# DD e-Mall

# 1 Introduction

## 1.1 Purpose

The main challenge of a massive deployment of electric mobility is the reduction of transportation's impact on climate by limiting the carbon footprint caused by our urban and sub-urban mobility everyday needs.

With this in mind, our aim is to develop and implement a new system called eMall having the purpose to expand the electric charging infrastructure with the means of realizing a world-wide network which fully connects all the actors involved in the charging processes. This is achieved by making all the back-end operations and transactions transparent to the everyday user which is fully supported in the charging process.

eMall will act as a platform deployed as a mobile application and implemented through several servers (eMSPs) located throughout the globe allowing to monitor electric mobility by communicating with various CPOs, each of them administrated through a CPMS.
Thereby the platform with expose several services such as:
- knowing where to charge the e-vehicle (locate charging stations owned and managed by CPOs)
- planning charging processes in a way to limit constraints on our daily schedule giving the possibility to the system to access our personal calendar
- choose from various charging possibilities based on special offers set by CPOs which can dynamically decide from where to acquire energy to

be distributed (from which DSO)

In order to guide the development of the system to be step by step, the document will focus on the description of the architectural design for the system's components alongside with their interaction. Finally, several mock-ups of the user interface are displayed followed by a guideline regarding testing and integration.

## 1.2 Scope

This document focuses on the architectural design choices made to guide the building of the system.

Interaction between the most important stakeholders as our scope is concerned is also taken into consideration, even though several ones might be left undefined.

In particular we consider the users, which might want to use the services provided by the eMSP sub-system, the CPOs (e.g. ENEL X, IONIX or others) physically deployed through charging stations and administrated by ad hoc pre-implemented CPMS sub-systems, and DSOs, external third party energy providers.
The system we are focusing on is the e-MSP, a remote sub-system (passive server) physically deployed worldwide, which receives requests from several users and uses uniform APIs to interact with other external sources such as CPMSs and other external systems (DMV, Payment Provider, etc.) in order to execute requests and return a valid reply to the requester.
On the other hand we also focus on the CPOs who may want to apply sepcial offers based on the performance of its charging stations or automatically or manually decide from which DSOs to acquire energy from.

Our main purpose is also to achieve the maximum geographic coverage possible with our system. To do so we would want to deploy several eMSPs servers throughout the globe.

As far as the architectural design, our main purpose is to adopt a three tier architectural style for the eMall system, where the Business logic is to be integrated to that of the pre-existent CPMSs' one. The client will have only the presentation layer; this choice is taken because having a thin client can give the possibility to make requests from any device (smartphone or EV's infotainment system) regardless of its hardware specifications. Data layer is then decoupled in order to guarantee data safety, facilitate data migration and last but not least for importance to ensure Data Warehousing along with ETL operations.

Further details are to be found in the next sections.

## 1.3 Definitions, Acronyms, Abbreviations

API - Application Programming Interface
COTS - Component Off The Shelf
CPMS - Charge Point Management System
CPO - Charging Point Operator
DMV - Department of Motor Vehicles (US equivalent of italian "Motor-izzazione Civile")
DSO - Distribution System Operator
e-Mall - Electric Mobility for All (Software Name)
e-MSP - e-Mobility Service Provider
EV - Electric Vehicle
MPI - Message Passing Interface
RASD - Requirements Analysis and Specification Document
DD - Design Document
TCP/IP - Transmission Control Protocol (over Internet Protocol)

VIN - Vehicle Identification Number
ETL - Extract Tranform Load
OS - Operating System
COTS - Component On The Shelf
LAN - Local Area Network
ORM - Object-Relational Mapping
DB - Database
DW - Data Warehouse
DBMS - Database management system

### 1.4 Revision History

### 1.5 Reference Documents

The specification document: "Assignment RDD AY 2022-2023_v3.pdf"

### 1.6 Document Structure

# 2 Architectural Design

### 2.1 Overview: High-Level Components and Interactions

The architecture of eMALL shall adopt a three-tier Ref pattern architecture in order to decouple the presentation layer, the business logic one and the data layer.
Thereby we can separate the business logic for each subsystem (eMSP and CPMS) and integrate them to create the whole eMall system.

