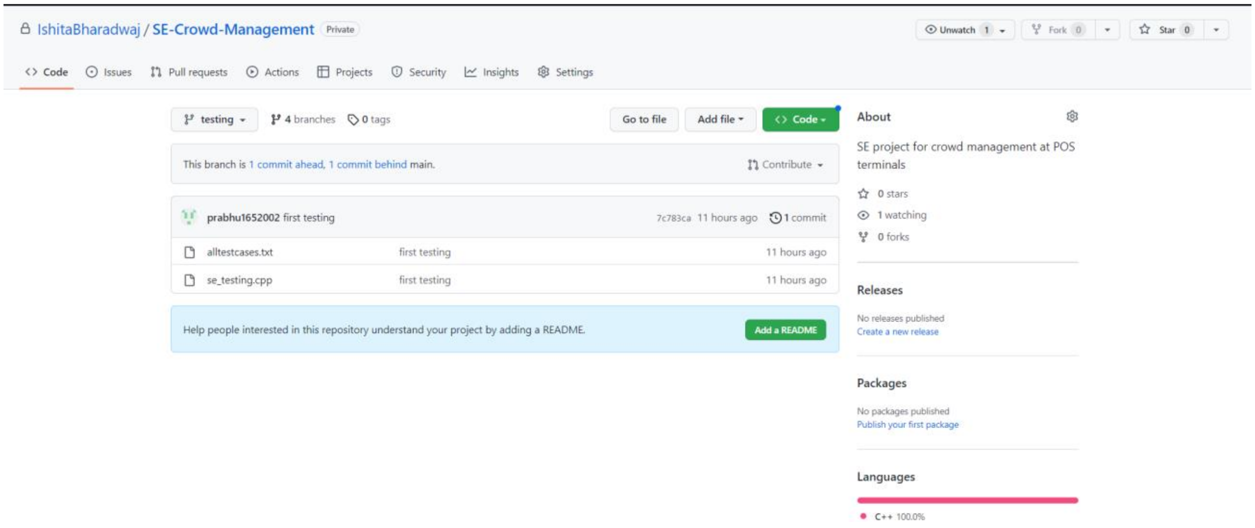
IshitaBharadwaj corrected basic code2 Latest commit c91012e 13 hours ago

3 contributors

254 lines (246 sloc) | 7.09 KB

```
#include <LiquidCrystal.h>


int in_a = 15;
int inpr_a = 16;
int out_a = 14;
int outpr_a = 17;
int in_b = 10;
int inpr_b = 13;
int out_b = 19;
int outpr_b = 18;
int in_c = 9;
int inpr_c = 8;
int out_c = 7;
int outpr_c = 6;
int ppl_a = 0;
int ppl b = 0;
```



This branch is 1 commit ahead, 1 commit behind main.

prabhu1652002 first testing 7c783ca 11 hours ago 1 commit

alltestcases.txt	first testing	11 hours ago
se_testing.cpp	first testing	11 hours ago

Help people interested in this repository understand your project by adding a README.

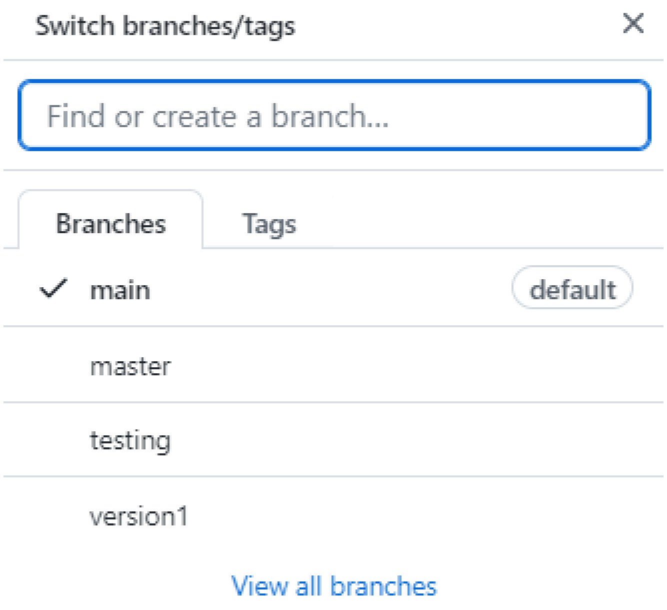
SE project for crowd management at POS terminals

0 stars
1 watching
0 forks

No releases published
Create a new release

No packages published
Publish your first package

C++ 100.0%

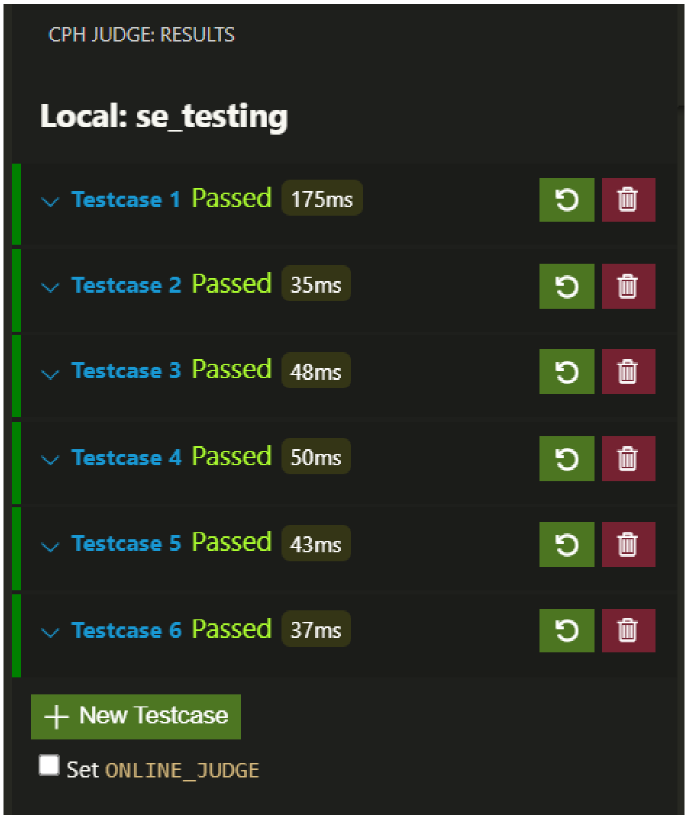


1h) Test strategy, test plan, test suite, test cases created

<u>Test strategy:</u>

- Unit testing:
  - Each counter was tested individually as and when they were created
- Integration testing
  - Once all components i.e. counters were implemented- synchronization among the counters were tested
- System testing
  - All boundary conditions were tested
  - Tested the entire system to check if all requirements are met
  - The test cases are shown below.
- Acceptance testing

- Final system was tested in a similar environment as our development site did not have the exact hardware environment in which the project has to be deployed at the customer’s site. Acceptance testing was done on Tkinter CAD simulating environment.
- Link for the same is given below
- https://www.tinkercad.com/things/0TPcagZJzr9-copy-of-project-visitor-counter/editel?sharecode=wpiWYT0ti3PIG66k9L30L5GLEihvwDnKIHzDk9_mxQM

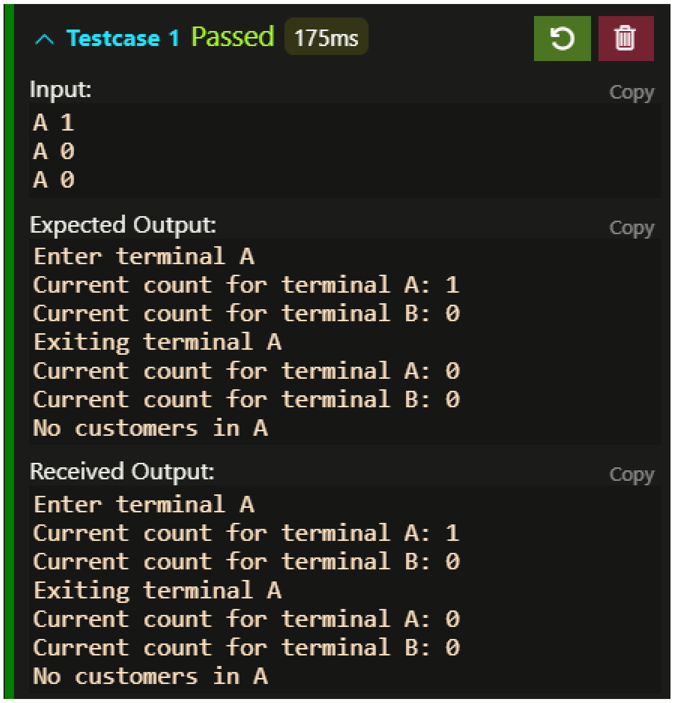


CPH JUDGE: RESULTS

Local: se_testing

```
Testcase 1 Passed 175ms
Testcase 2 Passed 35ms
Testcase 3 Passed 48ms
Testcase 4 Passed 50ms
Testcase 5 Passed 43ms
Testcase 6 Passed 37ms
```

