

---

# REQUIREMENTS DOCUMENT

---

## Team Identification:

Team Number: 8

## Project Information:

Project Choice: Project 2

## Team Members:

Full Name	UoB Number	UoB Email Address
Adam Hamza Khan	22009241/1	h.a.khan6@bradford.ac.uk
Aliya Iqbal	21009042/2	a.iqbal116@bradford.ac.uk
Azeem Hussain	21004184/2	a.hussain418@bradford.ac.uk
Bilal Shahid	23012993/2	b.shahid@bradford.ac.uk
Humairah Ali	23006977/2	h.ali129@bradford.ac.uk
Shuaib Syed Shah	22010295/1	s.shah50@bradford.ac.uk
Yaasmeen Abdulkareem	23028337/1	y.abdulkareem@bradford.ac.uk
Zainah Mahmood	22007256/1	z.mahmood38@bradford.ac.uk

### A.1.1. INTRODUCTION

The City of Bradford Metropolitan Council necessitates a web interface that enables users to interact with a map of assets by utilising a suitable JS/PHP mapping library. This utility is designed to facilitate the efficient visualisation, management, and analysis of asset-related activities within the district. The system will enable users to add new assets, update details, and monitor job counts recorded against each asset, as well as to categorise assets, create custom category types, and display selected assets on a map. Furthermore, it will facilitate the use of data visualisation tools such as heatmaps, search, filtering, and task summarisation to improve usability. The system will facilitate the tracking of asset status, the logging of tasks, and the identification of high-activity areas by mapping and categorising assets. This will result in improved decision-making, resource allocation, and service efficiency. This specification document was created using an initial scenario and an interview with the client, during which the key functionalities, database structure, and user interface were discussed.

### A.1.2. TEAM EXPERTISE

Our team consists of individuals with diverse skills, each contributing to different aspects of the project. Below is an overview of the expertise brought by each member:

- **Adam Hamzah Khan (Team Leader)** – Provided good leadership, managed project coordination, ensured tasks were completed on time and contributed to the making of Gantt chart and worked closely with Azeem to code the index of the web interface. Expertise in project management, recruitment analysis and backend development.
- **Aliya Iqbal (Team speaker)** – Communicated team progress efficiently, documented requirements and contributed to the UI design and Gantt chart. Furthermore, Aliya coded the functions of the web interface. Her strong communication skills, attention to detail in documentation and talent for UI development were an invaluable asset to the team.
- **Zainah Mahmood (Secretary)** – Responsible for meeting minutes, project introduction and contributed to Gantt chart, ensuring clarity in documentation and tracking project discussions. Zainah assisted in coding the register page with Humairah, helping with debugging the code and refining components. She was also responsible for developing the setting page of the interface.
- **Humairah Ali** - Contributed to the UI design and Gantt chart and worked on ethical considerations (LSEPI) and risk & economic aspects. Along with Zainah, she was involved in developing and debugging the register page. Ensured compliance and feasibility of implementation. Expertise in ethical analysis, risk assessment and UI styling.
- **Yaasmeen Abdulkareem** - Managed achievements and scoring contributions while contributing to the UI design and towards the Gantt chart. Yaasmeen was responsible for developing and debugging the login/logout page as well as the categories section of the web interface. Excellent at evaluating progress, ensuring quality control and implementing UI elements.
- **Bilal Shahid** – Led the reasoning behind the topic choice and data description, ensuring logical structuring of information. Furthermore, coded the assets page. Strong analytical, research skills and data-handling skills.

- **Shuaib Syed Shah** – Conducted research on existing solutions, tools, equipment, helping with informed decision-making and contributed to coding the backend logic. Strong research, analytical skills API integration, and backend development skills. He was also responsible for setting up the database, writing the SQL queries, handling JavaScript and the main CSS of the web interface; ensuring data was stored and retrieved efficiently.
- **Azeem Hussain** – Worked on functional and non-functional requirements, ensuring technical feasibility and clarity. Helped with database structuring. Strong problem-solving, analytical thinking and database management skills. Furthermore, worked with Adam to code the index section of the project.

## FINAL REFLECTION

- Throughout the first part of the project, we found that backend coding was the most technically challenging part, requiring debugging and testing. However, every team member played a crucial role in balancing different aspects of the project, from UI design to documentation and research.
- One of biggest strengths was communication whenever challenges arose, we supported each other and found solutions together. If we could improve anything, it would be better integration between front-end and backend components earlier in the project to streamline the final development phase.

