# Knowledge Transfer Report

DRP - Team 7

# Knowledge Transfer Report

Table of Contents

# 1. Purpose

## 1.1. Scope

1.1.1. The knowledge transfer includes infrastructure structure, database structure, tools used for front-end and back-end development, structure of website functionalities, and developmental environment portability.

1.1.2. The knowledge transfer excludes but not limited to integration of the infrastructure into pre-existing the VTN and VEN and deployment outside the local environment.

## 1.2. Terminology

1.2.1. Docker: A platform that allows developers to package applications into containers, ensuring consistency across environments.

1.2.2. OpenADR: An open protocol for automated demand response that facilitates communication between the Demand Response Portal and users.

1.2.3. Typhoon HIL: High-level integration layer used for communication and data exchange, ensuring seamless operation between providers and consumers.

1.2.4. ARIMA Models: Statistical models used for predictive analytics within the portal to forecast energy needs and grid management.

1.2.5. Figma: A design tool used for creating UI/UX mockups for the portal, ensuring a user-centric design approach.

1.2.6. Demand Response (DR): A strategy to adjust or reduce electricity usage by end-users during peak demand periods in response to pricing signals or incentives.

1.2.7. Virtual End Node (VEN): A client entity in a DR system that receives and responds to event information, representing the consumer's equipment or systems.

1.2.8. Virtual Top Node (VTN): A server entity in a DR system that sends out event information to VENs, typically operated by utility companies or grid operators.

1.2.9. DR Event: A specific period during which power consumers are requested to reduce or shift their electricity usage.

## 1.3. References

1.3.1. The link of the whole project is at this link: "https://github.com/tntech-se-drp/drp". To install it as a developmental environment, go to "drp>src>documentation>initial-setup.md". The "initial-setup.md" has a short, easy to follow step-by-step instructions.

# 2. Key Personnel

## 2.1. Business Users

2.1.1. Roles and Responsibilities:

2.1.1.1. Providing Feedback: Business users are essential in offering practical insights into the system's usability, functionality, and overall performance. They help in identifying areas for improvement from a real-world application standpoint.

2.1.1.2. Validating Business Processes: They ensure that the portal's features align with existing business processes and facilitate seamless integration into daily operations.

2.1.1.3. Engaging in Demand Response Activities: By actively participating in demand response events through the portal, business users help to refine the system's effectiveness in real-time scenarios.

2.1.1.4. Contributing to Continuous Improvement: Through their interaction with the portal, business users contribute valuable data that can be analyzed for further enhancements of the platform.

2.1.1.5. Promoting Adoption: As primary users of the portal, business users aid in promoting the adoption of the system within their organizations, showcasing the benefits and efficiencies gained through its use.

2.1.2. Business users are responsible for leveraging these tools to make informed decisions regarding energy management, contributing to the overarching goal of creating a smarter and more sustainable energy grid.

## 2.2. Subject Matter Experts (SMEs)

2.2.1. The roles of subject matter experts contributing to the knowledge transfer include:

2.2.1.1. Lead Back-End Developer / Database Administrator: Oversees the back-end development and database management. This SME has deep knowledge in MariaDB database administration, optimization, and security. They are responsible for the architectural integrity and data management of the back-end systems.

2.2.1.2. Lead Front-End Developer: Shapes the user interface and experience. The expertise lies in HTML, CSS, JavaScript, and using Bokeh for dynamic data visualization. They ensure that the application is intuitive and responsive.

2.2.1.3. Lead Security Specialist: Focuses on the application's security posture. With expertise in cybersecurity principles and practices, they implement security measures to protect against threats and vulnerabilities.

2.2.1.4. Lead Data Scientist: Brings expertise in data analytics and modeling. They are adept at using statistical models like ARIMA for predictive analytics, contributing to the system's intelligence and decision-support capabilities.

2.2.1.5. Scrum Master: As a facilitator for the Agile development process, this SME ensures that the team adheres to Agile methodologies, meeting project goals efficiently and effectively.

2.2.1.6. Data Scientist: Supports the Lead Data Scientist with data analysis tasks. This SME is skilled in algorithm development and data processing to enhance the portal's analytic functions.

2.2.2. Each SME is integral to the project's success, contributing their specialized knowledge to different aspects of the application, from design and development to data analysis and security.

## 2.3. Developers

2.3.1. The developers involved in the Demand Response Portal carry significant responsibilities that are critical for the success of the project. They possess a diverse range of technical knowledge, which they apply in various aspects of the development process:

2.3.1.1. Lead Front–End Developer: Responsible for designing the user interface and enhancing user experience. Requires expertise in HTML, CSS, JavaScript, and libraries like Bokeh for data visualization, as well as an understanding of responsive web design principles.