☐ Set ONLINE_JUDGE



Input:

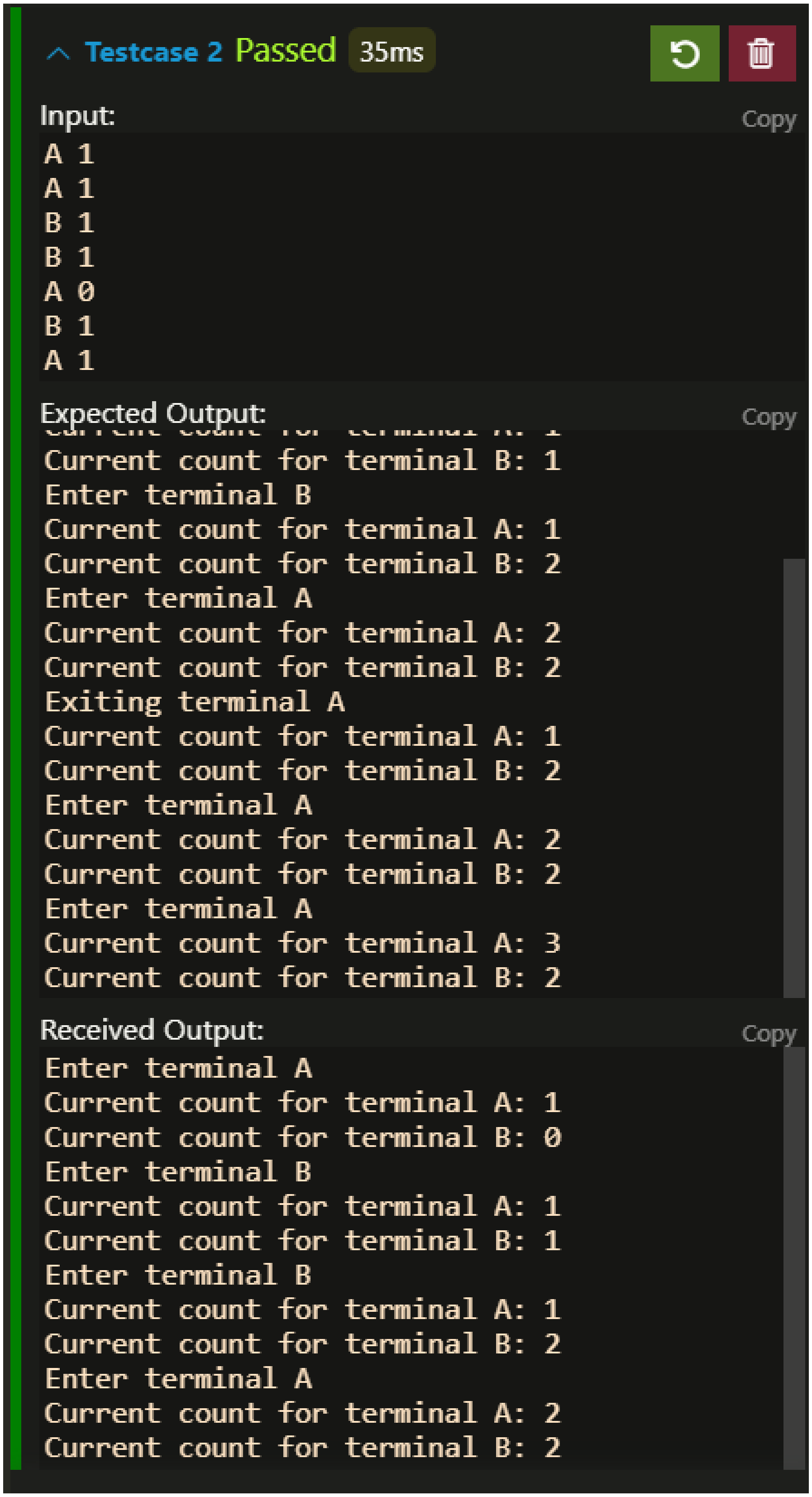
```
A 1
A 0
A 0
```

Expected Output:

```
Enter terminal A
Current count for terminal A: 1
Current count for terminal B: 0
Exiting terminal A
Current count for terminal A: 0
Current count for terminal B: 0
No customers in A
```

Received Output:

```
Enter terminal A
Current count for terminal A: 1
Current count for terminal B: 0
Exiting terminal A
Current count for terminal A: 0
Current count for terminal B: 0
No customers in A
```



Input:

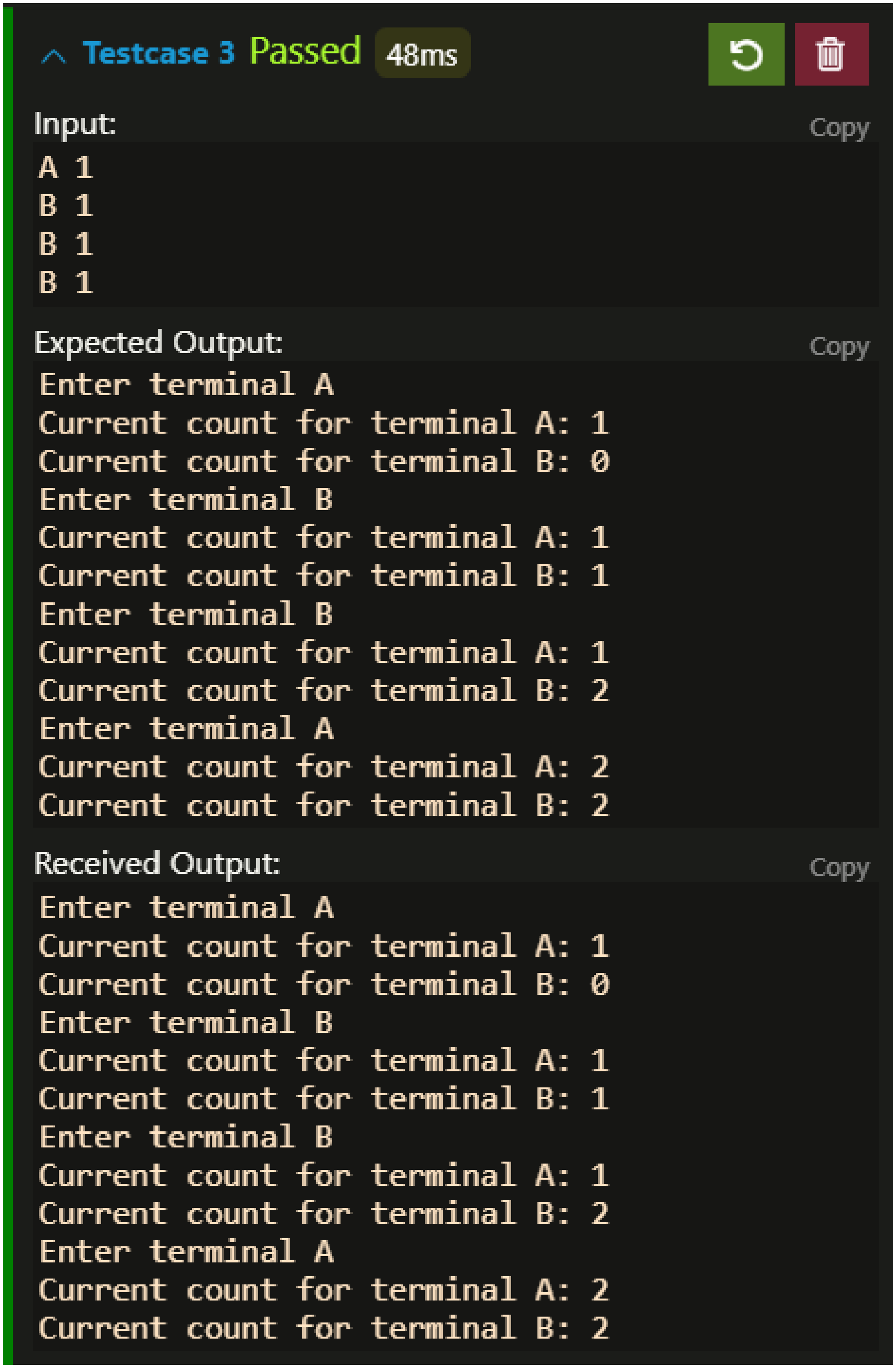
```
A 1
A 1
B 1
B 1
A 0
B 1
A 1
```

Expected Output:

```
Current count for terminal A: 1
Current count for terminal B: 1
Enter terminal B
Current count for terminal A: 1
Current count for terminal B: 2
Enter terminal A
Current count for terminal A: 2
Current count for terminal B: 2
Exiting terminal A
Current count for terminal A: 1
Current count for terminal B: 2
Enter terminal A
Current count for terminal A: 2
Current count for terminal B: 2
Enter terminal A
Current count for terminal A: 3
Current count for terminal B: 2
```