### A.1.3. RATIONALE

The reason why our team chose this topic is because Bradford, like all cities, develop over time which makes managing their resources—including free services and infrastructure — more difficult. We can assist the City Council in effectively tracking, maintaining, and arranging these assets in a data-driven manner by developing a basic online interface. Individuals will be able to allocate resources, prioritize maintenance, and visualize asset locations in a streamlined manner with the help of this system. The objective is to make the city's resource management more accessible and transparent while guaranteeing quick repairs, better planning, and overall improved services for people.

Additionally, we chose this topic because it provides an opportunity to be up to date with modern day technology. By developing a user-friendly web interface, we can ensure that Bradfords assets management is more efficient and saves time. The system allows individuals to easily categorise assets, track activities such as maintenance, and to respond to issues effectively. As cities continue to grow, its essential that their infrastructure management continues to evolve with modern day technology. This project will not only improve current operations but also sets out the foundation for future development and sustainability.

### A.1.4. LITERATURE REVIEW

This section provides a brief overview for our prototype that is using PHP, HTML, JavaScript, MySQL, Google Authenticator and SMFTP. These will be mainly used for the authentication, data management and email communication. This system aims to provide secure user authentication, database storage as well as email notifications using best practices.

#### Overview

Several web applications use a combination of PHP, JavaScript, and MySQL for backend and fronted development. Additionally, Google Authenticator is a widely accepted Multi-Factor Authentication solution, while SMFTP ensures purely just email communication.

#### ***Similar Systems and Tools***

1. Content Management Systems (CMS):
  - WordPress: This uses PHP and MySQL for database management and integrates the use of
  - Google Authenticator this is used for the MFA and supports SMTP for emails.
  - Joomla: This Provides authentication plugins and secure email handling via the use of SMTP.
  - Drupal: Uses PHP and JavaScript with MySQL, supporting modular authentication.
2. Authentication Platforms:
  - Google Authenticator: This provides TOTP-based MFA for login security.
  - Auth0: this is a cloud-based authentication service that support the use of MFA and secure logins.
3. Email Communication Services:
  - SMTP this stands for (Simple Mail Transfer Protocol): This is used for sending verification and password resets emails too the users / clients.
  - SendGrid & Mailgun: This is a third-party service for handling automated email transactions.

#### Role of the Selected Technologies

Technology	Purpose
PHP	Used for Backend processing and server-side logic
HTML & JavaScript	Used for Frontend UI/UX and client-side scripting
MySQL	Used for Database management for storing user details
Google Authenticator	Used for Multi-factor Authentication (MFA)
SMFTP	Used for Securing email notifications such as password resets

## INTERIM PROTOTYPE IMPLEMENTATION

### Development Environment

<u>Component</u>	<u>Technology</u>
Web Server	Apache or Nginx
Database	MySQL
Programming languages	PHP, HTML, JavaScript
Security APIs	Google Authenticator API
Email API	SMTP

### Prototype Features

- User Registration and Login
- Database Setup (MySQL Schema for Users, Roles, and Logs)
- Google Authenticator integration for MFA
- Email Sending Via SMTP for password resets.
- Basic Admin Panel for Managing Users.

## FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

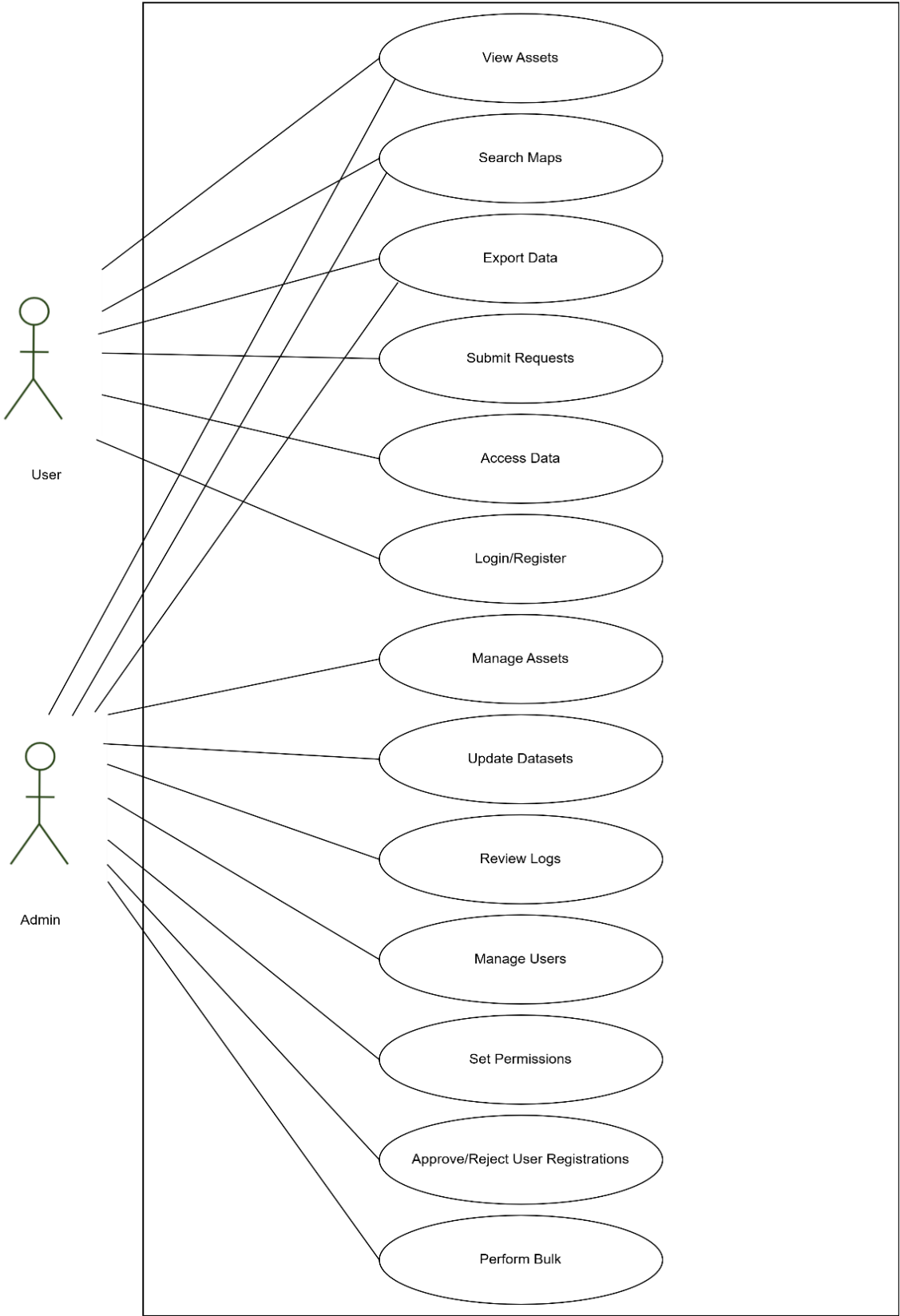
### Users

- **View and filter assets:** Users can view the list of assets and apply filters to narrow down the displayed results based on various criteria.
- **Search maps:** Users can utilise search functionalities to locate specific assets on interactive maps.
- **Export data:** Users can export asset data for external use.
- **Submit asset requests:** Users can submit requests for new assets or modifications to existing assets.
- **Access departmental data:** Users can access data related to specific departments as needed.
- **Registration screen for user accounts:** New users can register their accounts through a dedicated registration screen.

### Admins

- **Manage assets:** Admins can add, update, and delete assets as needed.
- **Update datasets:** Admins can update asset datasets to reflect the most current information.
- **Review logs:** Admins have access to logs to review user activities and system events.
- **Manage users:** Admins can manage user accounts, including creating, updating, and deleting accounts.
- **Set permissions:** Admins can set permissions to control user access to different system functionalities.
- **Approve or reject user registrations:** Admins are responsible for approving or rejecting new user registrations.
- **Perform bulk actions:** Admins can efficiently perform bulk actions such as batch approvals, rejections, or deletions.

web interface to interact with assets



# Functional Requirements

## **MAP DISPLAY WITH ASSET INTEGRATION**

Integrate a JavaScript/PHP library to create interactive maps that can load and display asset data dynamically. Asset information, including latitude and longitude, will be loaded from a CSV file, and displayed on the map, allowing users to visualise asset locations in real time.

## **ASSET MANAGEMENT**

The system will enable comprehensive asset management, allowing users to categorise assets beyond the provided CSV data. Users can create and manage custom categories, select, and display assets, add new assets, and log job counts per asset. Administrators will have the authority to approve or reject modifications to assets, ensuring data integrity and oversight.

## **DATA INTEGRATION**

The system will support data integration by importing and parsing CSV/Excel files to extract asset data. This data will be stored in a structured database with category associations. An optional API will be available for dynamic data updates and external integrations. Data validation rules will be specified to handle missing latitude/longitude, duplicates, and incorrect formats, and the maximum CSV file size the system can manage will be clarified.

## **USER INTERFACE**

The user interface will be designed to provide an intuitive experience for asset visualisation. It will include search and filtering capabilities with auto suggestions and ensure compliance with Bradford Council's branding and colours. The interface will align with accessibility standards (WCAG 2.1), be responsive on mobile devices, and include error handling for incorrect inputs, such as invalid asset searches or CSV format errors.



