Sprint planning is done on the Jira

Jira Link to show the sprints: <https://malini.atlassian.net/jira/projects>

FoodBox Project is completed in 4 sprints.

|  |  |
| --- | --- |
| Sprint Number | Sprint Tasks |
| Sprint week – 1 | UI Designing |
| Sprint week – 2 | backend designing and development |
| Sprint Week – 3 | 1. implement automation testing before the application enters the CI/CD pipeline 2. Git branching to do basic automation testing of the application in it separately |
| Sprint Week – 4 | deploy and host the application on an AWS EC2 instance  1. Testing and Documentation |

## Home Page of my Application…

## user can login to be able to view items and add to cart and buy the food if he trying to add food without login there is error showing that you can login first then you can able to add this food to cart

**frontend running on port 4200**

**When user can open application then he can see the register and login button and once he register and login then this login and register button will disappear and In place of that we can see cart icon and in Navbar also we can see my Account list in y account user can see he’s profile and order related thing**

**Even user can search food by using food name, type(veg), category(Indian, Chinese, Briyani etc)**

* 1. **Project Description:**

DESCRIPTION

Create a dynamic and responsive online food delivery web application for ordering food items of different cuisines from a restaurant.

**Background of the problem statement:**

Foodbox is a restaurant chain that delivers food items of different cuisines at affordable prices. It was established in 2014 in Bengaluru, India. It had been serving fine all these years, however, the business analysts noticed a decline in sales since 2016. They found out that the online ordering of food items with companies, such as Swiggy and Foodpanda were gaining more profit by eliminating middlemen from the equation. As a result, the team decided to hire a Full Stack developer to develop an online food delivery web application with a rich and user-friendly interface.  
You are hired as the Full Stack Java developer and are asked to develop the web application. The management team has provided you with the requirements and their business model so that you can easily arrange different components of the application.

**Features of the application:**

1. Registration
2. Login
3. Payment gateway
4. Searching
5. Filtering
6. Sorting
7. Dynamic data

**Recommended technologies:**

1. Database management: MySQL and Oracle
2. Backend logic: Java programming, NodeJS
3. Frontend development: JSP, Angular, Bootstrap, HTML/CSS, and Javascript
4. Automation and testing technologies: Selenium, Jasmine, and TestNG
5. DevOps and production technologies: Git, GitHub, Jenkins, Docker, Kubernetes, and AWS

**Project development guidelines:**

* The project will be delivered within four sprints with every sprint delivering a minimal viable product.
* It is mandatory to perform proper sprint planning with user stories to develop all the components of the project.
* The learner can use any technology from the above-mentioned technologies for different layers of the project.
* The web application should be responsive and should fetch or send data dynamically without hardcoded values.
* The learner must maintain the version of the application over GitHub and every new change should be sent to the repository.
* The learner must implement a CI/CD pipeline using Jenkins.
* The learner should also deploy and host the application on an AWS EC2 instance.
* The learner should also implement automation testing before the application enters the CI/CD pipeline.
* The learner should use Git branching to do basic automation testing of the application in it separately.
* The learner should make a rich frontend of the application, which is user- friendly and easy for the user to navigate through the application.
* There will be two portals in the application, namely admin and user portal.

**Admin Portal:**  
The admin portal deals with all the backend data generation and product information. The admin user should be able to:

* Add or remove different cuisines to or from the application to build a rich product line
* Edit food item details like name, price, cuisine, description, and offers to keep it aligned to the current prices
* Enable or disable the food items

**User Portal:**  
It deals with the user activities. The end-user should be able to:

* Sign-in to the application to maintain a record of activities
* Search for food items based on the search keyword
* Apply filters and sort results based on different cuisines to get the best deals
* Add all the selected food items to a cart and customize the purchase at the end
* Perform a seamless payment process
* Get an order summary details page once the payment is complete
  1. **Core concepts used in the project:**
  2. **Front End:**

HTML.

CSS3 and Bootstrap4. TypeScript.

Angular, Angular Material for View.

* 1. **VS Code:** As an IDE to design frontend of the application.
  2. **Git:** To connect and push files from the local system to GitHub.
  3. **GitHub:** To store the application code and track its versions
  4. **Scrum:** An efficient agile framework to deliver the product incrementally.
  5. **Back End:**

Eclipse IDE.

Java Programming.

Searching and Sorting Spring Boot DevTools.

Spring Web and Spring Data JPA.

* 1. **Testing and DevOps:**

Selenium. Jenkins.

