**1.1 DASHBOARD DISPLAY**

**EXPECTED FUNCTIONALITY:**

* All sensor charts corresponding to the selected production line should be displayed in the dashboard when a specific line is chosen from the line selector.
* The sensor charts must automatically refresh with live data updates every 30 seconds to ensure up-to-date information is presented.
* The data points within each chart must be color-coded based on the traffic light system:
  + **Green**: Data points within the acceptable range.
  + **Amber**: Data points within 10% above or below the acceptable range.
  + **Red**: Data points exceeding the acceptable range.
* Product operators and managers should be able to use the calendar functionality to filter historical sensor data. The filter should allow users to specify a custom time range (start and end date and time) for the data to be displayed.

**INCORRECT FUNCTIONALITY:**

* The line selector does not display the correct sensor charts for the selected production line, or displays only a partial set of data.
* Sensor charts do not update every 30 seconds as expected, showing outdated data.
* Data points appear in grey due to a malfunction in the traffic light color-coding system- this is only acceptable if there is no forecasted data generated from the prophet models for that timestamp.
* The filtering functionality does not display data correctly when a specific time range is selected; historical data is not properly refined based on the chosen filters.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | Corresponding system feature | Priority Level | Input provided by user | Expected output | Actual Output | Test Outcome |
| Line selector | Dashboard UI (Home page)- | High | Select ‘line 5’ from dropdown | All sensor charts for Line 5 are displayed- this includes donut charts, bar chart, scatter graph and pie chart for the aggregated sensors. Small start cards at the top of the dashboard display additional information per line, including average temperature, optimal and warning sensors and total sensors (17) | All sensor charts (17 in total) for Line 5 are displayed | PASS |
| Live Data Refresh | Dashboard UI (Home page)- | High | Wait 30 seconds upon opening the home page and observe the donut chart for r02 of Line4 | After 30 seconds, r02 sensor value should change | Website ran at 18:41:13- r02 initial temperature value was 201.24, at 18:41:43, temperature value for r02 increases to 202.09 | PASS |
| Historical data filtering | Dashboard UI (Historical Data page)- | High | Enter specific start and end date - |  |  | PASS |
| Historical data filtering | Dashboard UI (Historical data page) | High | Enter specific date incorrect (e.g. start date is after the end date) |  |  |  |

**STATISTIC (SENSOR-DATA) DISPLAY**

**EXPECTED FUNCTIONALITY:**

* When a specific sensor is selected, the user is redirected to a detailed sensor page that displays both a scatter plot and a temperature heatmap, populated with live and historical data.
* The sensor page should feature time filters at the top, enabling users to filter the historical data displayed on the scatter plot by predefined time intervals, such as the past hour, past 2 hours, past day, or past week.
* The scatter plot should include a toolbox at the top, allowing users to interact with the graph. Features include zoom in, zoom out, and reset zoom, which adjust the x-axis (time) and the clusters of scatter points accordingly.
* The sensor page must also display key statistics including the mean, minimum, and maximum values.

**INCORRECT FUNCTIONALITY:**

* Selecting a sensor does not redirect the user to its corresponding sensor-data page.
* The sensor page does not display either live or historical data on the scatter plot or temperature heatmap.
* The statistics (mean, minimum, and maximum) for the selected sensor are either missing or displayed incorrectly.
* The time filters are non-functional or do not load the correct historical data when applied.
* The interactive features of the scatter plot, including zoom in, zoom out, and reset zoom, are not functioning correctly, and they do not adjust the x-axis (time) or scatter point clusters as expected.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Sensor-data page navigation | Dashboard UI – Sensor-data page | High | Click on the R03 donut chart for line4 | Redirect to detailed sensor-data page for R03 | Redirected to sensor data page for R03- | PASS |
| Time Filter | Dashboard UI – Sensor-data page | High | Select ‘Past Hour’ filter on R03 (line4) sensor data page | Graphs will show data for past hour – (previous 60 mins up to current time period) | ‘Past Hour’ filter selected at | PASS |
| Zoom in functionality | Dashboard UI – Sensor-data page | Medium | Specific sensor specified |  |  | PASS |
| Zoom out functionality | Dashboard UI – Sensor-data page | Medium | Specific sensor specified |  |  | PASS |
| Reset zoom | Dashboard UI – Sensor-data page | Medium | Specific sensor specified |  |  | PASS |
| Statistical metrics | Dashboard UI – Sensor-data page | High | Scroll down to view stat cards displaying mean, min and max metrics for R03 | Correct mean, min and max values for R03 – values should update, fluctuate |  | PASS |

**SIGNUP PROCESS**

**EXPECTED FUNCTIONALITY:**

* New users can register by providing their full name and email address.
* After registration, users can create and confirm their password.
* A "Passwords match" message is displayed when the password and confirmed password entries are identical.
* An eye icon allows users to view their password as they type.
* Upon successful registration, users are shown a notification informing them that they are awaiting product manager approval before they can log in, and the user is redirected to login page
* The registered user account is stored in the **user\_accounts** table, but remains inactive until approved by the product manager.
* A dark mode toggle is displayed, allowing users to switch the sign-up page to dark mode if desired.