# Non-Functional Requirements

## **PERFORMANCE**

Performance requirements ensure that the system operates efficiently, providing a smooth user experience without delays. This includes optimising map rendering and data loading speeds to ensure that assets are displayed quickly. To achieve this, the system should define the expected system load, specifying the number of concurrent users and assets that can be displayed without affecting performance. Techniques such as lazy loading and caching mechanisms should be implemented to load only necessary data at a time, reducing server load. Additionally, database indexing and optimised queries should be used to prevent slowdowns when fetching asset data, ensuring quick search and retrieval functionalities.

## **SCALABILITY**

Scalability ensures that the system can handle an increasing number of assets and users over time without performance degradation. As the asset database expands and more users access the system, it should be able to scale seamlessly by using efficient data structures, cloud computing solutions, and load-balancing techniques. This allows the system to adapt to growing demands without requiring a complete redesign.

## **SECURITY**

Security is critical for protecting user data and system integrity. The system must implement authentication and authorisation mechanisms, ensuring only authorised users can access specific features. Data storage and transmission should be secured using AES-256 encryption for data at rest and SSL/TLS encryption for data in transit. Additionally, security features like password reset functionality and Multi-Factor Authentication (MFA) should be implemented to enhance account security. To prevent unauthorised access, session timeouts should be defined for inactive users. Role-Based Access Control (RBAC) should be enforced to grant appropriate permissions based on user roles, while audit logs should track user and admin activities for accountability.

## **USABILITY**

Usability ensures that the system is user-friendly and accessible to a diverse range of users. A well-designed interface should be intuitive, allowing users to interact with the system efficiently and effectively. Compatibility with screen readers and assistive technologies should be ensured to accommodate users with disabilities. Following best practices in UI/UX design helps enhance engagement and reduce learning curves for new users.

## **MAINTAINABILITY**

Maintainability refers to the system's ability to be easily updated, modified, and debugged. This requires developing clean, modular, and well-documented code, making it easier for developers to troubleshoot issues and implement updates. By following coding best practices and using standardised frameworks, the system can support future enhancements and long-term maintenance.

## **TESTING AND QUALITY ASSURANCE**

Testing ensures that the system functions correctly and meets user expectations. The system relies on external JS/PHP libraries for map display and structured CSV files for asset data, meaning these dependencies must be tested thoroughly. Constraints such as alignment with gov.uk design standards must also be met. Different testing strategies should be implemented, including unit tests (checking individual components), integration tests (verifying different modules work together), and user acceptance testing (ensuring the system meets business needs).

## **SECURITY & COMPLIANCE**

To comply with data protection regulations and ensure robust authentication and authorisation, the system must follow industry standards for data privacy and security. Encryption techniques should be applied to protect sensitive data at rest and in transit, reducing the risk of cyber threats or data breaches.

## **PERFORMANCE & SCALABILITY**

The system must be designed to handle high traffic and large datasets efficiently, preventing slowdowns or crashes. Implementing load balancing, database optimisations, and cloud-based solutions ensures that system performance remains consistent under varying workloads.

## **USER INTERFACE & UX DESIGN**

A well-structured User Interface (UI) and User Experience (UX) design is essential for accessibility and ease of use. The system must adhere to gov.uk and Bradford Council branding, ensuring a consistent and professional look. Additionally, compliance with WCAG 2.1 accessibility standards guarantees that users with disabilities can navigate and use the platform effectively.

## **DATA MANAGEMENT**

Data management ensures that all asset information is securely stored, organised, and backed up. Backup and recovery mechanisms should be in place to prevent data loss due to system failures or cyberattacks. The system should also provide efficient data retrieval and storage techniques to maintain high performance.

## **TESTING & QUALITY ASSURANCE**

Comprehensive testing strategies are required to ensure system reliability and compatibility across different devices and browsers. This includes conducting unit, integration, and user acceptance testing to validate that each feature works as expected. Load testing should assess the system's ability to handle high traffic, while cross-browser testing ensures a consistent experience across different web platforms.

## DATA DESCRIPTION

Data description refers to the process of summarizing and explaining the characteristics of a dataset. It provides insights into the structure, meaning, and key attributes of the data. The relationships between variables, the existence of outliers, and any missing values are frequently highlighted in data descriptions. It also includes data visualizations that aid in identifying patterns and trends, such as box plots and histograms. Forming hypotheses, choosing suitable analytical techniques, and making sure the data is prepared for more intricate modelling or decision-making procedures all depend on this descriptive process. This can include:

### METADATA

Metadata is crucial information that explains a database's properties, organization, and context. It offers useful information that promotes proper data interpretation, use, and understanding by users. The dataset's column names, which identify the variables or qualities contained within the data, and the data types, which indicate whether a column contains textual, numerical, categorical, or other sorts of data, are important elements of metadata.