We use firewalls at the entry of each business logic layer in order to associate to each subsystem its own data layer to provide data protection and prevent SQL injections and other malicious exploits.

Each tier can run on a separate OS and server platform, so the services of each tier can be customized and optimized without impact on the other tiers, granting in this way flexibility and scalability.

Some benefits of tree-tiers architecture:
- Since each tier can be developed simultaneously by different teams, an organization can bring the application to market faster, and programmers can use the latest and best languages and tools for each tier
- Any tier can be scaled independently of the others as needed.
- An outage in one tier is less likely to impact the availability or performance of the other tiers

- Presenation Tier
  The presentation tier handles the interaction with the users, it communicates with the Business Logic of the eMSP, showing the information retrived by this one and collect information from the user (the eMSP's Business Logic).
  This top-level tier can run on a web browser, as desktop application, or a graphical user interface (GUI). Therefore we have chose the GUI implementation in order to allow it to run both on the smart-phone and on the vehicle's infotainment system.

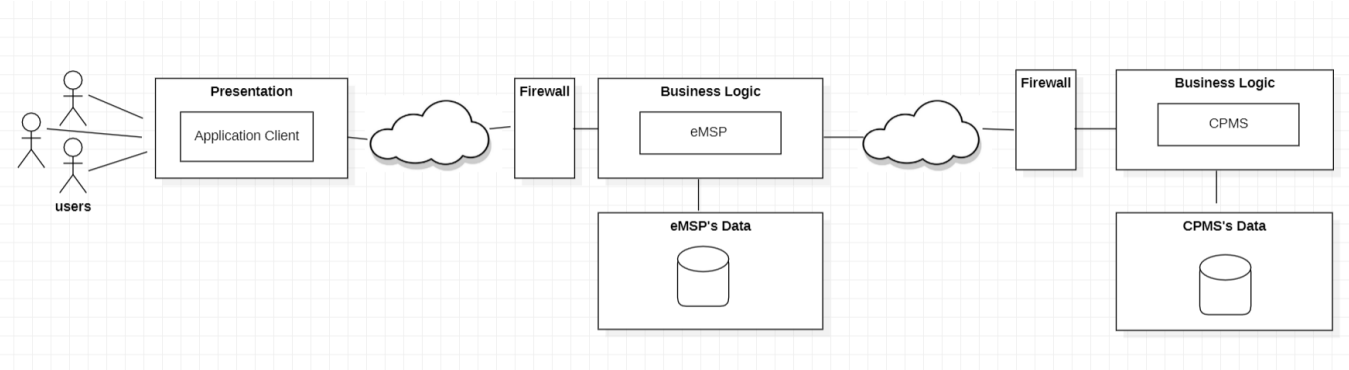
- Business Logic

  - eMSP's Business Logic:
    This tier contains all of the eMSP's Business Logic, containing for example the looking service procedure, the booking service one and the payment one along with other procedures.
    This component is responsible to pass the information to the

presentation tier and communicate with external services such as DMV, Digital Identity Provider and Payment Provider.