**PASSWORD VALIDATION:**

* The password must meet the following criteria:
  + Minimum of 8 characters.
  + At least one special character (e.g., !, @, #, $, etc.).
  + At least one number.

**Incorrect Functionality:**

* The employee account is not recognized during login, even though the email address provided is a valid Rakusen email. This prevents users from accessing their accounts, despite having a registered company email.
* Users are unable to set their own password during registration. Additionally, the password validation process is malfunctioning, meaning the system fails to validate or confirm passwords properly. For instance:
* The "Passwords match" message is not displayed when the password and confirm password fields match.
* The eye icon that allows users to view their password while typing is either not displayed or is non-functional. This prevents users from seeing the password they are entering and could lead to input errors.
* After a user successfully registers, they do not receive a notification informing them that they must await product manager approval before being able to log in. This leaves users unaware that their account is inactive and awaiting approval.
* The registered user account is not being stored correctly in the **user\_accounts** table, or it is not marked as inactive pending product manager approval. This causes issues with account activation and login.
* The dark mode toggle for the sign-up page is not displayed, or it is non-functional, preventing users from switching to dark mode if they prefer it.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Email validation (valid) | Dashboard UI (Signup page) | High | [barirah@rakusens.com](mailto:barirah@rakusens.com) entered into the email box | Provided that password validation has been met, a ‘Registration successful, awaiting production manager approval’ message is displayed to the user, and the user is redirected to the login | The message ‘Registration successful! Waiting for production manager approval. Redirecting to login page…’ message is displayed to the user, and the user is redirected to the login page | PASS |
| Email validation (invalid) | Dashboard UI (Signup page) | High | [barirah@gmail.com](mailto:barirah@gmail.com) entered into the email box | When registration button is clicked on the message ‘Email must end with @rakusens.com’ is displayed to the user, indicating incorrect email credentials | Upon attempting to register the message ‘Email must end with @rakusens.com’ message is displayed, preventing successful registration | PASS |
| Password Validation (valid) | Dashboard UI (Signup page) | High | Hello123\* entered into the password validation box and confirm password box | Message displayed to the user ‘Passwords Match’ message displayed to the user indicating that inputs for both password and confirmation of password match. Provided that the email inputted is also valid, a message is displayed to the user indicating successful registration, and the user is redirected to the login page | The message ‘Registration successful! Waiting for production manager approval. Redirecting to login page…’ message is displayed to the user, and the user is redirected to the login page | PASS |
| Password validation (Invalid) | Dashboard UI (Signup page) | High | Hi is inputted in both the password and confirm password box | Whilst the passwords match and a message is displayed indicating this, when the user presses the register button a message is displayed indicating that the password doesn’t match the validation criteria to register | The message, ‘ Passwords must be Minimum eight characters, at least one letter, one number and one special character:’ is displayed to the user, and the user is unable to register successfully | PASS |
| Passwords do not match | Dashboard UI (Signup page) | Medium | Hello123\* entered in first password box but Hello123 entered into second | Underneath the confirm password box, a passwords do not match message is displayed , and this message immediately changes to passwords match once they match | Passwords do not match message is displayed- this does not change to passwords match as both inputs are deliberately different and are therefore mismatched | PASS |
| Passwords match | Dashboard UI (Signup page) | Medium | Hello123\* entered into both create password and confirm password boxes | Underneath the confirm password box, a passwords do not match message is displayed , and this message immediately changes to passwords match once they match | Passwords do not match message is displayed- this changes to passwords match as both inputs are identical | PASS |
| Eye toggle functionality | Dashboard UI (Signup page | Medium | Hello12 and  Hello123 entered into create and confirm password boxes respectively | Passwords do not match msg displayed indicating a mismatch- the user can use the eye toggle on both the create and confirm password boxes to see their inputs into both boxes | Passwords do not match msg displayed indicating a mismatch- the user can use the eye toggle on both the create and confirm password boxes to see their inputs into both boxes | PASS |
| Dark mode toggle | Dashboard UI (Signup page) | Low | User toggles between dark and light mode 5 times on the Signup page | There is no delay when toggling between light and dark mode- the dark mode appearance should also be clearly visible with input boxes and inputs being clearly visible | There is no delay when toggling between light and dark mode- the dark mode appearance should also be clearly visible with input boxes and inputs being clearly visible | PASS |