Additionally, metadata describes the dataset's sources, including surveys, sensors, and third-party databases. This guarantees access and helps in evaluating the reliability and accuracy of the data. Additionally, metadata frequently contains information about the dataset's quality, including its completeness, accuracy, and any known biases or limits.

### STATISTICAL SUMMARY

Statistical summary measures are essential tools for analysing and describing data sets. The **mean** is the average of a set of numbers, calculated by adding all values and dividing by the number of values. It provides a central value for a distribution but can be skewed by extreme values (outliers). The **median** is the middle value when data is ordered, dividing the dataset into two equal halves. It is less affected by outliers, making it a better measure for skewed distributions. The **mode** refers to the value that occurs most frequently in the data set. A dataset may have one mode (unimodal), more than one mode (bimodal or multimodal), or no mode at all.

The **standard deviation** measures the spread or dispersion of data points from the mean. A higher standard deviation indicates more variation, while a lower value suggests that data points are closer to the mean. The **range** is the difference between the highest and lowest values in a dataset, offering a simple way to understand the extent of data variation. These measures collectively help summarize the characteristics of a data set, aiding in decision-making, comparisons, and further statistical analysis.

### DATA DISTRIBUTION

Data distribution describes how data points spread over a dataset's many values or categories. To understand the basic framework of the data, it offers insights into patterns, trends, and variability. Several tools facilitate the visualization of these distributions, which facilitates data interpretation and analysis.

The histogram, which shows the frequency of data points inside specified ranges (bins), is one of the most often used visualization tools. The height of the bar indicates how many data points fall within each bin, which represents a range of values. When displaying the distribution of continuous data, histograms are very helpful in determining the dataset's skewness, spread, and central tendency. They also highlight any gaps or outliers in the data.

Another effective tool for showing data distribution is a box plot, sometimes referred to as a box-and-whisker plot. This is particularly useful when comparing different datasets. It displays a dataset's median, quartiles, and possible outliers. The "whiskers" show the range in which most of the data fall, the box shows the interquartile range (IQR), and the line inside the box indicates the median. Box plots are especially useful for identifying outliers and detecting data spread.

When summarizing categorical data, a frequency distribution table is a straightforward yet powerful tool. It displays the frequency of each value or category in the dataset. The frequency of each unique value is presented, which aids in determining the most prevalent categories and evaluating the data's overall distribution.

When combined, these visualization tools give researchers and data analysts a thorough grasp of the distribution of data, enabling them to make valid choices.

## **DATA TYPES**

Programming and data analysis depend heavily on data types since they specify the types of data that may be saved and processed. They assist in determining the amount of memory needed and the operations that can be carried out on the data. Data types are typically divided into three primary categories in the context of data analysis: text, numerical, and categorical.

**Numerical Data Types:** Quantitative data is represented using numerical data types. They fall into one of two categories:

Variables that assume discrete, independent values—often counts—are included in this category. The number of cars in a parking lot or the number of students in a class are two examples.

Variables of the continuous data type are those that have an endless range of possible values. Time, temperature, weight, and height are a few examples.

**Categorical Data Types:** These data types describe traits or qualities and are used to express qualitative data. These are divided into two more categories:

Variables that reflect categories without a natural order or ranking are known as nominal data. Variables that describe categories with a meaningful order or ranking but no clear distinction between them are known as ordinal data.

**Text Data Types:** Strings or character sequences are stored in text data types. These are frequently used for textual information of any type, including names and addresses. Text data can range in length from a single word to lengthy paragraphs and is frequently stored as strings.

In conclusion, data analysis and modelling require an awareness of how variables are categorized into numerical, category, and text data categories. Choosing the appropriate statistical techniques or machine learning algorithms for data analysis is aided by proper classification.

## **MISSING VALUES**

The term "missing values," sometimes referred to as "null data," describes a dataset's undefined or missing values. This can happen for several reasons, including intentional absence of numbers, data unavailability, or mistakes made during data entry. Incomplete data can distort results and lower model reliability, so it's critical to identify and handle missing values in data analysis.

Data analysts usually use methods and tools found in programming languages like R or Python to find missing values. To identify null values in datasets, the Pandas library in Python, for instance, provides functions like `isnull()` and `isna()`. Boolean values, which indicate whether each value is missing, are returned by these functions.

The next step after identifying missing values is to handle them. Typical methods for dealing with missing data consist of:

Delete any columns or rows that contain missing values. When there is little to no missing data that has no apparent effect on the dataset, this is helpful.