2.3.1.2. Back-End Developers: Tasked with server-side logic, API integration, and database management. Technical knowledge includes proficiency in Flask for creating API endpoints, experience with MariaDB for database operations, and familiarity with database triggers for data integrity and automation.

2.3.1.3. Lead Back-End Developer / Database Administrator: Takes on the additional role of database administration, ensuring the performance, integrity, and security of the database. Proficient in database schema design, query optimization, and implementing database triggers in MariaDB.

2.3.1.4. Lead Data Scientist: Focuses on the data analytics aspect, using statistical models like ARIMA to process and analyze large datasets for predictive insights.

2.3.1.5. Lead Security Specialist: Oversees the security architecture of the project, implementing best practices for data protection, authentication, and authorization.

2.3.1.6. Scrum Master: Coordinates the development process using Agile and Scrum techniques, ensuring the team meets project milestones and deliverables efficiently.

2.3.1.7. Data Scientist: Supports the Lead Data Scientist in analyzing data and developing algorithms to enhance the predictive capabilities of the system.

2.3.2. Each developer is also expected to collaborate closely with team members, contribute to the Agile process management, and engage in knowledge transfer activities to ensure continuity and sustainability of the project.

# 3. Technical Knowledge

## 3.1. Core Languages

3.1.1. Python: The primary back-end coding language used for data manipulation and API creation within the project.

3.1.2. HTML and CSS: The primary front-end coding languages used for displaying the website's user interface.

3.1.3. SQL: The primary database coding language used for managing and querying the MariaDB database.

3.1.4. JavaScript: A programming language used for creating dynamic and interactive elements within the portal, especially callbacks in Bokeh.

## 3.2. Dependent Languages

3.2.1. Not applicable

### 3.3. Reporting Tools

3.3.1. Python libraries including Pandas and Bokeh for data visualization reports
3.3.2. Synk for reporting security analysis
3.3.3. Lighthouse for reporting the efficiency of the website

### 3.4. Databases

3.4.1. For the Demand Response Portal, we have established a robust and efficient data management system using the following databases and technologies:

3.4.1.1. MariaDB: This open-source database is the primary data storage solution for our project. Chosen for its compatibility with MySQL, MariaDB offers advanced features, performance improvements, and cost-effectiveness. It handles all our data requirements with ACID compliance, supporting various storage engines and ensuring high performance.

3.4.1.2. Database Triggers: We have implemented a series of database triggers within MariaDB to automate data management tasks, enforce data integrity, and manage associations and permissions. This includes triggers for creating user IDs, ensuring one user per meter, auto-assigning admin roles, and updating meter data deltas, among others.

3.4.2. These database components ensure the reliability and scalability of our system, facilitating the efficient and secure management of data throughout the Demand Response Portal.

### 3.5. Operating Systems

3.5.1. Servers: deployed with any operating system that can install Docker
3.5.2. Client-side: any modern machines with a web-browser

## 4. Business Knowledge

### 4.1. Business-related knowledge required for the knowledge transfer:

4.1.1. Energy Market Dynamics: Understanding the fluctuations in energy supply and demand, pricing strategies, and how they are represented and managed within the system.
4.1.2. Demand Response Strategies: Insights into how the system supports demand response initiatives, including load reduction during peak periods and incentivization for users.
4.1.3. Regulatory Compliance: Knowledge of relevant regulations and standards, such as those pertaining to energy conservation, user privacy, and data security, which the system must comply with.
4.1.4. Operational Processes: Familiarity with the operational procedures of power distributors and consumers, and how the portal facilitates these processes for efficiency and reliability.
4.1.5. Data Management: How the system manages and utilizes data, including data collection via smart meters, processing and storage in MariaDB, and real-time visualization with Bokeh.

4.1.6. Technology Integration: Understanding the integration of OpenADR and Typhoon HIL for communication and data exchange, ensuring seamless operation between providers and consumers.

4.1.7. Analytics and Forecasting: Comprehension of the ARIMA models used for predictive analytics and how these forecasts support business decisions and grid management.

4.1.8. User Engagement: Strategies employed by the portal to enhance user engagement and satisfaction through a responsive and intuitive interface, as designed with Figma, and executed with HTML, JavaScript, CSS, and Bokeh.

4.1.9. Security Measures: Awareness of the security protocols and measures in place, as implemented by the Lead Security Specialist, to protect the system and user data.