**LOGIN/LOGOUT PROCESS**

**EXPECTED FUNCTIONALITY:**

* After the product manager's approval, users can log in to the system using their email address and personalised password.
* Once logged in, users can navigate to their account page via the navigation bar, where they can change their password as needed.
* Password validation is implemented on the account page, ensuring that the same validation requirements apply when a logged-in user changes their password.
* When users log out, they should be redirected to the login page immediately. There should be no delay, and subsequent logins should not require re-approval from the product manager.
* The eye icon allows users to view their password as they type.
* A dark mode toggle is displayed, allowing users to switch the login-up page to dark mode if desired.

**INCORRECT FUNCTIONALITY:**

* Users are unable to log in after being approved by the product manager, despite using their email and personalised password. This prevents approved users from accessing the system.
* After logging in, users cannot navigate to their account page via the navigation bar to change their password. This prevents users from updating their passwords as needed.
* The eye icon to reveal the password while typing is not working, preventing users from verifying their password input during login or when changing their password.
* Password validation on the account page is not functioning correctly, allowing users to set an invalid password or bypass validation when changing their password. This violates the expected password validation requirements for logged-in users.
* After logging out, users are not redirected to the login page immediately. Additionally, when attempting to log in again, users are prompted to wait for product manager approval, even though they have already been approved. This introduces unnecessary delays and prompts that should not be triggered after the initial approval .
* The dark mode toggle for the log-in page is not displayed, or it is non-functional, preventing users from switching to dark mode if they prefer it.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Login following account activation | Dashboard UI-Login page | High | Email: ‘barirah@rakusens.com’  Password:  Hello123\* | User is redirected to the main dashboard (home) page | User is redirected to the main dashboard (home) page | PASS |
| Login without account activation | Dashboard UI-Login page | High | Email: ‘barirah@rakusens.com’  Password:  Hello123\* | Despite successful registration, account has not been activated by production manager so a message should be displayed indicating that account is currently inactive when the user presses the Login button | After pressing the Login button, the message ‘ account is inactive’ is displayed | PASS |
| Change password – password validation (valid) | Dashboard UI-Account page | High | The user input the password ‘Hello1234\*’ | Upon logging out, when the user enters with this new password, they are able to log in to their existing account, as password validation has been met | User can log in with new set password, and can no longer log in with their previous password – upon attempting to do so the message ‘invalid email or password’ is displayed to the user | PASS |
| Change password – password validation (invalid) | Dashboard UI- Account page | High | The user inputs ‘hi’ | Password is not reset, so when the user logs out and attempts to login with this password, a message is displayed indicating invalid email or password. | Password is not reset, so when the user logs out and attempts to login with this password, a message is displayed indicating invalid email or password. | PASS |
| Eye toggle | Dashboard UI- Login page | High | Enter passwords in both fields, then click eye icon | Eye toggle reveals/hides password text as expected | Users can toggle password visibility on both fields using the icon | PASS |
| Logout access | Dashboard UI-Any page | High | Click on the logout icon from navigation bar | User is logged out and redirected to the login page | User is successfully logged out and redirected to login | PASS |
| Logout functionality | Dashboard UI-Any page | High | Log out and immediately try to log in again | No delays or re-approval prompts; login works as expected | User logs back in smoothly without re-approval issues | PASS |
| Dark mode Functionality |  |  | Toggle between dark and light mode multiple times | UI should switch without delay and retain visibility | Theme switches instantly and remains visually consistent | PASS |

**FORGOT PASSWORD FUNCTIONALITY**

**EXPECTED FUNCTIONALITY:**

* Users can access the "Forgot Password" link on the login page.
* Upon entering their operator ID and email, a confirmation pop-up message should appear indicating that the password reset request is being processed.
* A second message should confirm the successful reset of the password.
* The user account history is deleted, including all previous sessions, and the registration process restarts.