Docker and Testing.

1. **Architecture diagram / flow chart**

A picture containing text, screenshot, diagram, rectangle

Description automatically generated

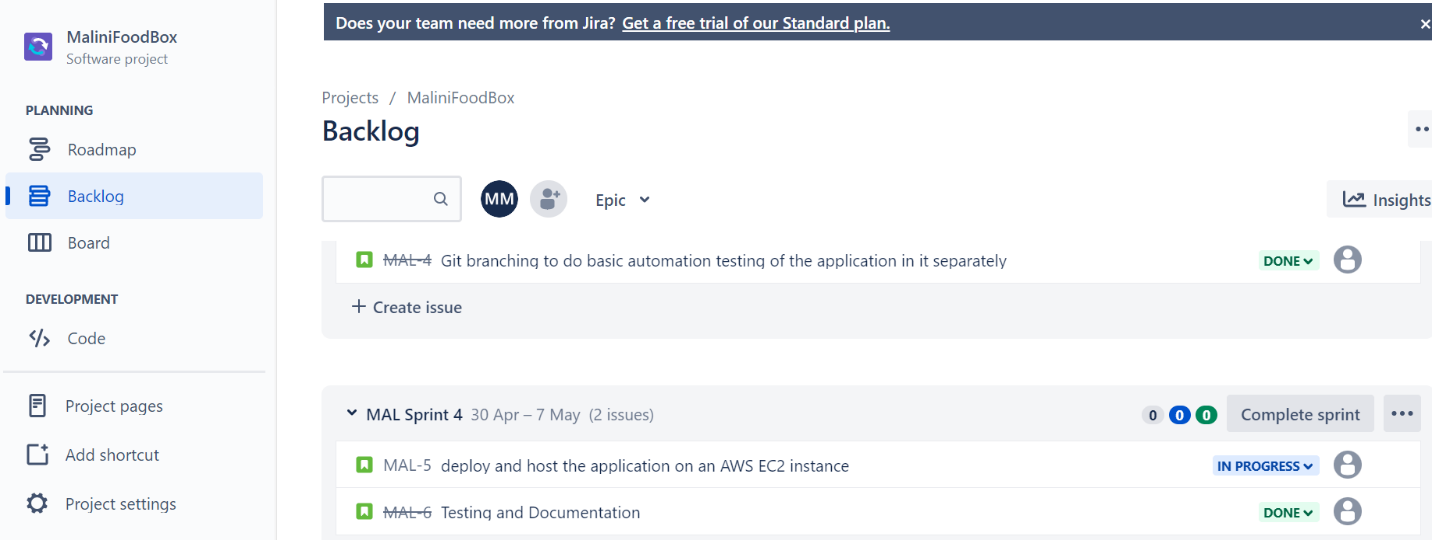
* 1. **Project Users Stories : ( Agile and Scrum )**