Imputation: Use statistical techniques to fill in missing values, such as substituting the column's mean, median, or mode. More sophisticated techniques like machine learning algorithms or regression can also be applied.

Leaving them alone: If a machine learning model is capable of handling null values directly, it may be possible to let them alone in particular circumstances.

## USER INTERFACE

Log in/Register:

[Register](#)

Register

First Name

Last Name

Email

Password

Confirm Password

Address Line 1

Address Line 2

Town

County

Postcode

Local Area Status

☐ I live in the local area

☐ I work in the local area

☐ I study in the local area

Register



[Home](#) [Reports](#) [Assets](#) [Categories](#) [Settings](#)

[Login](#)

[Sign In](#)

Search this site



Login

Email

Sohal56@bradford.ac.uk

Password

\*\*\*\*\*

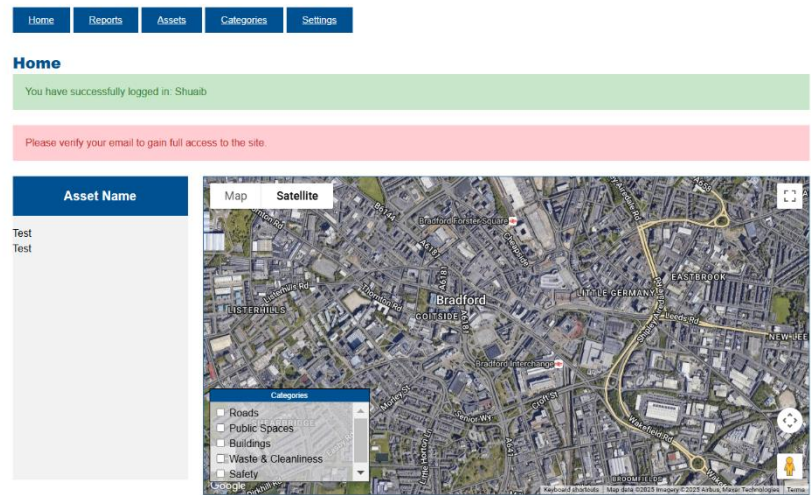
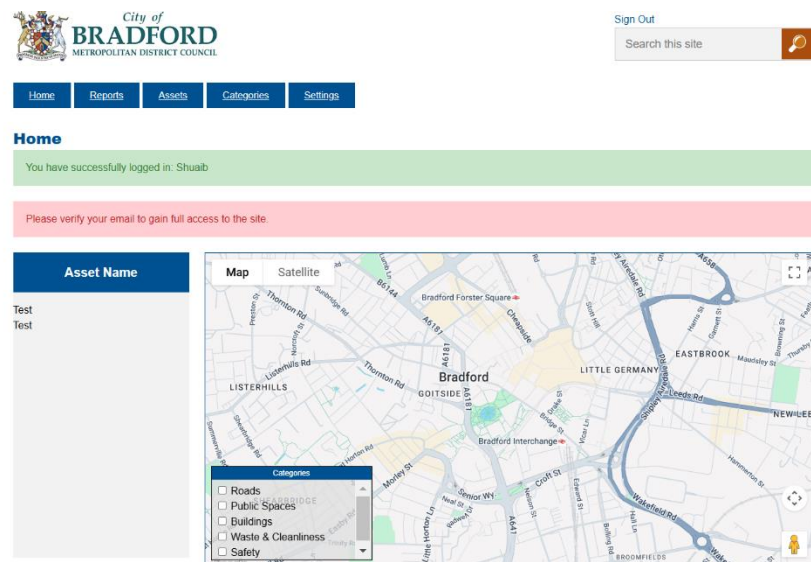
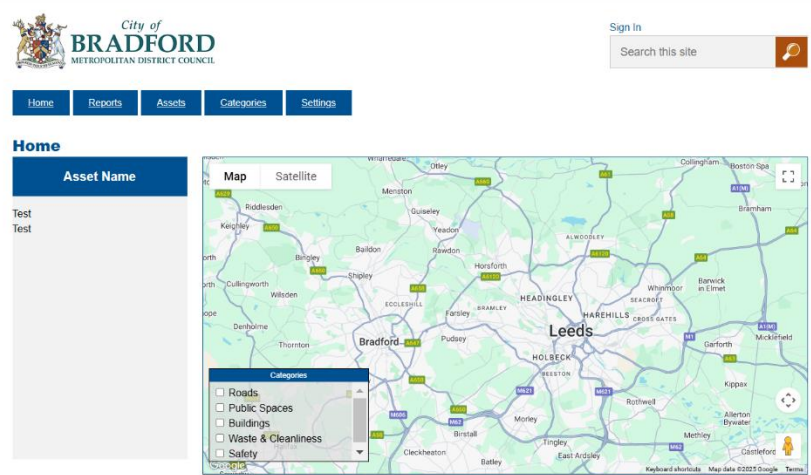
Login