**INCORRECT FUNCTIONALITY:**

* The "Forgot Password" link does not work, and is therefore inaccessible
* Users are not prompted to enter necessary details (operator ID, email).
* Users are able to reset their passwords despite entering incorrect credentials (e.g. incorrect operator ID)
* Users do not receive a confirmation pop-up or message to indicate password reset request is being processed.
* Users do not receive a confirmation pop-up or message to indicate successful password reset
* The reset process does not clear the user’s account history, meaning previous sessions are not deleted, and the registration does not restart as expected.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Forgot password link- accessibility | Forgot password link | High | Click on "Forgot Password" link on the login page | User is redirected to the forgot password form where they are prompted to enter their operator ID and email address | User is successfully redirected to the password reset form | PASS |
| Forgot password flow- entering incorrect credentials | Forget Password Link | High | Enter incorrect operator ID or invalid email (e.g., "wrongID" and "user@rakusens.com") | System displays an error message indicating invalid Operator ID or Email and prevents reset | "Invalid operator ID or email " message is displayed; reset is not triggered | PASS |
| Forgot password flow- entering correct credentials | Forget Password Link | High | Enter 7 for ID and [barirah@ruksens.com](mailto:barirah@ruksens.com) for the email. | A message appears confirming that the password reset is being processed. | Confirmation messages are displayed of reset, system proceeds with reset | PASS |
| Absence of confirmation message | Forget Password Link | High |  |  |  | PASS |
| Deletion of previous account |  | High | Manager login approves the deletion of a user | Account history is cleared from the system; registration | System deletes previous sessions and user is redirected to the registration flow | PASS |

**API FUNCTIONALITY:**

**DATABASE CONNECTIVITY**

**EXPECTED FUNCTIONALITY:**

* **Production Line Data Storage:**
  + Data from production lines (line4 and line5) are stored in **distinct and clear tables**. For example, line4\_table and line5\_table, which makes it easy to query and manage data for each production line separately.
  + Data stored in these tables should include sensor values, timestamps, and any relevant metadata for each data point.
* **User Account Data Storage:**
  + User account information is stored securely in the **user\_accounts table**. This includes essential details such as the user’s email, name, role (e.g., manager, operator), and a hashed password.
  + **Hashing of passwords** is done using industry-standard algorithms (e.g., bcrypt, SHA-256) to ensure that sensitive data is not stored in plaintext.
* **Data Integrity and Security:**
  + **Secure Storage:** The database should be designed with proper access controls, ensuring that sensitive user information is encrypted and passwords are never stored in plaintext.
  + **Consistent Data Integrity:** Data is never corrupted, and proper foreign key relationships are maintained between tables (e.g., user accounts and production line data).

**INCORRECT FUNCTIONALITY:**

* **Improper Data Storage:**
  + Data for production lines is stored in the wrong table (e.g., sensor data from line4 being stored in the line5\_table), making data retrieval and analysis difficult.
* **Unsecured Data Storage:**
  + User passwords are stored in plaintext or using weak, outdated hashing algorithms (e.g., MD5), making them vulnerable to security breaches.
* **Broken Relationships:**
  + Tables are not properly related, or foreign key constraints are missing, which could lead to data inconsistencies and orphaned records.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Correct table storage | Database- backend | High | Observe simulated data storage in database for line4 is stored in line4 | Live data generated for line4 should be stored in line4 table of database, not any other table |  | PASS |
| Metadata integrity | Database- backend | High | Check timestamps/sensor metadata | All fields (timestamp, timezone..etc) populate- no NULL values |  | PASS |
| Password hashing | Database- backend  API code | High |  |  |  | PASS |
| Role-based storage | Database- backend | High | Promote operator to manager | Role change for user should be observed in database |  | PASS |

**LIVE DATA SIMULATION**

**EXPECTED FUNCTIONALITY:**

* **Data Generation Frequency:**
  + Live data is generated **every 30 seconds** to simulate real-time sensor data.
  + The generated data is consistent with realistic temperature values for each sensor
* **Data Fluctuation:**
  + The simulated live data fluctuates randomly by **2%** above or below the average, mimicking real-life sensor behaviour.
  + The fluctuation ensures that data is dynamic and not static, to show how sensors should typically experience small variances in output.
* **Data Storage:**
  + The live data generated every 30 seconds is stored in the **production line tables (line4, line5)** as it is generated.
  + This data becomes **historical data** once generated stored and should be retrievable such as on historical data and sensor pages.