Received Output:

```
Enter terminal A
Current count for terminal A: 1
Current count for terminal B: 0
Enter terminal B
Current count for terminal A: 1
Current count for terminal B: 1
Enter terminal B
Current count for terminal A: 1
Current count for terminal B: 2
Enter terminal A
Current count for terminal A: 2
Current count for terminal B: 2
```



Input:

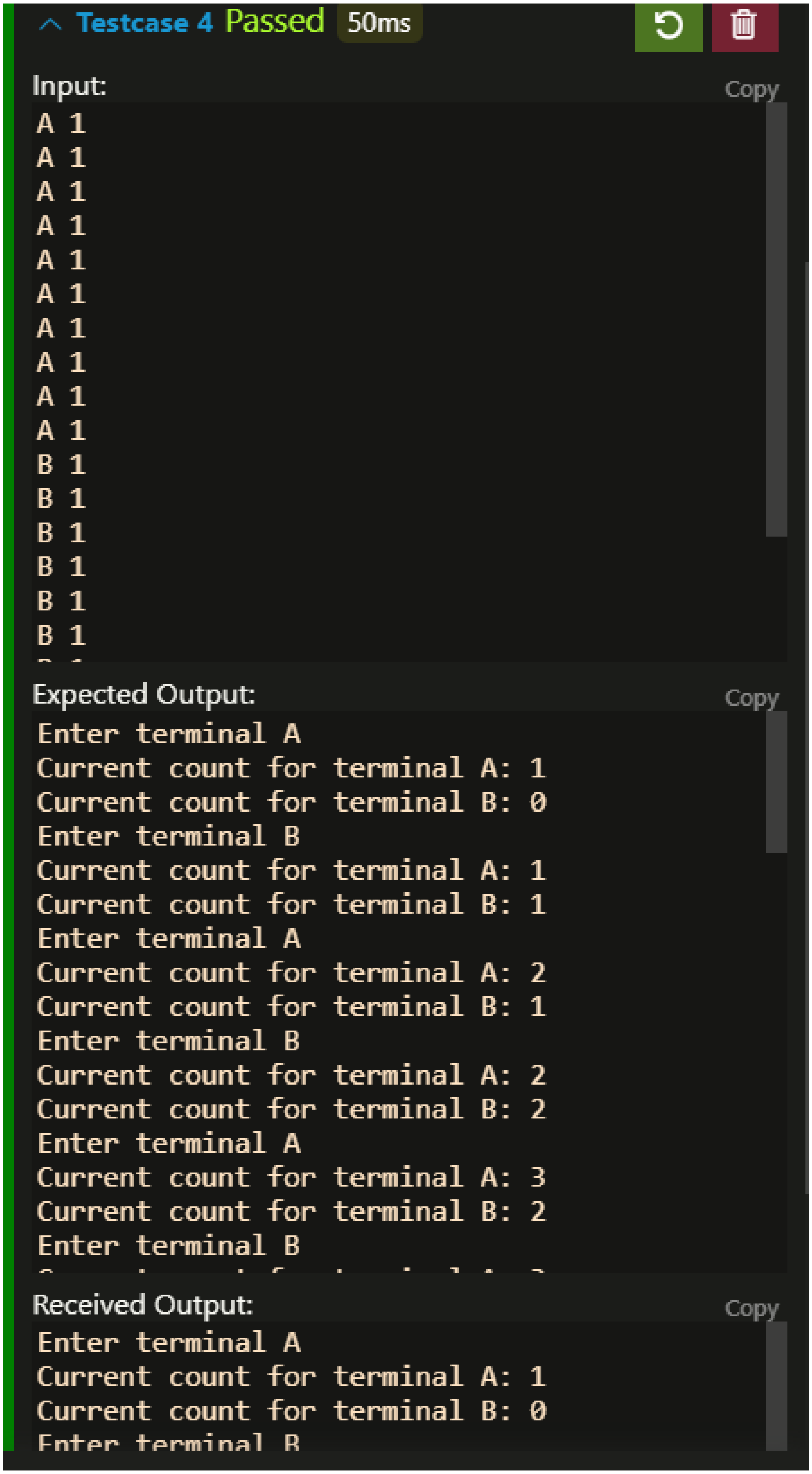
```
A 1
B 1
B 1
B 1
```

Expected Output:

```
Enter terminal A
Current count for terminal A: 1
Current count for terminal B: 0
Enter terminal B
Current count for terminal A: 1
Current count for terminal B: 1
Enter terminal B
Current count for terminal A: 1
Current count for terminal B: 2
Enter terminal A
Current count for terminal A: 2
Current count for terminal B: 2
```

Received Output:

```
Enter terminal A
Current count for terminal A: 1
Current count for terminal B: 0
Enter terminal B
Current count for terminal A: 1
Current count for terminal B: 1
Enter terminal B
Current count for terminal A: 1
Current count for terminal B: 2
Enter terminal A
Current count for terminal A: 2
Current count for terminal B: 2
```



Testcase 4 Passed 50ms

Input:

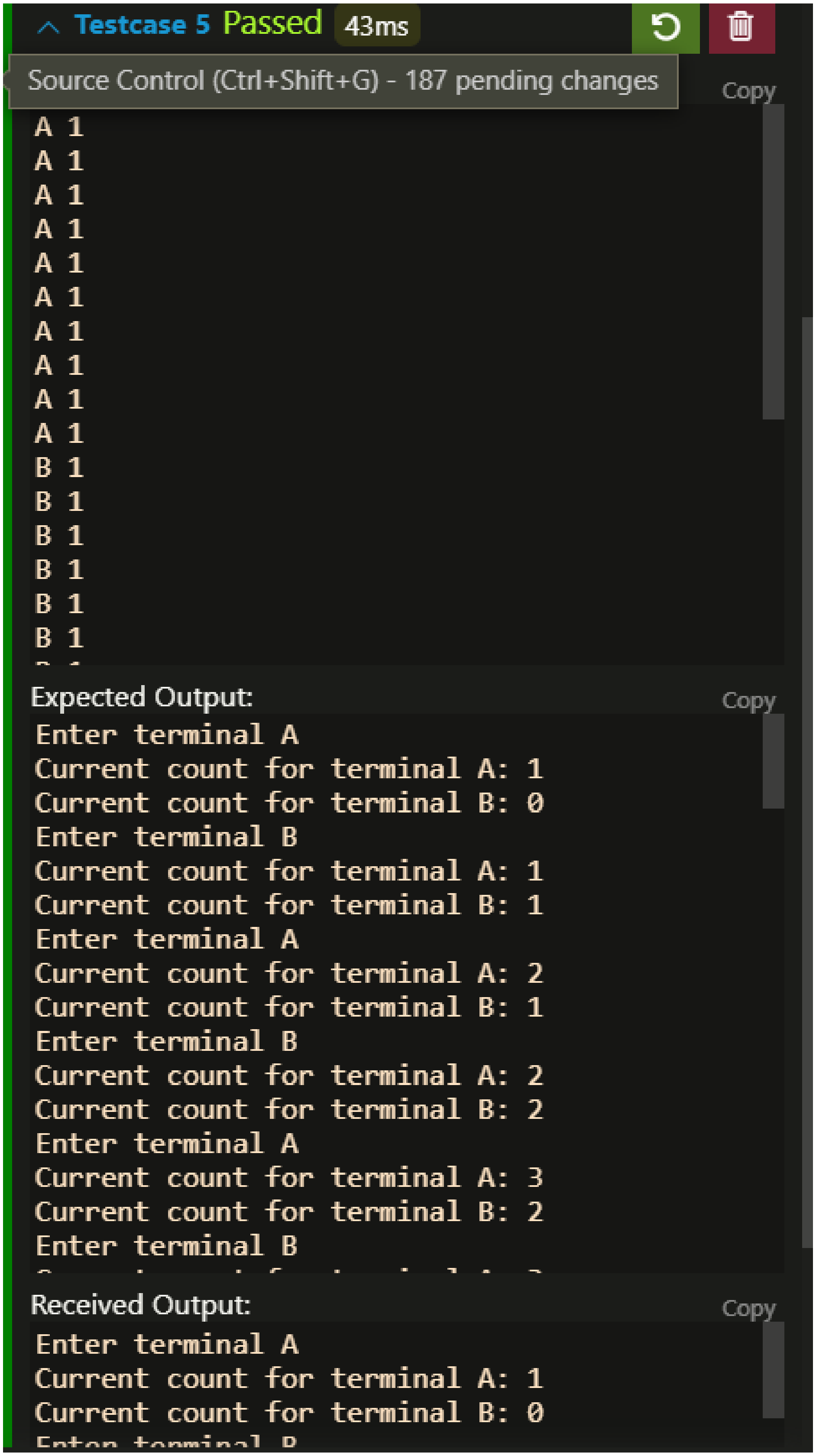
```
A 1
A 1
A 1
A 1
A 1
A 1
A 1
A 1
A 1
A 1
B 1
B 1
B 1
B 1
B 1
B 1
```