# 5. Application Knowledge

## 5.1. Application Level Function: primary functions and features of the application

5.1.1. The Demand Response Portal is a comprehensive solution designed to manage and visualize demand response and smart grid data. The primary functions and features of the application include:

5.1.1.1. Real-time Data Visualization: The portal offers real-time data visualization of energy consumption and grid performance, leveraging Bokeh for interactive and dynamic charting.

5.1.1.2. Demand Response Management: It allows users to participate in demand response events, effectively reducing load during peak periods and contributing to grid stability.

5.1.1.3. Incentive System: Users can view and earn incentives for their energy-saving behaviors, promoting active participation in energy conservation.

5.1.1.4. Load Shedding Capabilities: The portal facilitates load shedding by allowing users to adjust their consumption in response to grid demands.

5.1.1.5. Transparency and Insight: Both energy providers and consumers are provided with valuable insights into energy usage patterns, fostering transparency and enabling better energy management.

5.1.1.6. Advanced Analytics: Integrating ARIMA models, the portal offers advanced predictive analytics to forecast energy needs and optimize grid performance.

5.1.1.7. User Accessibility: Designed with a focus on UX/UI, the portal is accessible and easy to navigate, ensuring a seamless user experience for power customers and utilities.

5.1.1.8. Responsive Web Interface: The portal is responsive, meaning it's accessible on various devices and screen sizes, allowing users to interact with their data anytime, anywhere.

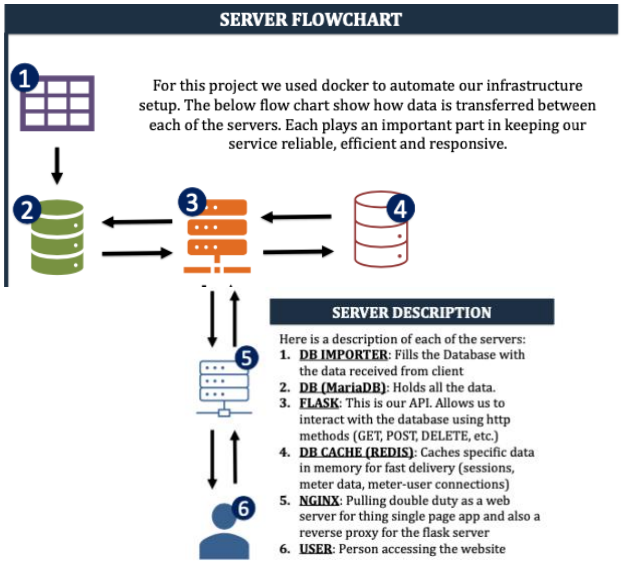
5.1.2. These features are supported by the underlying technologies such as Flask for the API, MariaDB for the database, Redis for caching, and Figma for UI/UX design mockups, ensuring that the application is robust, efficient, and user-friendly.

# 6. Application Environment

## 6.1. Flows/Diagrams

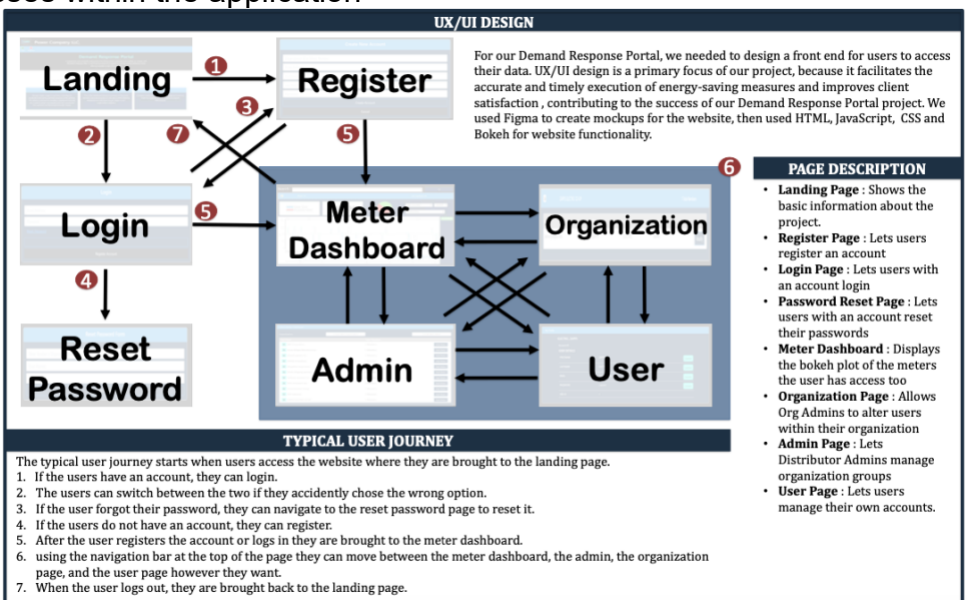
### 6.1.1. Flow of Data

6.1.1.1.