**INCORRECT FUNCTIONALITY:**

* **Inconsistent Data Generation:**
  + Live data is not generated on schedule (i.e., not every 30 seconds) or data generation fails intermittently, which disrupts the real-time monitoring.
* **Lack of Fluctuation:**
  + Data is generated with no variation (no fluctuation), making it unrealistic. All values might be constant and not reflect the expected random changes over time.
* **Data Not Storing Properly:**
  + Live data is not stored in the correct tables, or data might be missing from the tables, making it impossible to track historical sensor readings over time.
* **Data Storage Errors:**
  + Data might fail to store in the database, or errors might occur during storage, resulting in missing data or corrupted records.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| 30-second data generation | Database- backend | High | Monitored the database tables line4 and line5 on 7/4/2025 at 07:50 | New data entries should appear in the database very 30s without any delays or missing intervals | Observed data generation every 30s with no missing periods | PASS |
| 2% fluctuation around the average | Database- backend | High | Collect 25 consecutive data entries for sensor r01 in line 4 from 7:44 to 7:56. Analyse the patterns in the temperature data | Sensor values should show random fluctuations around the average of 129.10 | Analysis of r01 data from the 25 entries: average 129.54 with all values within 2% fluctuations above or below | PASS |
| Correct table insertion | Database- backend | High | Run the dashboard App.py at 7:44 and monitor both line 4 and line5 tables to verify data insertion | All live data should be correctly inserted into line4 and line5 tables with the right sensor IDs, timestamps and temperature values | All data was generated correctly and all timestamps were accurate. Each sensor had appropriate temperatures data with no gaps or insertion errors | PASS |
| Historical data retention- historical data page | Dashboard UI- historical data page | High | Run the dashboard App.py at 7:44 and access the historical data page at 8:15 to assess if all data was retained | All generated data from 7:44 onwards should be viewable on the historical data page showing all the correct timestamp and temperature values and allowing filtering by time periods | The historical data page successfully showed all the data at 30s intervals from 7:44 to 8:15 showing all the sensor values and respective timestamp values | PASS |
| Historical data retention – sensor data page | Dashboard UI- sensor- data page |  | Run the dashboard App.py at 7:44 and access the sensor data page at 8:15 to assess if all data was retained | Each sensor dedicated sensor page should show historical data on each of their respective graphs with the correct timestamps and values for the time period | All dedicated sensor pages contained data from 7:44 onwards on the 3 graphs on each page showing correct data for the right timestamps | PASS |

**FILTERING**

**EXPECTED FUNCTIONALITY:**

* **Time Range Filtering:**
  + The API supports filtering of historical data by time ranges (e.g., specific start and end dates or a rolling window of the last day, week, or month).
  + Users should be able to specify **start and end times** to see only the relevant data from that period.
* **Sensor Value Filtering:**
  + Users can filter the data based on sensor values (e.g., only display data where the temperature exceeds a certain threshold or falls within a specific range).
  + The filtering is fast and accurate, refining the data returned by the API without affecting performance.
* **Combined Filters:**
  + Users can apply both time range and sensor value filters simultaneously to narrow down the data results.

**INCORRECT FUNCTIONALITY:**

* **Filtering Does Not Work:**
  + Users are unable to apply any filters, and all data is returned regardless of the filter parameters, leading to information overload and slow performance.
* **Incorrect Results from Filters:**
  + Filters do not work as expected. For example, filtering by a date range might not return the correct data, or filtering by sensor value might include incorrect values in the results.
* **Performance Issues with Filters:**
  + Applying filters causes significant performance degradation, with the API response time increasing considerably when filters are applied, especially for large datasets.
* IS API even handling filtering or is the other pages now? This is what chatgpt gave but im not sure

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
|  |  |  |  |  |  | PASS |
|  |  |  |  |  |  | PASS |
|  |  |  |  |  |  | PASS |
|  |  |  |  |  |  | PASS |

**ML MODEL INTEGRATION**

**EXPECTED FUNCTIONALITY:**

* **Prophet Model Execution:**
  + Prophet models should be executed **every 30 seconds** to predict sensor values for the next specific future timestamp (e.g., the next 24 hours).
  + These models should generate **predictions** for each sensor (e.g., predicted temperature, pressure) for both production lines (line4 and line5).
* **Storing Predictions:**
  + The results of the Prophet model, including the predicted values, upper bounds, and lower bounds, should be stored in distinct forecast tables, such as forecastLine4 and forecastLine5.
  + Each prediction must be timestamped with the time it was generated to ensure it can be traced to the specific time it forecasts for.
* **Model Accuracy:**
  + The models should be accurate enough to provide realistic predictions within the expected error margin, based on historical data.