Expected Output:

```
Enter terminal A
Current count for terminal A: 1
Current count for terminal B: 0
Enter terminal B
Current count for terminal A: 1
Current count for terminal B: 1
Enter terminal A
Current count for terminal A: 2
Current count for terminal B: 1
Enter terminal B
Current count for terminal A: 2
Current count for terminal B: 2
Enter terminal A
Current count for terminal A: 3
Current count for terminal B: 2
Enter terminal B
```

Received Output:

```
Enter terminal A
Current count for terminal A: 1
Current count for terminal B: 0
Enter terminal B
```



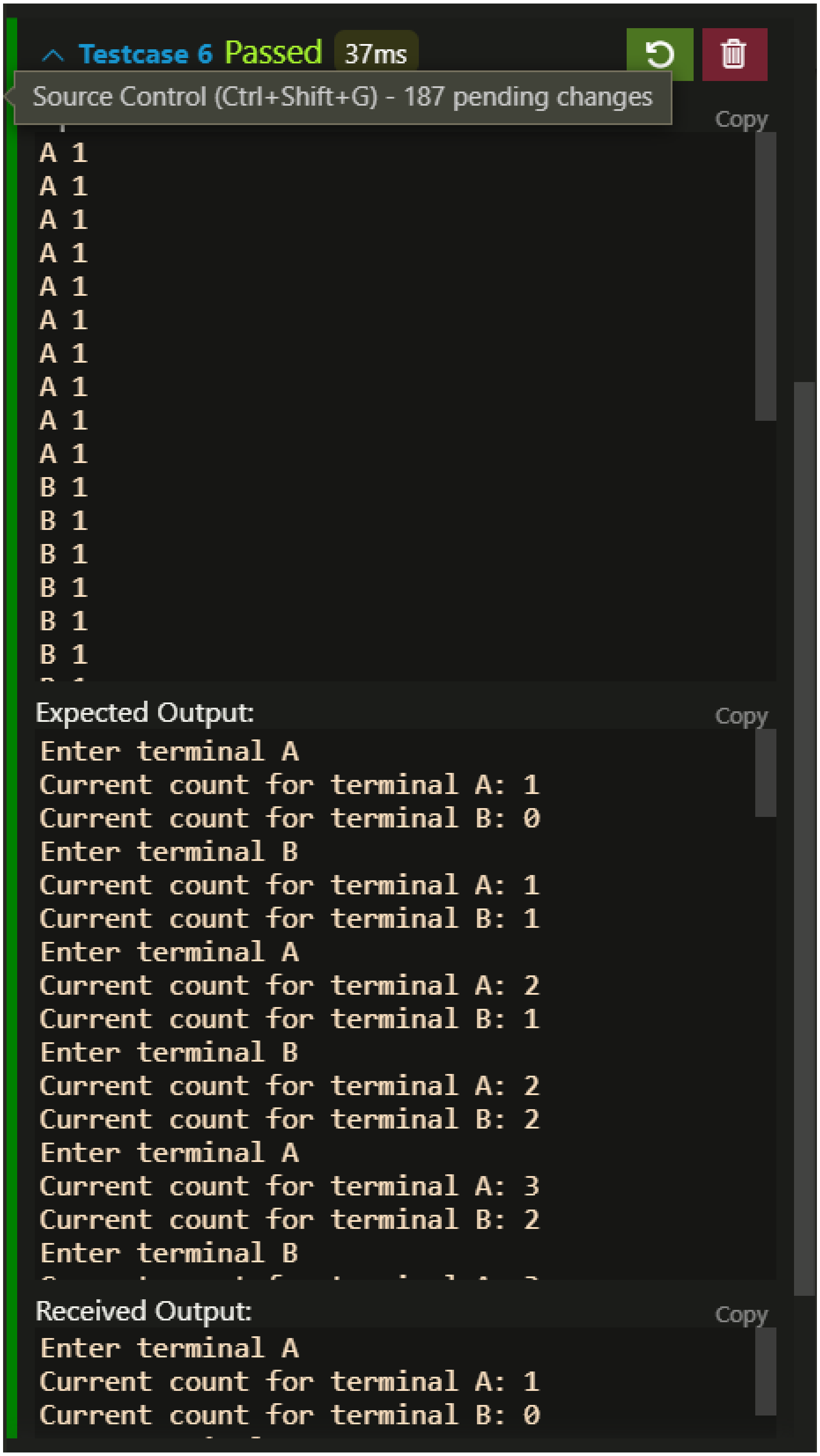
```
A 1
A 1
A 1
A 1
A 1
A 1
A 1
A 1
A 1
A 1
B 1
B 1
B 1
B 1
B 1
B 1
```

Expected Output:

```
Enter terminal A
Current count for terminal A: 1
Current count for terminal B: 0
Enter terminal B
Current count for terminal A: 1
Current count for terminal B: 1
Enter terminal A
Current count for terminal A: 2
Current count for terminal B: 1
Enter terminal B
Current count for terminal A: 2
Current count for terminal B: 2
Enter terminal A
Current count for terminal A: 3
Current count for terminal B: 2
Enter terminal B
```

Received Output:

```
Enter terminal A
Current count for terminal A: 1
Current count for terminal B: 0
Enter terminal B
```



```
A 1
A 1
A 1
A 1
A 1
A 1
A 1
A 1
A 1
A 1
B 1
B 1
B 1
B 1
B 1
B 1
```