Forgotten Password


OR

Create account

Home page:



Assets:



City of

BRADFORD

METROPOLITAN DISTRICT COUNCIL

Home

Reports

Assets


Categories

Settings

Assets


Sign In

Search this site



© 2025 City of Bradford Metropolitan District Council

Categories:



City of

BRADFORD

METROPOLITAN DISTRICT COUNCIL

Home

Reports

Assets


Categories

Settings

Categories

Sign In


Search this site



© 2025 City of Bradford Metropolitan District Council



Reports:



City of

BRADFORD

METROPOLITAN DISTRICT COUNCIL

Home

Reports

Assets

Categories

Settings


Reports

Sign In

Search this site

© 2025 City of Bradford Metropolitan District Council

Settings:



City of

BRADFORD

METROPOLITAN DISTRICT COUNCIL

Home

Reports

Assets

Categories

Settings

Settings

Sign In

Search this site

© 2025 City of Bradford Metropolitan District Council

Example street view:



## RISK ASSESSMENT

The project uses geographical data to show asset locations on a map. Potential hazards which are related to creating a web-based asset management interface for the City of Bradford Metropolitan Council are identified in this risk assessment. By addressing these risks, we can ensure that responsible and secure features and methods are implemented into the system.

### Risks:

Legal Issues	Social Issues	Ethical Issues	Professional Issues
Data Privacy and Protection	Accessibility	Data Ownership	System Downtime and Dependency
Data security Measures	Transparency and Fairness	Fairness in Asset categorisation	Cybersecurity
Liability and Data Accuracy	User adoption and engagement	Quality Assurance	User Authentication & Data security

#### **DATA PRIVACY AND PROTECTION:**

The system will keep and process sensitive and personal data. If the data is handled unlawfully, it would violate the UK Data Protection Act 2018. The impact of this would result in penalties and loss of trust. To avoid breach of personal data, ensure to anonymise sensitive data where possible. Encrypt sensitive data and provide a clear consent procedures on how a user's data is being used.

#### **DATA SECURITY MEASURE:**

Implementing robust security measures, such as access controls and regular backups, is essential to protect users' information from unauthorised access. Compliance with relevant data protection laws and regulation should be ensured. Access controls restrict data access to authorised personnel only. Frequent backups decrease the chance of data loss from cyberattacks or system breakdowns.

#### **LIABILITY AND DATA ACCURACY:**

Two important concerns while developing the web interface are liability and data accuracy. Inaccurate data on the map may cause users to make poor decisions, misuse the system, and compromise its reliability. For example, displaying an asset's location incorrectly could ultimately provide the user with inaccurate information and result in inefficiencies or even community safety risks. You may be responsible for any errors especially if the data is utilised for important business processes or decision-making. Therefore, it is important that the data is updated regularly, verified and cross-checked for accuracy before displaying it on the web interface.

#### **ACCESSIBILITY:**

The application should be designed so that it is accessible to all users, considering any physical abilities. To do this the application must comply with the accessibility standards and guidelines such as the WCAG (Web Content Accessibility Guidelines). To ensure that all users, regardless of ability, have an inclusive experience, features like keyboard navigation, screen reader compatibility and alternate text for images. The system should also accommodate assistive technologies like speech recognition software's.

#### **TRANSPARENCY AND FAIRNESS:**

Fairness and transparency are important for gaining the user's trust. User's must be given the opportunity to provide their consent and be fully informed about how their personal data is being gathered, utilised and stored by the system. To prevent any prejudice, users should also be informed about the classification and representation of the assets and associated data. Users should have access to their data and should be able to adjust. The platform should respect their right to control their own data.

#### **USER ADOPTION AND ENGAGEMENT:**

The system's success depends on the user involvement and how easy it is to adopt the web interface. If it is difficult to use many consumers could hesitate to use it. The system needs to be simple to use and clear to encourage adoption. This would increase engagement, especially if it delivers relevant features and offers useful directions. Maintaining user engagement can also be achieved by offering continuing support, gathering user input and issuing frequent updates in response to this input.

#### **DATA OWNERSHIP:**

As mentioned before, users should have control over their own data and they should be informed about how it is being used, stored and shared within the system. Prior consent should be obtained to collect this data before it is collected- usually via a popup. In accordance with their ownership rights, people or organisations must be able to view, update or remove their data without any interference. To avoid misuse, the system should include explicit data ownership policies and procedures that outline how data will be shared, kept or used.