**INCORRECT FUNCTIONALITY:**

* **Models Not Running on Schedule:**
  + The Prophet models are not executed regularly (i.e., not every 30 seconds), or they fail to run at all, leading to missing predictions or outdated forecasts.
* **Incorrect Predictions Stored:**
  + Predicted values are stored incorrectly, with missing or inaccurate upper and lower bounds, or the forecasts are stored in the wrong tables (e.g., forecastLine4 data stored in forecastLine5 table).
* **Missing Timestamps:**
  + Predictions are not timestamped correctly, or the timestamp does not correspond to the actual forecast time, making it impossible to track or validate the predictions.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Prophet models run every 30s | Machine learning integration-backend API | High | Monitored r01 for forecast line4 table on 7/04/2025 at 2:28 PM for a 15-minute time frame. Ran the machine learning program with the specific time requirements. | Forecasted data generated every 30s (30 data entries for 15 minutes) – data generation should be consistent, not sporadic. Check the database to see if forecastline4 table contains forecasted data inserted every 30s for the time range. | On 7/04/2025 at 2:28 PM, entered the current time for a forecast time of 15 minutes and observed a new row for each 30s interval in forcastline4 table. All 30 expected entries (covering the full 15-minute period) were present in the database. | PASS |
| Correct table storage | Database-backend | High | Executed the machine learning program at 2:45 pm and observed the database tables changing as the program an | Ensure the machine learning program inserts into forecastline4/forcastline5 tables and doesn’t modify any other tables | Confirmed by observations that new data was being added correctly to only forcastline4 and forecastline5. No other tables were modified | PASS |
| Clear existing table | Database-backend | High | Executed the machine learning program at 2:45pm knowing forcastline4 and forecastline5 already contained previous forecast data | Ensure when machine learning program is ran, if there is a forcastline4/5 table already, it is cleared before new forecast data is inserted | When the machine learning program was executed it first cleared the table meaning all new data being added was added to a new table. After reviewing the tables, no old data was found | PASS |
| Timestamp consistency | Database-backend | High | Ran the machine learning program at 2:45pm and compared timestamps between rows | All timestamps across both forcastline4 and forcastline5 tables should be consistent following the Year-Month-Day Hour-minute-second format | Examined all rows in both tables and confirmed all timestamps followed the Year-Month-Day Hour-minute-second format with 30s increments | PASS |

**TRAFFIC LIGHT SYSTEM**

**EXPECTED FUNCTIONALITY:**

* After integrating ML models, the traffic light system should compare simulated live data against the upper and lower bounds generated by the Prophet model for each corresponding timestamp.
* Data points that fall within the acceptable range (within the upper and lower bounds) should be displayed in **green**.
* Data points that are within 10% of the upper or lower bound should be displayed in **amber**.
* Data points that deviate by more than 15% from the upper or lower bounds should be displayed in **red**. – again just double check if this is how tl system should be set to according to previous emails with client
* Charts should appear grey on dashboard UI if no forecasted data is available at the point of live data generation- suitable pop-up msg should be displayed on backend if this is the case

**INCORRECT FUNCTIONALITY:**

* The traffic light system does not correctly display colours according to the upper and lower bounds (e.g., data points are displayed as grey or incorrect colour).
* Data points that should be marked as red or amber are incorrectly displayed as green or not highlighted at all.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Data within bounds (GREEN) | Traffic light threshold definitions- backend API | High | Observe the home.html dashboard page for line4 | Majority if not all sensors should show current ‘live data’ with an indication of the traffic light system working | All sensors in line4 display green donut charts and all 8 sensors appear green on the pie chart | PASS |
| Data 10% near bounds (AMBER) | Traffic light threshold definitions- backend API | High |  |  |  | PASS |
| Data exceeding bounds (RED) | Traffic light threshold definitions- backend API | High | Observe the home.html dashboard page for line5 | All sensors should have an indication showing “red” by the traffic light system | When viewing line5 on the home page, all sensors' donut charts and pie charts appear in red | PASS |
| No forecast data available | Dashboard UI | High | Observe the home.html dashboard page for either line | There should be an indication telling the user the traffic light system is inactive and there should be no green/amber/red indicators on any sensors | When viewing line4 on the home page, all sensors' donut charts appear grey with no indication of a traffic light system | PASS |

**WEB APPLICATION RESPONSIVENESS**

**EXPECTED FUNCTIONALITY:**

* Sensor charts should display consistently across devices, ensuring they are visible and not partially obstructed on both desktop and tablet devices.
* The web application should be fully responsive to touch on tablet devices, with charts and controls functioning correctly when using touch gestures.

**INCORRECT FUNCTIONALITY:**

* Sensor charts do not display correctly on tablet devices, either appearing too large or partially obscured.
* The web application is not responsive on tablets, preventing users from interacting with charts or controls via touch.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Tablet UI Rendering | Dashboard UI – tablet device | High | View dashboard on tablet device | All charts are visible- no obstruction or partial cut-offs |  | PASS |
| Touch Interaction | Dashboard UI – tablet device | High | Press arrows to view donut charts in the slider for line5 | Smooth navigation |  | PASS |

**PRODUCTION MANAGER PRIVILEGES**

**EXPECTED FUNCTIONALITY:**

* Production managers should have access to an **admin panel** where they can:
  + Activate or deactivate user accounts.
  + Promote or demote users between different roles (e.g., operators to managers).
  + Delete user accounts as necessary.
  + Add or remove sensors from production lines.
* The **admin panel** should be easily navigable via the navigation bar, specifically through the "myAccount" button.