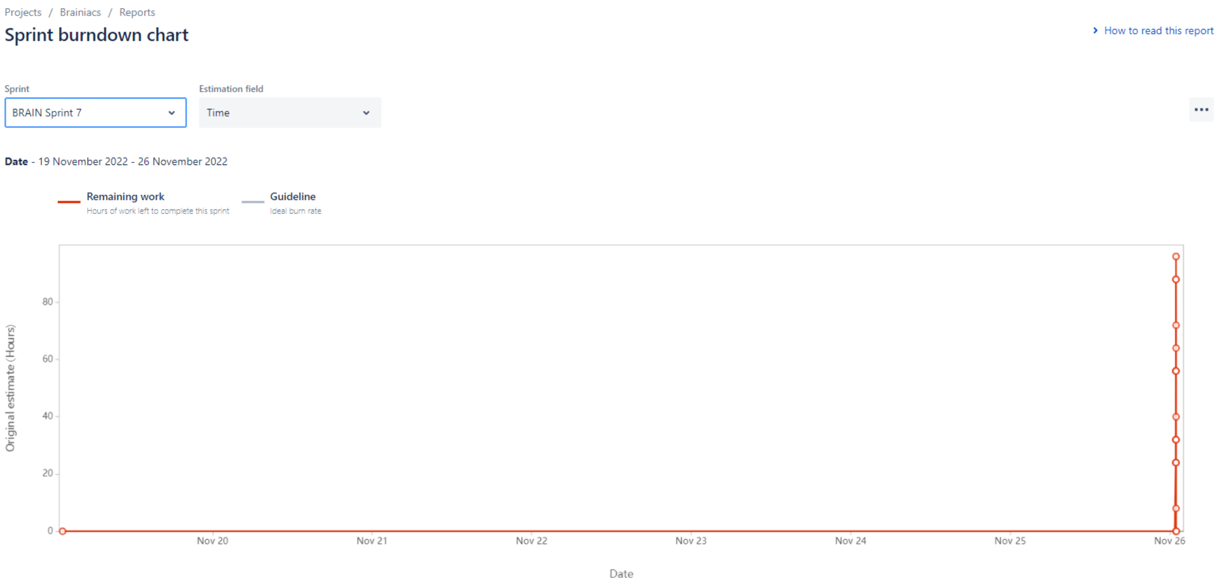
Expected Output:

```
Enter terminal A
Current count for terminal A: 1
Current count for terminal B: 0
Enter terminal B
Current count for terminal A: 1
Current count for terminal B: 1
Enter terminal A
Current count for terminal A: 2
Current count for terminal B: 1
Enter terminal B
Current count for terminal A: 2
Current count for terminal B: 2
Enter terminal A
Current count for terminal A: 3
Current count for terminal B: 2
Enter terminal B
```

Received Output:

```
Enter terminal A
Current count for terminal A: 1
Current count for terminal B: 0
```

1i) Burndown Chart



Projects / Brainiacs / Reports

Sprint burndown chart

› How to read this report

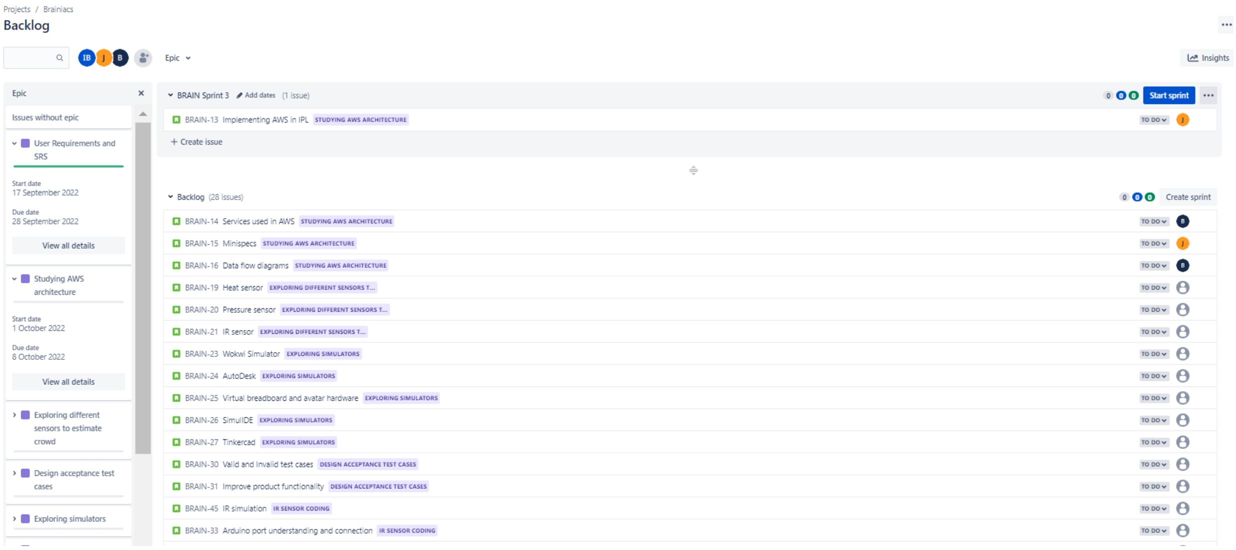
Sprint: BRAIN Sprint 7

Estimation field: Time

Date - 19 November 2022 - 26 November 2022

1j) JIRA/GanttPRO report showing all the tasks created, tracked, updated, monitored and closed

List of all Issues



Projects / Brainiacs

Backlog

IB J B Epic

Insights

Epic

Issues without epic

User Requirements and SRS

Start date
17 September 2022

Due date
28 September 2022

View all details

Studying AWS architecture

Start date
1 October 2022

Due date
8 October 2022

View all details

Exploring different sensors to estimate crowd

Design acceptance test cases

Exploring simulators

BRAIN Sprint 3 Add dates (1 issue) 0 0 0 Start sprint

BRAIN-13 Implementing AWS in IPL STUDYING AWS ARCHITECTURE TO DO J

+ Create issue

Backlog (28 issues) 0 0 0 Create sprint

BRAIN-14 Services used in AWS STUDYING AWS ARCHITECTURE TO DO B

BRAIN-15 Minispecs STUDYING AWS ARCHITECTURE TO DO J

BRAIN-16 Data flow diagrams STUDYING AWS ARCHITECTURE TO DO B

BRAIN-19 Heat sensor EXPLORING DIFFERENT SENSORS T... TO DO

BRAIN-20 Pressure sensor EXPLORING DIFFERENT SENSORS T... TO DO

BRAIN-21 IR sensor EXPLORING DIFFERENT SENSORS T... TO DO

BRAIN-23 Wokwi Simulator EXPLORING SIMULATORS TO DO

BRAIN-24 AutoDesk EXPLORING SIMULATORS TO DO

BRAIN-25 Virtual breadboard and avatar hardware EXPLORING SIMULATORS TO DO

BRAIN-26 SimulIDE EXPLORING SIMULATORS TO DO

BRAIN-27 Tinkercad EXPLORING SIMULATORS TO DO

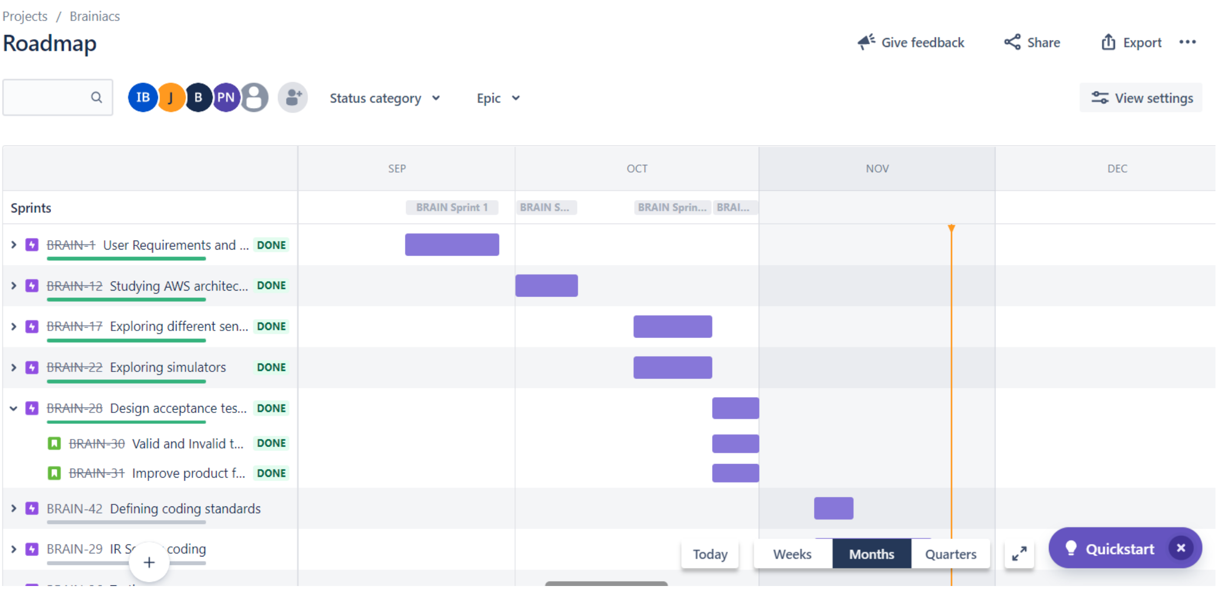
BRAIN-30 Valid and Invalid test cases DESIGN ACCEPTANCE TEST CASES TO DO

BRAIN-31 Improve product functionality DESIGN ACCEPTANCE TEST CASES TO DO

BRAIN-45 IR simulation IR SENSOR CODING TO DO

BRAIN-33 Arduino port understanding and connection IR SENSOR CODING TO DO

Issues completed by 4th sprint (done designing acceptance test cases)



Projects / Brainiacs

Roadmap

Give feedback Share Export

IB J B PN Status category Epic View settings

SEP OCT NOV DEC

Sprints BRAIN Sprint 1 BRAIN S... BRAIN Sprin... BRAI...