#### **FAIRNESS IN ASSET CATEGORISATION:**

When creating the system, assets must be categorised honestly to prevent any unfair disadvantage or misinterpretation of any organisation. It ensures that every asset is handled properly. To avoid any discrimination being unintentionally included into the system, each algorithm used to categorise assets must undergo routine audits. User's must also have the confidence that all categories are well-supported and justified and no group is favoured over another. By making sure that everyone feels treated fairly and without bias, transparent categorisation enforces user's trust.

#### **QUALITY ASSURANCE:**

Since inaccurate data can result in poor decision-making, it is imperative that the data presented is accurate and current. Before the system is used by the user, possible faults in its functioning, design and data can be found and fixed. This can be done by carrying out routine audits, checking for errors and confirming that every feature functions as planned. The system must be tested under real-world circumstances to fulfil the needs of a wide range of users. Furthermore, to increase the system's dependability, you should make sure that any updates do not unintentionally harm the quality of the interface.

#### **SYSTEM DOWNTIME AND DEPENDENCY:**

This describes a faulty system due to server failures or technological difficulties. System dependency and downtime can cause serious problems since extended outages can result in service interruptions and monetary losses. Implementing a strong infrastructure and monitoring tools is necessary to mitigate these risks and guarantee system reliability and reduces downtime. Reducing the impact of dependency can be achieved by providing backup solutions during outages.

#### **CYBERSECURITY:**

Cybersecurity for the web interface guarantees confidentiality and integrity of data held in the system. Maintaining strong cybersecurity standards, is essential to guard against breaches, hacking attempts as the web interface will store sensitive data. Inadequate cybersecurity safeguards can result in the user's personal data being accessed without authorisation and ultimately resulting in legal issues. Encryption, firewalls and frequent security audits are an important part of cybersecurity. Reducing the vulnerabilities requires keeping the system updated with recent security updates.

#### **USER AUTHENTICATION:**

A strong user authentication method should be implemented, such as multi-factor authentication, to help prevent unauthorised access to the system and safeguard sensitive data from malicious actors like cyber attackers and data breaches within the application. Strong authentication and data security procedures are put in place to safeguard user information, lower the possibility of breaches and uphold the system's credibility.

## WORK PLAN

[illegible]

## GITHUB

### Prototype

<https://github.com/ahuss418/Enterprise-Pro/tree/main/Software>

### Meeting Minutes

[https://github.com/ahuss418/Enterprise-](https://github.com/ahuss418/Enterprise-Pro/blob/main/NDA%20%26%20Minutes/team%20mintues.docx)

[Pro/blob/main/NDA%20%26%20Minutes/team%20mintues.docx](https://github.com/ahuss418/Enterprise-Pro/blob/main/NDA%20%26%20Minutes/team%20mintues.docx)

### NDA

<https://github.com/ahuss418/Enterprise-Pro/blob/main/NDA%20%26%20Minutes/NDA%20.docx>

## PEER REVIEW

Team Member	Contribution	Achievements	Scoring
Adam	Managed team coordination, Gantt Chart, Functional/ non-functional requirements, and assisted in coding the index page of the web interface.	Ensured smooth workflow, timely progress and contributed to key coding logic.	10
Aliya	Team representation, Requirement document, Gantt chart and user interface. She was also responsible for coding and debugging the functions of the web interface.	Communicated progress effectively, contributed to planning and implemented UI features.	10
Humairah	User interface, Gantt chart ethics (LSEPI), Risk and economic aspects, UI styling. Furthermore, assisted in developing and debugging the register page.	Contributed ethical insights, ensured accessibility and risk assessment.	10
Yaasmeen	Team expertise, User interface, Gantt chart, achievements, coding/ debugging the login & logout and categories page.	Balanced multiple tasks, ensured quality control and high UI quality.	10
Zainah	Introduction, Team meeting minutes, Gantt chart and assisted in developing and debugging the register page. She was also responsible for developing the settings page.	Kept track of discussions, ensured clarity in documentation and tested code components.	10
Azeem	Functional/non-functional requirements and database structuring. He also assisted in developing and debugging the index of the web interface.	Focused on technical clarity and feasibility. Assisted in database setup.	10
Bilal	Rationale (reasoning) of topic choice, data description and data processing scripts. Developed the assets page of the web interface.	Provided structured analysis, researched data requirements and handled data processing efficiently.	10
Shuaib	Literature review (similar or existing solutions, tools, and equipment) and backend coding.	Gathered relevant information for informed decisions, developed robust backend logic and API integration.	10