### 6.1.2. Processes within the application

6.1.2.1.




## 6.2. Server-Side Application Components

6.2.1. The server-side application components of the Demand Response Portal are meticulously designed to ensure seamless data flow, high performance, and robust security. The architecture is comprised of the following key components:

6.2.1.1. DB Importer: This component is responsible for populating the MariaDB database with data received from clients, ensuring that the data is stored and ready for processing and retrieval.

6.2.1.2. MariaDB Database: Chosen for its open-source flexibility and compatibility with MySQL, the MariaDB database stores all necessary

data for the application, including user information, transaction records, and meter data.

6.2.1.3. Flask API: The Flask framework is utilized to create a set of HTTP-based API endpoints (GET, POST, DELETE, etc.), allowing for secure and efficient interactions with the database from the client side.

6.2.1.4. Redis Cache: Employed as a data caching system, Redis significantly improves the performance of the application by storing sessions, meter data, and meter-user connections in memory for fast access.

6.2.1.5. NGINX: Serving a dual role, NGINX acts as a web server for the single-page application and as a reverse proxy for the Flask server, managing client requests and server responses efficiently.

6.2.1.6. Docker: The application's infrastructure is containerized using Docker, ensuring that the server environment is consistent, isolated, and reproducible across different development and production setups.

6.2.2. By integrating these technologies, the server-side of the Demand Response Portal is engineered to be efficient, scalable, and secure. The use of containerization with Docker also ensures that deployment is streamlined and consistent across any platform.

## 6.3. Client-Side Application Components

6.3.1. The client-side of the Demand Response Portal is engineered to provide an intuitive and responsive user experience, utilizing a combination of modern web technologies and design practices. The components and technologies used include:

6.3.1.1. HTML: Serves as the backbone of the web portal's structure, defining the content and layout of the web pages.

6.3.1.2. CSS: Provides the styling for the web portal, ensuring a consistent and engaging user interface across different devices and screen sizes.

6.3.1.3. JavaScript: Enables dynamic content and interactivity, allowing users to engage with the data visualization features and receive real-time updates without page reloads.

6.3.1.4. Bokeh: A Python library that is used for creating interactive charts and visualizations in the web portal, allowing users to visually analyze their energy consumption and savings.

6.3.1.5. Figma: Utilized for designing the UI/UX mockups, ensuring that the web portal is user-centric and meets the requirements for an efficient user journey.

6.3.1.6. Responsive Design: The web portal is designed to be responsive, meaning it adapts seamlessly to different screen sizes, from desktops to mobile devices, ensuring accessibility and ease of use.

6.3.2. By integrating these client-side technologies, the Demand Response Portal offers a rich and accessible user experience, focusing on real-time data interaction, and providing a platform that is both informative and easy to navigate.

## 6.4. Users-Network Environment

6.4.1. The network environment of the Demand Response Portal is designed to be robust and secure to support a variety of user types, including residential customers, commercial enterprises, and utility administrators. The environment is characterized by:

6.4.1.1. Distributed Access: Users can access the portal through a web-based interface, ensuring availability across different geographic locations.

6.4.1.2. Secure Communication: All data transmitted between the client and the server is encrypted using industry-standard protocols to protect against unauthorized access.

6.4.1.3. Scalable Infrastructure: The network infrastructure is capable of scaling to accommodate a growing number of users and an increasing volume of data traffic.

6.4.1.4. High Availability: The system architecture is designed for high availability with redundancy and failover mechanisms in place to minimize downtime.

6.4.1.5. Load Balancing: Network load balancers distribute traffic across multiple servers to ensure optimal performance and responsiveness.

6.4.1.6. Firewalls and Intrusion Detection Systems: Advanced firewall protections and intrusion detection systems are in place to monitor and protect against potential threats.

6.4.2. The user base is segmented into:

6.4.2.1. Residential Users: Typically access the portal to monitor and manage their energy consumption and participate in demand response events.

6.4.2.2. Commercial Users: Use the portal for more extensive energy management, tracking, and reporting functionalities tailored to business needs.

6.4.2.3. Utility Administrators: Have privileged access to oversee and manage the system, analyze energy consumption patterns, and handle customer data.

6.4.2.4. Each user type interfaces with the system via distinct access levels and permissions, ensuring they can only interact with the parts of the system relevant to their role.