~~BRAIN-1~~ User Requirements and ... DONE

~~BRAIN-12~~ Studying AWS architec... DONE

~~BRAIN-17~~ Exploring different sen... DONE

~~BRAIN-22~~ Exploring simulators DONE

~~BRAIN-28~~ Design acceptance tes... DONE

~~BRAIN-30~~ Valid and Invalid t... DONE

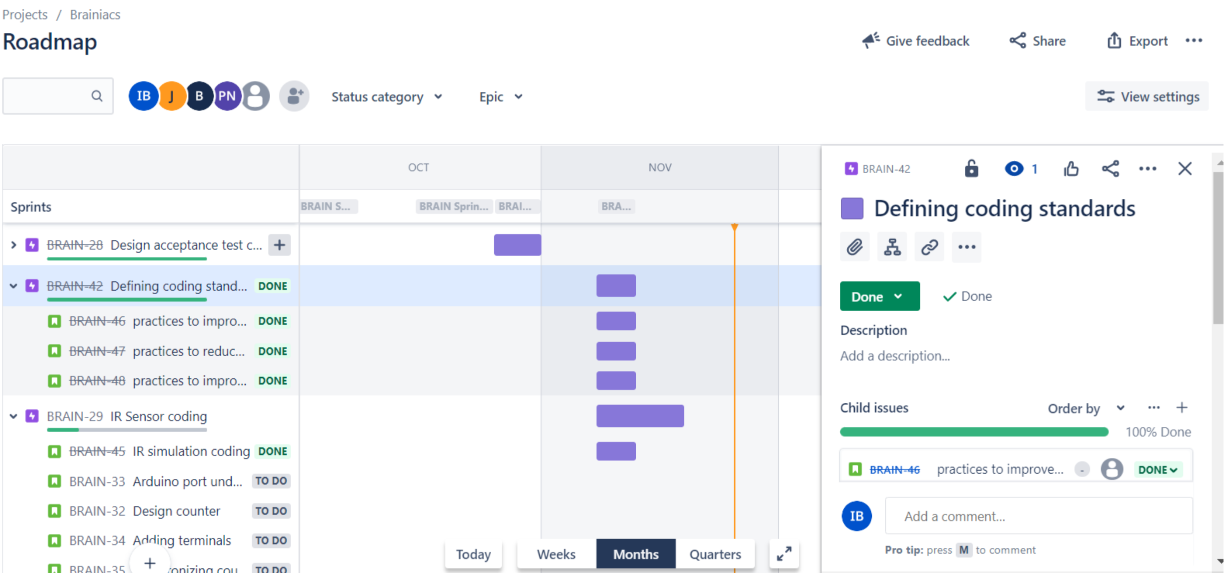
~~BRAIN-31~~ Improve product f... DONE

BRAIN-42 Defining coding standards

BRAIN-29 IR S coding

Today Weeks Months Quarters Quickstart

Completed 5th sprint (Defined coding standards and started IR sensor coding)



~~BRAIN-28~~ Design acceptance test c...	
~~BRAIN-42~~ Defining coding stand...	DONE
~~BRAIN-46~~ practices to impro...	DONE
~~BRAIN-47~~ practices to reduc...	DONE
~~BRAIN-48~~ practices to impro...	DONE
BRAIN-29 IR Sensor coding	
~~BRAIN-45~~ IR simulation coding	DONE
BRAIN-33 Arduino port und...	TO DO
BRAIN-32 Design counter	TO DO
BRAIN-34 Adding terminals	TO DO
BRAIN-35 ...onizing cou...	TO DO

BRAIN-42

Defining coding standards

Done ✓ Done

Description

Add a description...

Child issues Order by

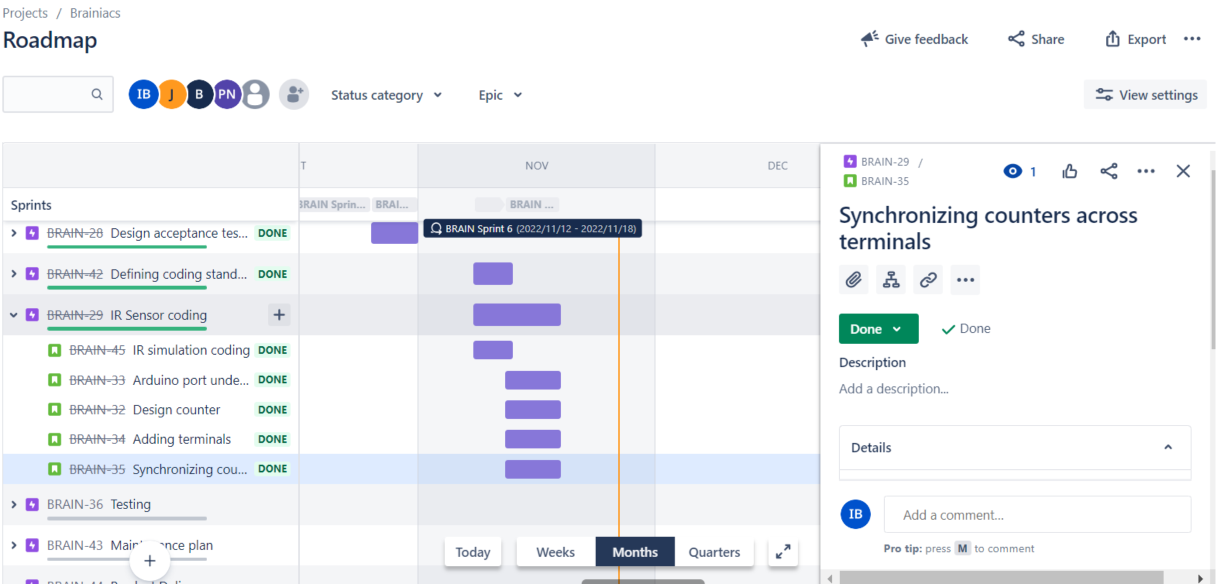
100% Done

~~BRAIN-46~~ practices to improve... DONE

IB Add a comment...

Pro tip: press M to comment

Completed 6th sprint (IR sensor coding and synchronizing terminals)



~~BRAIN-28~~ Design acceptance tes...	DONE
~~BRAIN-42~~ Defining coding stand...	DONE
~~BRAIN-29~~ IR Sensor coding	
~~BRAIN-45~~ IR simulation coding	DONE
~~BRAIN-33~~ Arduino port unde...	DONE
~~BRAIN-32~~ Design counter	DONE
~~BRAIN-34~~ Adding terminals	DONE
~~BRAIN-35~~ Synchronizing cou...	DONE
BRAIN-36 Testing	
BRAIN-43 Mair...nce plan	

BRAIN-29 / BRAIN-35

Synchronizing counters across terminals

Done ✓ Done

Description

Add a description...