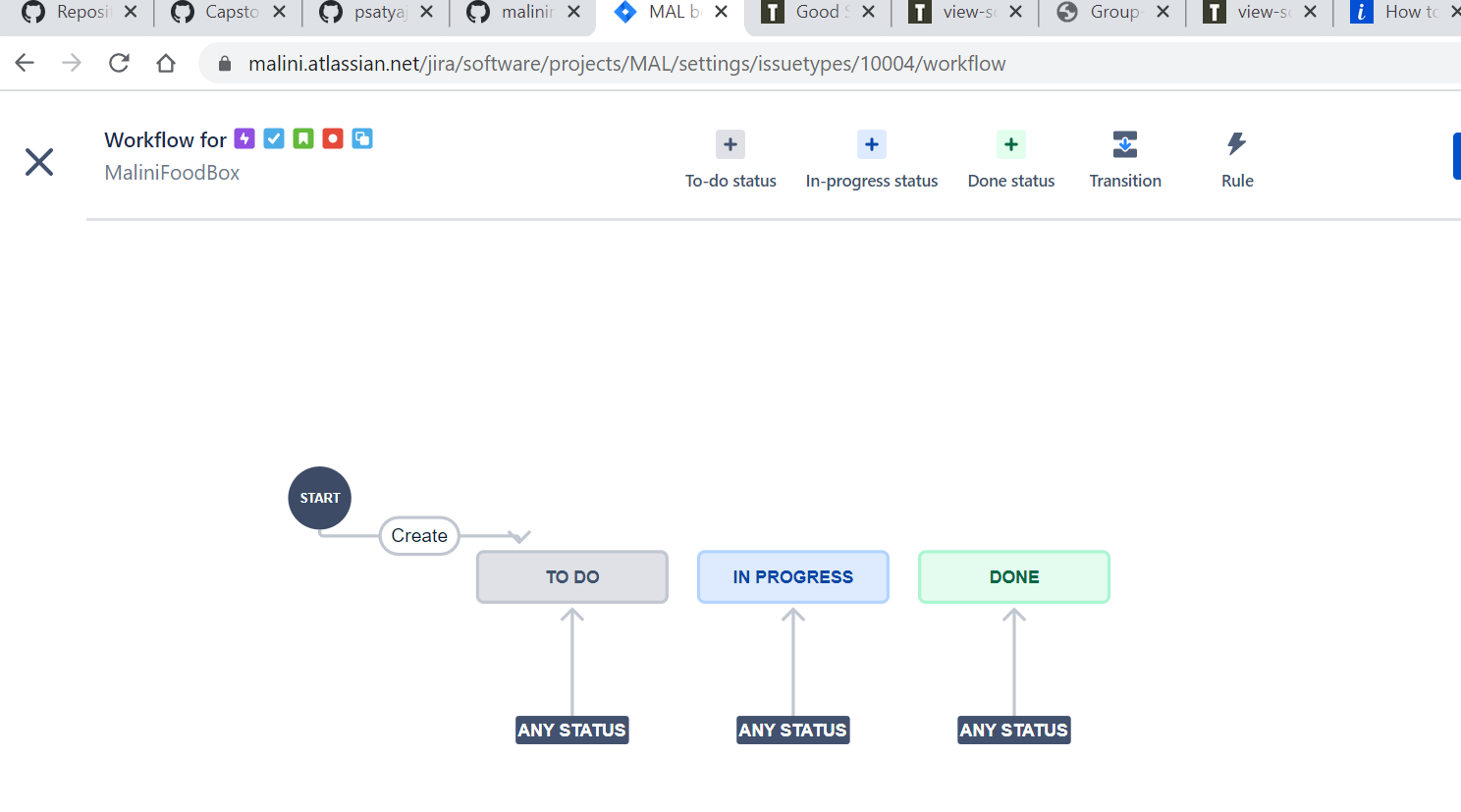
The project is planned to be completed in 3 sprints. Tasks assumed to be completed in the sprint are:

* Creating the flow of the application
* Initializing git repository to track changes as development progresses.
* Writing the Java program to fulfill the requirements of the project.
* Testing the Java program with different kinds of User input
* Pushing code to GitHub.