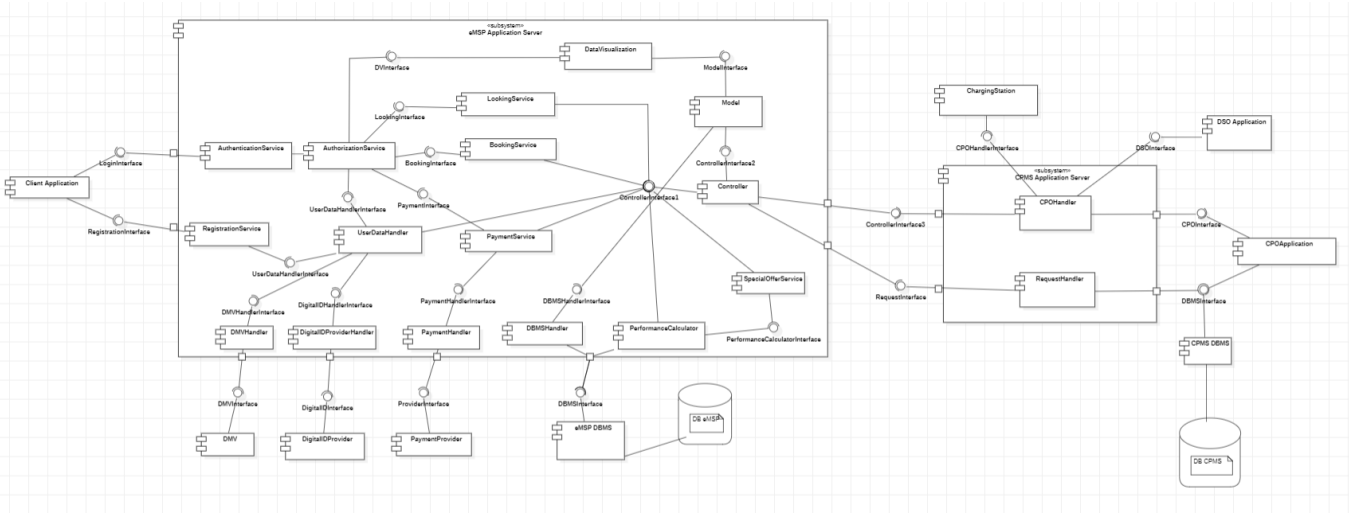
- CPMS's Business Logic
  This tier containes all the CPMS's Business Logic which is already implemented, deployed and up and running. For this case we assume the presence of two ad-hoc components that helps us integrate our eMSP subsytem: the Request Handler and the CPO handler.
  The first one is responsible to receive the requests sent from the eMSP Business Login and retrieve the information after having done a query to their own data base.
  The second one manages the operations that a CPO could perform such as the application of a special offer initiative or the decision to acquire energy from certain DSOs rather than others.

- Data
  The data tier, sometimes called database tier, data access tier or back-end, is where the information processed by the application is stored and managed.
  Both eMSP's Data and CPMS's Data handles the persistent data of the system
  In a three-tier application, all communication goes through the application tier. The presentation tier and the data tier cannot communicate directly with one another.




## 2.2 Component View

This section regards the component view of three tier eMall system presented within a component diagram. Each component showed in the diagram is then briefly described.



- Presentation
  - *ClientApplication*: It's an application that runs on the EV's infotainment system or on a smart-phone.
  - *DSOApplication*: It's the application used by DSO to charge their prices and energy availability.
  - *CPOApplication*: It's the application used by CPO to manage charging stations and dialogue with DSOs and eMSP's users.

- External Handlers:
  - *DMVHandler*: Component that manages the interaction with the DMV in order to perform license plate and VIN verification operations to associate the vehicle to the user.
  - *DigitalIDProviderHandler*: Component that manages the interaction with external public digital identity systems, such as SPID or CIE, in order to retrieve all user data more easily.
  - *PaymentProviderHandler*: Component that is responsible for communicating with third-party payment providers which verify user's payment details in order to commit the transactions.
  - *CarManufacturerHandler*: Component that provides vehicle status information when invoked by the eMSP subsystem. It periodically requests EV's status such as battery level.
  - *CalendarHandler*: Component that deals with external calendar system in order to match charging station reservation with user's activities.

- *ChargingStation*: Component physically deployed on a specific charging station which ensures the management and performance monitoring by responding to requests from the CPO-side system, via CPMS.

- Business Logic:
  - *AuthenticationService*: Service that takes care of user authentication by verifying the existence of an account in the database after a login has taken place. When it's used a digital identity system, the authentication and authorization processes happens together in the second one.