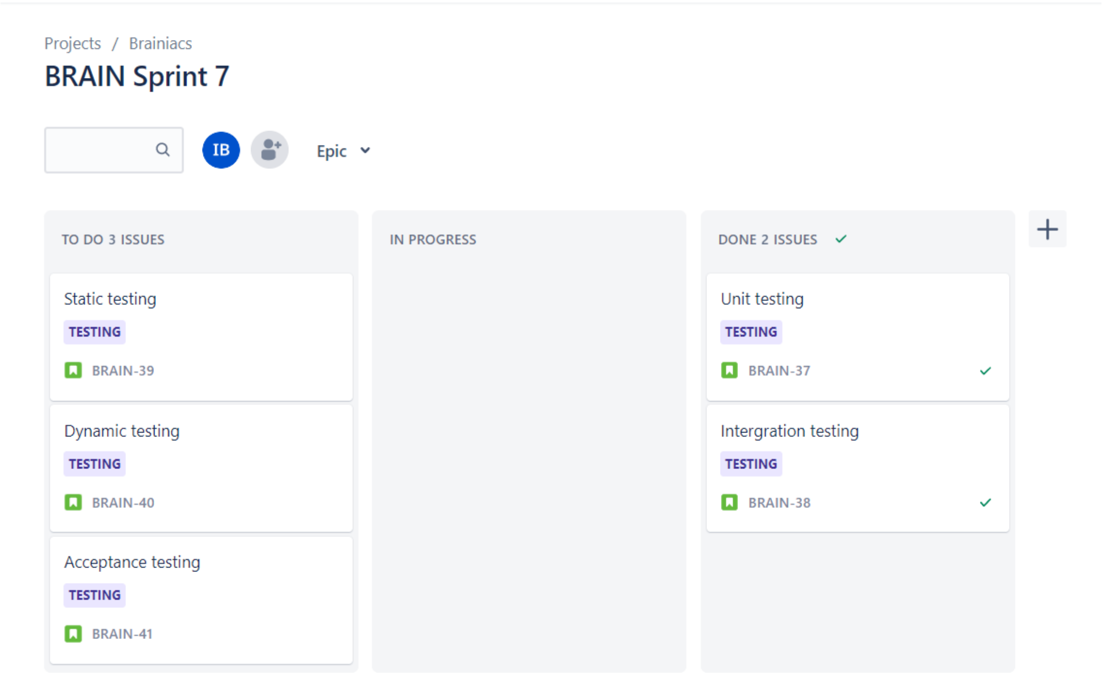
Details

IB Add a comment...

Pro tip: press M to comment

Sprint 7 - testing

During sprint



Projects / Brainiacs

BRAIN Sprint 7

IB Epic

TO DO 3 ISSUES

Static testing
TESTING
BRAIN-39

Dynamic testing
TESTING
BRAIN-40

Acceptance testing
TESTING
BRAIN-41

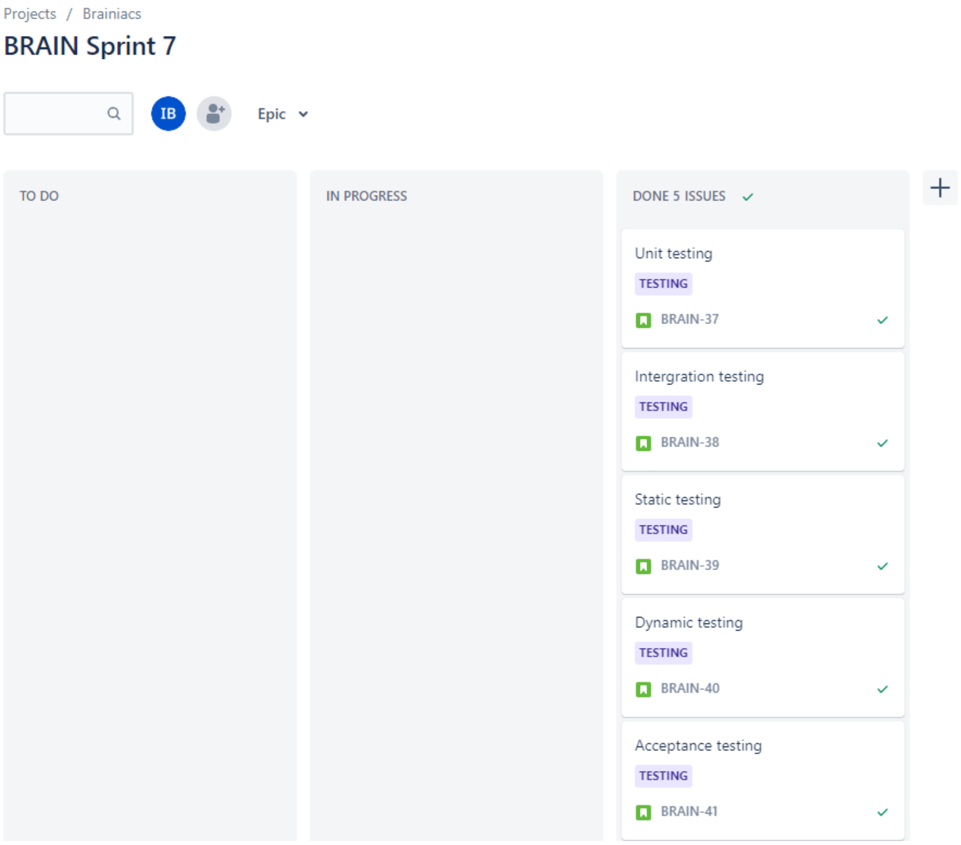
IN PROGRESS

DONE 2 ISSUES ✓

Unit testing
TESTING
BRAIN-37 ✓

Intergration testing
TESTING
BRAIN-38 ✓

After Completion of sprint 7



Projects / Brainiacs

BRAIN Sprint 7

IB Epic

TO DO

IN PROGRESS

DONE 5 ISSUES ✓

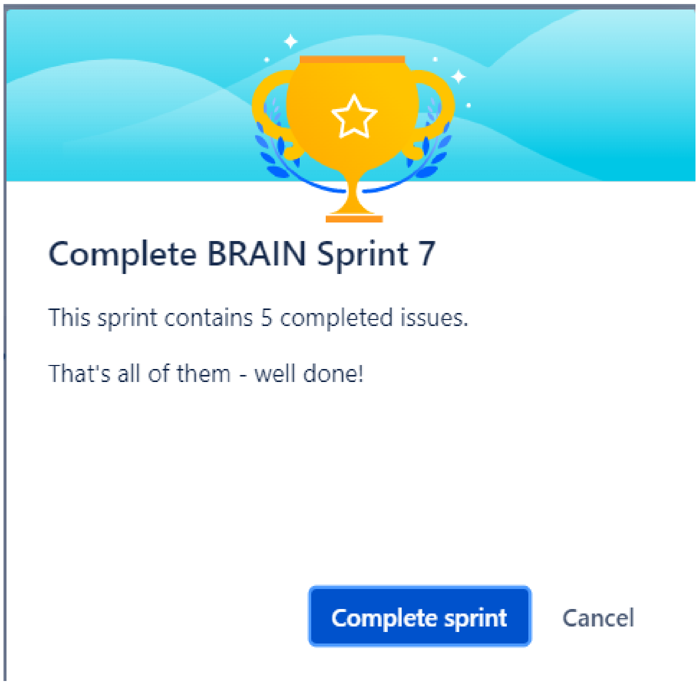
Unit testing
TESTING
BRAIN-37 ✓

Intergration testing
TESTING
BRAIN-38 ✓

Static testing
TESTING
BRAIN-39 ✓

Dynamic testing
TESTING
BRAIN-40 ✓

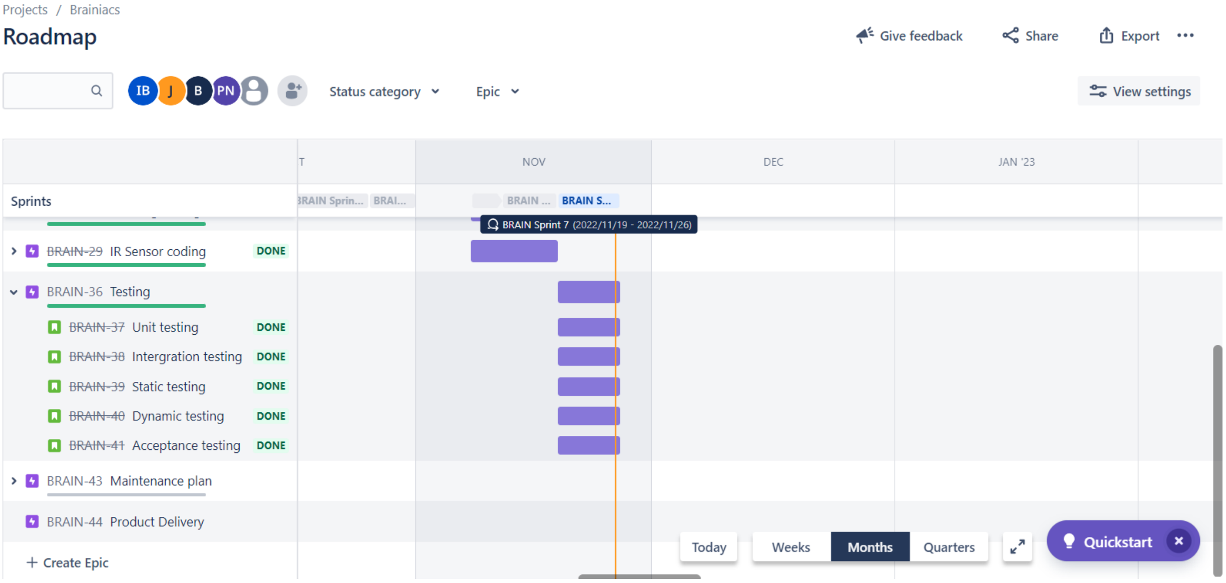
Acceptance testing
TESTING
BRAIN-41 ✓



Complete BRAIN Sprint 7

This sprint contains 5 completed issues.