Graphical user interface, text

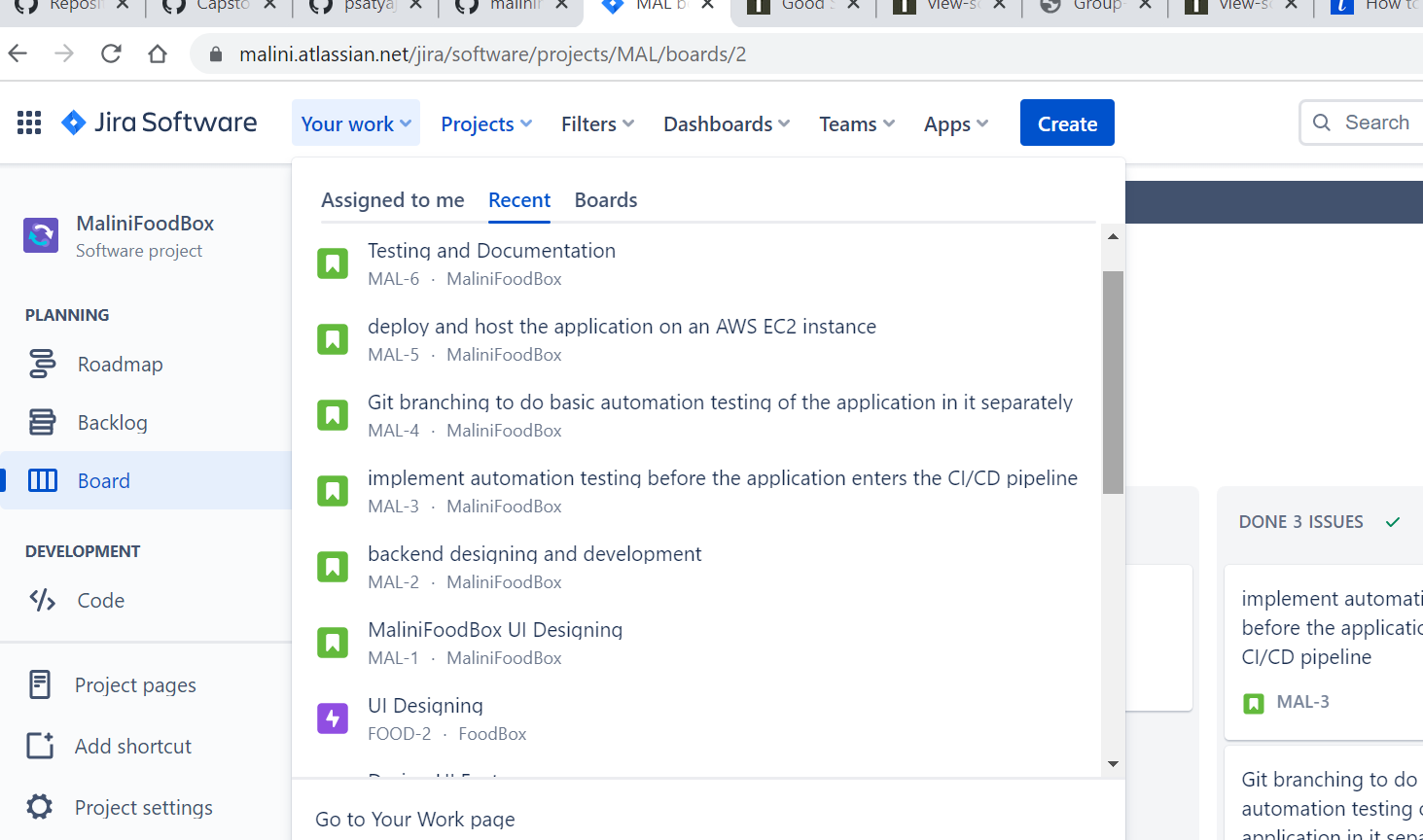
Description automatically generated