- *AuthorizationService*: Service that deals with user authorization process. It verifies the validity of privileges and information (e.g., an owned vehicle previously entered and verified through interaction with DMV) entered. Allows access to all software features related to the user.
- *RegistrationService*: Service for registering new users within the eMSP system. Dialogues with third-party providers for vehicle association (DMV), association with a public digital identity system (SPID or CIE), and association of payment methods (Payment) reached using their own handler.
- *DMV*: external system for DMV reached by eMSP sub-system through the handler, using REST API calls.
- *CarManufacturer*: external system that collects info about the vehicle status. It provides the info to eMSP when this last one performs requests.
- *Calendar*: external system for user calendar (it can be GoogleCalendar or Outlook) reached by eMSP sub-system through the handler in booking operations, through REST API calls.
- *DigitalIDProvider*: DMV: external system for digital identification reached by eMSP sub-system through the handler, using REST API calls.
- *Payment*: external system for payments reached by eMSP sub-system through the handler, using REST API calls.
- *DBMSHandler*: component that handles connection and sends queries to the DMBS.
- *UserDataHandler*: Component that handles user data in the moments after registration and authorization. It is used to create the session during which the user uses the application,

or to make changes to the profile, taking care of database updates.

- *LookingService*: Service that is responsible for conducting charging station search operations. It is requested by an authorized user, who submits his/her data (filters and/or position) to finalize the search.
- *BookingService*: Service that is responsible for carrying out booking operations for charging stations. It is requested by an authorized user and interacts with the availability of the charging station and user's calendar.
- *PaymentService*: Service that contacts the chosen provider for payments. It deals with requesting operations to verify the data entered for the payment, as well as managing the transaction and the outcome.
- *SpecialOfferService*: Service that deals with proposing special offers to the user based on the defined parameters and logic of CPOs and user characteristics.
- *PerformanceCalculator*: Component that is responsible for calculating charging station characteristics in relation to users' choices, in order to monitor the performance of charging stations and to propose offers and refine charging point access operations during search and/or booking.
- *Controller*: Component that responds to requests from the other sub-system by correctly routing between components the operations triggered by CPOs such as calculating the performance of the column in relation to system users, evaluation of special offers (SpecialOfferService and PerformanceCalculator), and more general requests such as displaying reports on application usage (DataVisualization). It is also responsible for addressing requests that comes from eMSP

sub-system itself by contacting model and the right components depending on the request. Finally, it's the component used to contact the CPMS's sub-system to obtain infos about charging station in lookup or booking operations.

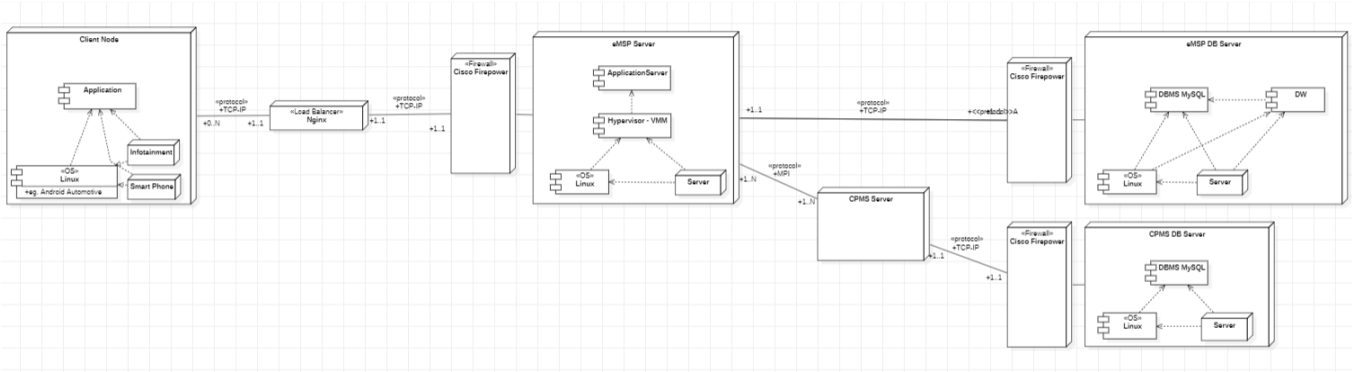
- *Model*: It's the main component which all other components refer to. It applies any kind of change to database: it contains an object-relation mapping (ORM) that other components use when retrieving data from the database. Therefore, it is the only component that deals with the data tier.
- *DataVisualization*: Component responsible for providing reporting on user activity and their relationship with charging stations. It also allows the creation of reports according to custom metrics. It can be used either by the authenticated and authorized user or invoked by the CPMS through the Controller.
- *CPOHandler*: Component that manages the operations of CPOs. It is responsible for configuring charging stations and interacts with DSOs to manage the process and decisions related to the supply and distribution of electricity to charging staizons. Requests to the eMSP subsystem for collecting data or sending decisions made to set up special offers depart from this component.
- *RequestHandler*: Component that manages requests sent by the eMSP subsystem (Controller) and so involves the component on CPMS subsystem properly.