**INCORRECT FUNCTIONALITY:**

* Production managers cannot access or use the admin panel properly, with missing or malfunctioning buttons.
* The admin panel does not allow the manager to activate/deactivate user accounts or modify user roles.
* Sensors cannot be added or removed from production lines through the admin panel.
* The functionality for deleting user accounts or editing account details is unavailable or malfunctioning.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Access admin panel (production manager) | Account profile UI- admin panel | High | Login as a production operator and click "MyAccount" → Admin Panel | Admin panel opens with all functionalities visible and accessible after logging in as a production manager | Admin panels are all open and visible once logged in as an production manager | PASS |
| Access admin panel (production operator) | Account profile UI – | High |  |  |  | PASS |
| Activate user account | Account profile UI- admin panel | High | Select user-activate | User account activated – account has been approved- status no longer pending | Account status updated to “active” and visible changes reflected in the UI | PASS |
| Promote operator to manager | Account profile UI- admin panel | High | Select an operator and click “Promote” | User role changes to manager and reflected across UI and database | Operator successfully promoted to manager, visible in role list | PASS |
| Demote operator to manager | Account profile UI- admin panel | High | Select a manager and click “Demote” | User role is updated to operator, changes reflected in UI and user record | Role change is successful and saved to database, user listed as operator | PASS |
| Delete user account | Account profile UI- admin panel | High | Select a user (e.g. manager) and click “Delete” | User deleted from user\_accounts table – no longer an employee of the company | User no longer appears in the admin panel; user record is deleted | PASS |

**LOGS**:

EXPECTED FUNCTIONALITY:

INCORRECT FUNCTIONALITY:

**NON-FUNCTIONAL REQUIREMENTS**

**LIGHT/DARK MODE**

**EXPECTED FUNCTIONALITY:**

* The system should allow users to toggle between **light** and **dark modes** at any time without experiencing delays or issues.
* The default mode should be **light**, but the system should remember the user’s preference across sessions.
* The theme change should be visually consistent across all pages of the application.
* Dark mode preferences should be recorded in the database, enabling it to persist across user sessions.

**INCORRECT FUNCTIONALITY:**

* There is a delay or lag when switching between light and dark modes.
* The light/dark mode does not persist when the user logs out or reopens the application.
* The system does not apply the selected mode correctly across different pages, leading to inconsistent experiences.
* The database does not store user preferences, causing the theme to reset to default on each session.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Account profile UI -Theme toggle mode | Account profile- dark mode toggle | Low | Switch to dark mode on their account profile page | UI updates instantly with no delays |  | PASS |
| Current pages- Preference persistence | Dashboard UI- current page | Low | Reload current page after toggle | Dark mode retained |  | PASS |
| All pages- Preference persistence | Dashboard UI- any/multiple pages | Low | Navigate to other pages on the navigation bar after changing themes | Dark mode retained across all pages |  | PASS |

**ADDITION/REMOVAL OF SENSORS**

**EXPECTED FUNCTIONALITY:**

* Product managers should have the ability to add new sensors to any production line.
  + Newly added sensors will only have live data, with no historical data initially available.
* Product managers should also be able to remove sensors from any production line.
  + Once removed, the sensor and its associated chart should no longer appear in the dashboard for any user, or in the database

**INCORRECT FUNCTIONALITY:**

* Newly added sensors fail to appear in the system or do not start receiving live data.
* The removal of sensors from a production line does not immediately reflect in the user interface (UI).
* Even after removal, the chart for the removed sensor still appears to users.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Remove sensor | Account profile-admin panel  Dashboard UI | Low | Remove r06 from line5 | Live data no longer simulated for r06  R06 sensor for line5 no longer displayed on dashboard |  | PASS |
| Add sensor | Account profile-admin panel  Dashboard UI | Low | Add sensor r09 to line4 | R09 column added to line4 table of database  Live data should be simulated for r09 every 30s  R09 sensor should be displayed on dashboard when line4 selected |  | PASS |
| UI delay after removal | Dashboard UI | Low | Production Manager removes r04 (line5) and refreshes dashboard UI | Sensor vanishes immediately without requiring manual refresh |  | PASS |
| Addition of sensor during live data simulation | Account profile-Admin panel  Dashboard UI | Low | Add sensor r09 to line4 during live data simulation | New data should be generated for r09 whilst being simultaneously generated for other sensors |  | PASS |
| Removal of sensor during live data simulation |  |  |  |  |  | PASS |
| Bulk removal of sensors | Account profile – admin panel  Dashboard UI | Low | Bulk remove 6 sensors (r01-r06) from line5 | Live data no longer simulated for deleted sensors  All deleted sensors no longer displayed on dashboard UI |  | PASS |

**PERFORMANCE**

**EXPECTED FUNCTIONALITY:**

* The system should display live data immediately when a user accesses the dashboard.
* Data charts should update every 30 seconds without delays, ensuring the information presented is always up-to-date.
* There should be no significant performance degradation even with a high volume of sensors or data points.