That's all of them - well done!



Projects / Brainiacs

Roadmap

BRAIN Sprint 7 (2022/11/19 - 2022/11/26)

~~BRAIN-29~~ IR Sensor coding DONE

BRAIN-36 Testing

~~BRAIN-37~~ Unit testing DONE

~~BRAIN-38~~ Intergration testing DONE

~~BRAIN-39~~ Static testing DONE

~~BRAIN-40~~ Dynamic testing DONE

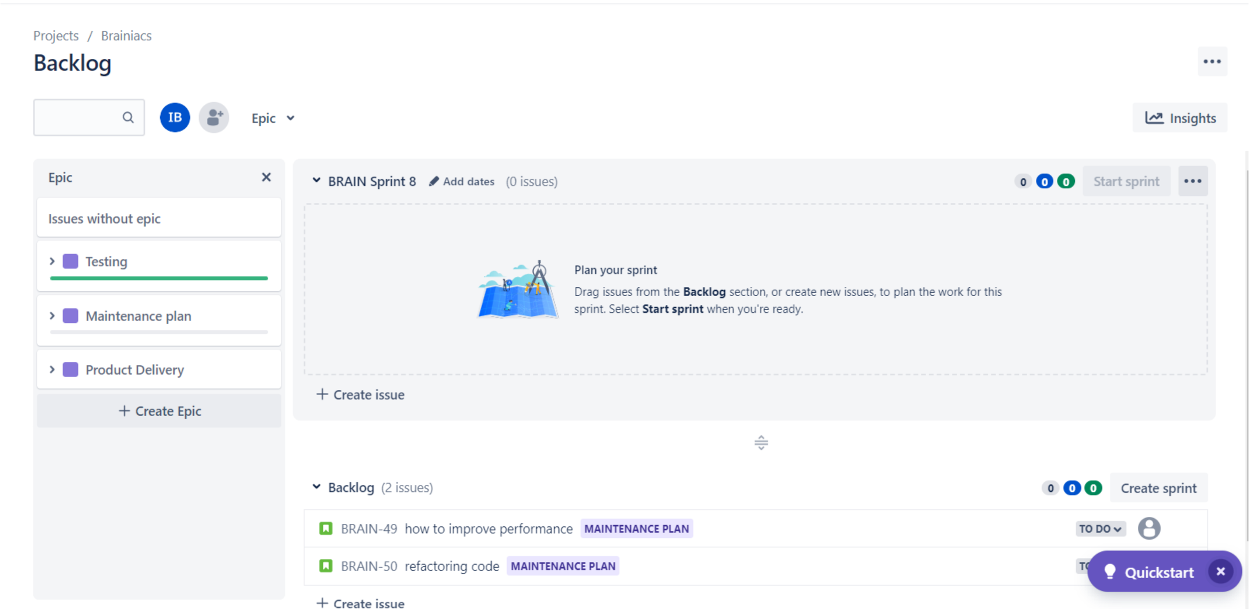
~~BRAIN-41~~ Acceptance testing DONE

BRAIN-43 Maintenance plan

BRAIN-44 Product Delivery

+ Create Epic

Backlog after testing



Projects / Brainiacs

Backlog

IB Epic Insights

Epic ×

Issues without epic

- Testing
- Maintenance plan
- Product Delivery

+ Create Epic

BRAIN Sprint 8 Add dates (0 issues) 0 0 0 Start sprint

Plan your sprint

Drag issues from the **Backlog** section, or create new issues, to plan the work for this sprint. Select **Start sprint** when you're ready.

+ Create issue

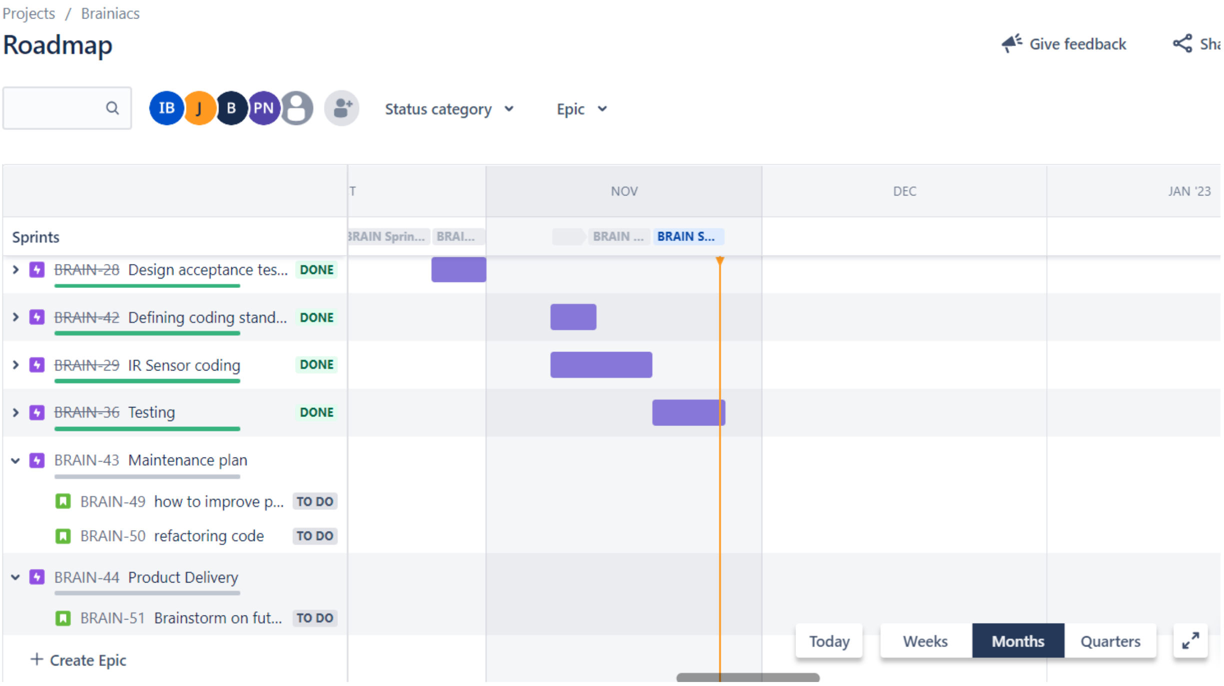
Backlog (2 issues) 0 0 0 Create sprint

BRAIN-49	how to improve performance	MAINTENANCE PLAN	TO DO
BRAIN-50	refactoring code	MAINTENANCE PLAN	TO

+ Create issue

Quickstart ×

Future work -
How to maintain and improve our product by refactoring code and optimizing our solution.



Projects / Brainiacs

Roadmap

Give feedback Sha

IB J B PN Status category Epic

T NOV DEC JAN '23

Sprints BRAIN Sprin... BRAI... BRAIN ... BRAIN S...

Issue	Status
~~BRAIN-28~~ Design acceptance tes...	DONE
~~BRAIN-42~~ Defining coding stand...	DONE
~~BRAIN-29~~ IR Sensor coding	DONE
~~BRAIN-36~~ Testing	DONE
BRAIN-43 Maintenance plan	
BRAIN-49 how to improve p...	TO DO
BRAIN-50 refactoring code	TO DO
BRAIN-44 Product Delivery	
BRAIN-51 Brainstorm on fut...	TO DO

+ Create Epic

Today Weeks Months Quarters

2. Github Link to Project:

[Link to github project]