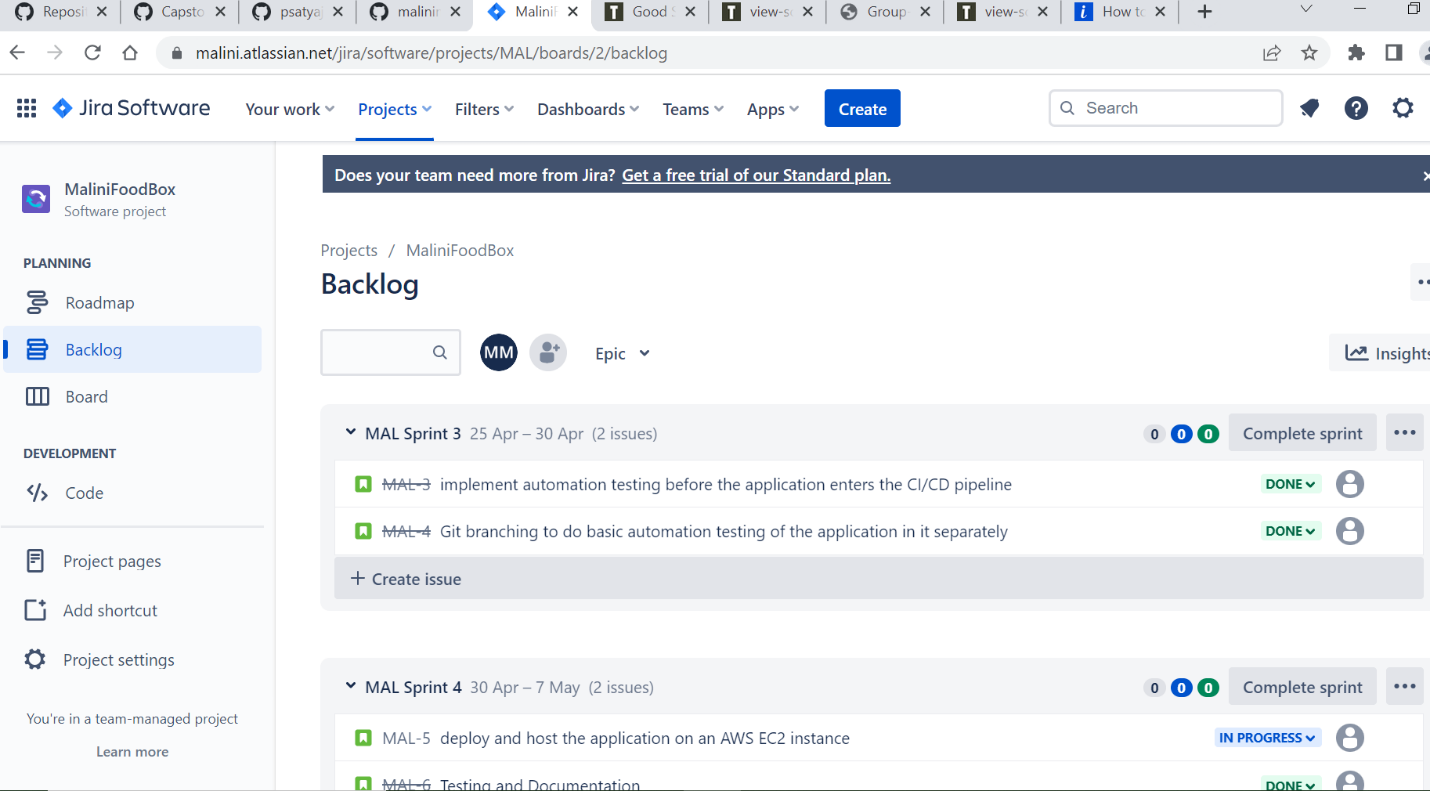


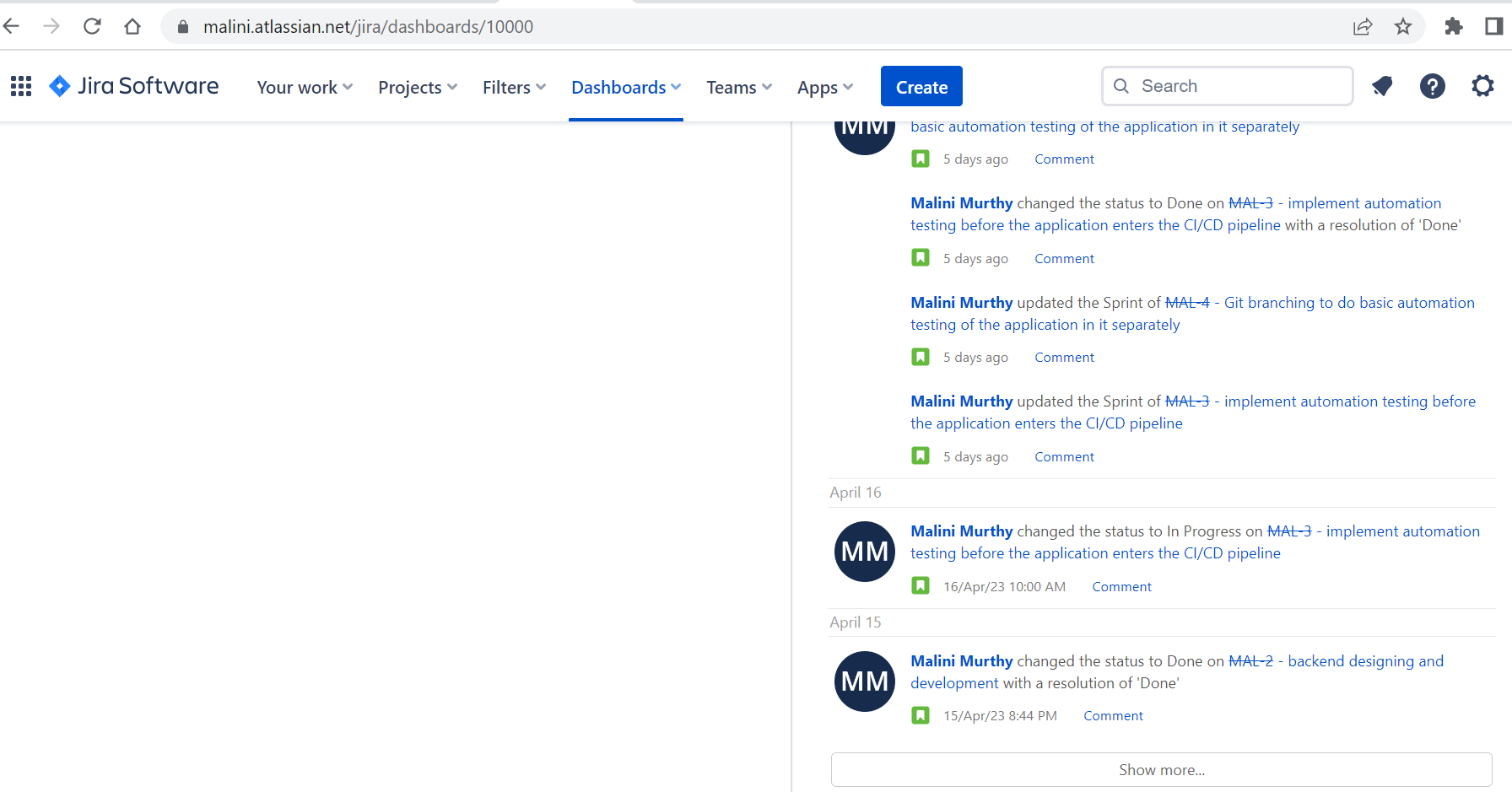


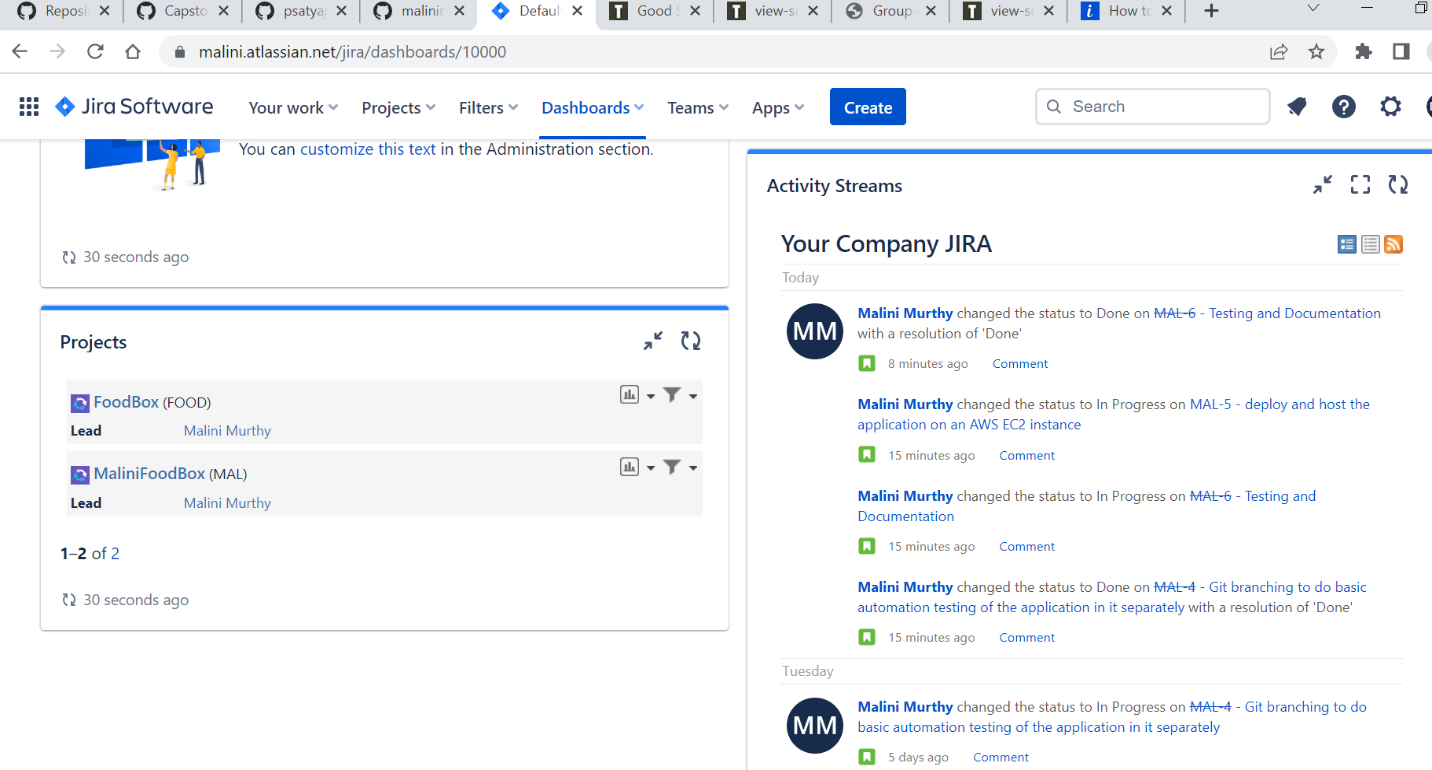
Graphical user interface, text, email

Description automatically generated









A screenshot of a computer

Description automatically generated