**INCORRECT FUNCTIONALITY:**

* The data charts are slow to load or update, resulting in delays or empty charts.
* Performance degrades significantly with a large number of sensors, leading to slow responses or system crashes.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | Corresponding system feature | Priority Level | Input provided by user | Expected output | Actual Output | Test Outcome |
| Dashboard Initial Load | Dashboard UI | High | Open dashboard | All charts render with live data in <10 seconds |  | PASS |
| Live data update | Dashboard UI | High | Observe dashboard for 30 seconds | Charts refresh automatically every 30s |  | PASS |
| Empty sensor-data page | Dashboard UI | High | Select r07 sensor for line4 | Graphs should immediately be populated with data- both current and live- depending on selecting live data and not appear blank or take >10 seconds to load |  | PASS |

**RELIABILITY**

**EXPECTED FUNCTIONALITY:**

* The system should handle fluctuations in live data smoothly, without unexpected errors or inconsistencies.
* Traffic light system behavior should always accurately reflect the simulated data, with consistent and reliable color coding (green, amber, red).
* The system should have uptime reliability, with minimal or no downtime during operation.

**INCORRECT FUNCTIONALITY:**

* Data inconsistencies or errors occur due to poor handling of fluctuating live data.
* The traffic light system fails to show accurate results or behaves unpredictably.
* The system experiences excessive downtime or crashes frequently, affecting user experience.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Data fluctuation handling | Database-backend | High | Observe r05 column of line4 table in database | Simulated live data should fluctuate randomly 2% above and below around average for line4 |  | PASS |
| Crash recovery | API – backend  Dashboard UI | High | Kill server process | Restart API after a minute- live data simulation restarts and previously simulated live data is not lost |  | PASS |

**USABILITY**

**EXPECTED FUNCTIONALITY:**

* The system should be intuitive and easy to navigate for all users, with clear layout, readable fonts, and well-defined buttons and actions.
* A help guide should be available at all times and should be easy to access from the navigation bar.
* The help guide should contain accurate, up-to-date, and concise instructions on how to use the system.

**INCORRECT FUNCTIONALITY:**

* The interface is cluttered or difficult to understand, causing confusion for users.
* The help guide is outdated or hard to navigate, leading to users being unable to find relevant information.
* Buttons or navigation elements are not clearly defined, making it difficult for users to interact with the system.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Dashboard UI- Help Guide page | Dashboard UI- Help Guide page | Low | Inspect traffic light description & FAQ section | Instructions match current UI with no discrepancy to current functionalities |  | PASS |
| Dashboard UI- Help Guide page accessibility | Dashboard UI- Help Guide page via nav bar | Medium | User can access help-guide from navigation bar when on other pages – e.g. historical data page | Guide opens |  | PASS |
| Dashboard UI- Layout | Dashboard UI- help page | High | When in dark mode, inspect the x and y-axis for scatter graphs for readability | Scales should be white and easy to read in dark mode, ensuring clear colour contrast |  | PASS |

**DATA PRIVACY**

**EXPECTED FUNCTIONALITY:**

* The system should ensure that only necessary user data (worker ID, email, name, hashed password) is collected and stored.
* User data should be encrypted or hashed using industry-standard algorithms to protect sensitive information.
* The application should comply with data privacy regulations and ensure that user data is only accessible by authorised users.

**INCORRECT FUNCTIONALITY:**

* The system collects unnecessary user data, such as phone numbers or addresses, without proper consent.
* Passwords or other sensitive data are stored in plain text or using weak hashing algorithms, making them vulnerable to exposure.
* The system fails to meet data privacy standards, exposing user data to unauthorized access.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CASE** | **Corresponding system feature** | **Priority Level** | **Input provided by user** | **Expected output** | **Actual Output** | **Test Outcome** |
| Hashed passwords | Database-backend User\_accounts table | High | Database-User\_accounts table  Inspect API for hashing algorithm implementation |  |  | PASS |
| Hashing algorithm implementation | API- backend | High | Inspect API for hashing algorithm used |  |  | PASS |
| Data information storage | Database-backend User\_accounts table |  |  |  |  | PASS |