- Data:

  - CPMS/eMSP DBMS: Component designed to enable efficient creation, manipulation and querying of databases.
  - CPMS/eMSP DB: Database of CPMS and eMSP subsystems.

# 2.3 Deployment View

The following section has the purpose to present a deployment diagram of the eMall system. The general environment and tools to be used in order to build the system from HW and SW points of view are presented along with the communication protocols used by the various nodes to exchange messages.




- Client Node: Client Node deployed on an EV's infotainment system or on a Smart-Phone running the eMall application on a Linux based OS.
  We have chosen Linux environment thanks to the advantages it brings and the compatibility with various devices. In fact, both Apple Car Play and Android Auto are based on Linux and so are Android and iOS OSs. Ref

- Load Balancer: A server which handles the requests coming from various clients and manages the workloud for a specific eMSP server deployed at a specific geographical location.
  We want to adopt a COTS so we decided to go for a Nginx solution due to its support and scalabaility's efficiency. Ref

- Firewall: A hardware defence point within the LAN's system which provides protection from sophisticated cyberattacks led by external agents trying to overcome system's vulnerabilities.
  Once again a COTS solution comes to our aid: Cisco Firepower for its over 99% threat blocking effectiveness seems to be a good choice. Ref

- eMSP Server: Server which hosts one part of the business logic of the eMall system. It runs on top of some physical machine. The main purpose is to take advantage of the benefits that virtualization provides:

  - Reduced upfront hardware and operating costs
  - Minimized or eliminated downtime
  - Increased responsiveness and scalability
  - Greater disaster recovery response.
  - Maximizing the usage of HW resources
    Two different approaches for virtualization can be used Ref :
  - Host Based approach: where VMs/Hypervisors run within an already installed OS on the machine's hardware
  - Bare Metal approach: where VMs/Hypervisors run directly on the machine's hardware without the support of an OS
    VM servers are ideal for dynamic workloads stressing apps that prioritize flexibility over consistently high performance.

- CPMS Server: Server which hosts the other part of the business logic of the eMall system. About what we stated above, the same reasoning can be donehere. But, as far as we know, the CPMS is already a deployed and up and running sub-system.
  We just need to integrate our eMSP subsystem with this last one in

order to provide services in the best way possible: this can be done by using adapter patterns and ensure compatibility with the CPMS Server exposed interfaces.

- eMSP DB Server: Server which hosts one part of the Data layer of the system.
  It consists of different componets:
  - a DBMS, a SW which ensures and manages the creation, the manipulation and the query efficiency of the database
  - a DB, physical machine storing data in a relational manner
  - a DW Ref which elaborates data with the means of ETL operations whose purpose is to analyze data and obtain information and metadata used then by Machine Learning techniques to predict future possible values for the data yet to be acquired

- CPMS DB Server: Server which hosts another part of the Data layer of the system. In particular this concerns the CPO's confiential information.

## 2.4 Runtime View

## 2.5 Component Interfaces

# 3 User Interface Design

# 4 Requirements Traceability

# 5 Implementation, Integration and Test Plan