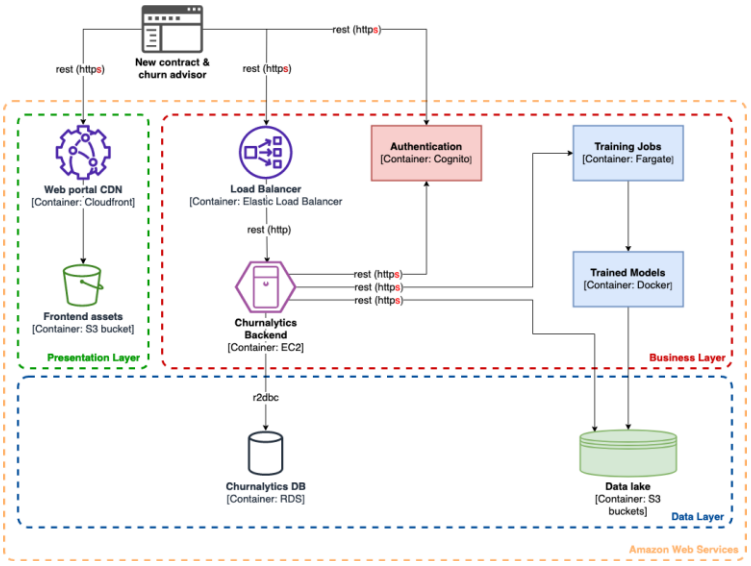
Technical Specification Double-side Page

1. **TECHNICAL SCOPE:** Summarize the solution developed during the EXPERIMENT phase: how have you finally addressed the challenge/Theme Challenges and tackled with its requirements and data. Include a diagram. Mention the different shared industrial and open data used.



The MVP is delivered in a classic 3-layer design deployed on an event-driven microservice architecture hosted in Amazon AWS. **Presentation Layer:** We built a web application to allow users to browse analytical data and forecasted results with several visual aids, and built-in graphs. The web application, developed with Vue, uses several libraries with graphs and dashboard capabilities and is connected to the backend by a REST API. It is deployed in AWS Cloudfront in order to speed up the distribution of the web content. **Business Layer:** The supporting backend server is being developed in SpringBoot and provides a REST API to the frontend. It is deployed in AWS Elastic Container Service. The ML models will be deployed on top of Docker images managed by AWS Fargate. The authentication is managed by AWS Cognito, which generates JWTs to be validated by the backend on each API call. **Data Layer:** The data fetched by the backend is stored in a PostgreSQL database and the datasets fetched by the ML models in a data lake hosted in AWS S3. Besides the datasets provided by EDP, we also have in store datasets about the market price for electricity.

2. **ALGORITHMS, TOOLS AND CONCLUSIONS:** Detail the algorithms and tools finally selected to accomplish the challenge/Theme Challenges. Summarize the main results that you have obtained during the EXPERIMENT phase: data, insights, conclusions and the main contributions to solve the challenge/Theme Challenges.

We employed various classification algorithms, including RandomForestClassifier, XGBClassifier, KNeighborsClassifier, and GradientBoostingClassifier, to predict the probability of churn in energy service contracts. It's important to note that while we are using classification models, our primary objective is to estimate the probability of the true class, which represents the probability of churn. In terms of tools and libraries, we utilised Python as our primary language, with pandas for data manipulation, numpy for numerical operations, scikit-learn and XGBoost for machine learning, and Optuna for Bayesian hyperparameter optimization. To facilitate cloud-based execution, we leveraged Docker for running our pipelines.

Our main dataset encompassed multiple datasets related to energy service contracts, including contract details, building information, and proposals. However, in this project phase, we focused exclusively on three datasets with contract data, resulting in approximately 130,000 rows after the join. Integrating data from these diverse sources presented challenges due to variations in data origins. Furthermore, we had plans to incorporate market price data for future analyses. Throughout our experiments, Bayesian optimization played a crucial role in fine-tuning hyperparameters, exploring feature selection, and testing oversampling techniques. Our objective function, utilising the F-beta metric with a beta greater than 2, consistently outperformed the traditional F1 score in enhancing true positive predictions.

Another critical objective was to determine the feature importance in churn contracts. To accomplish this, we initiated the process by calculating the correlation of each feature with the target variable (is_churn). Afterward, we conducted an in-depth analysis using the feature_importances_ function available in tree-based algorithms. This analysis is still an ongoing development that we are working on together with our data provider. Our results demonstrated a churn contract hit rate of approximately 62%, highlighting the effectiveness of our model in identifying potential contract churn.

3. **SCALABILITY AND FLEXIBILITY OF THE SOLUTION:** Explain how the solution copes with the challenge/Theme Challenges requirements and how can it be adapted to other similar problems. What work is still pending to create a real/stable product if any? What TRL level is it in? Is it a DVC?

The current architecture is prepared to scale both vertically and horizontally. RESTful services lend themselves to horizontal scaling since they are stateless and, therefore, easy to load balance. The containerization of the ML model training and optimization jobs using on-demand compute power allows us to continuously fine-tune the required CPU/GPU/RAM depending on how big the data is, how complex the problem becomes and how quickly do we need the optimised solutions. Our architecture can cope with bigger data and more complex optimization needs by tweaking the jobs on-demand compute power. Our core optimization platform may be applied to different business problems, including in other corporate verticals, by specifying new constraints and KPIs to optimise. Due to some delays out of our control, we had to reduce the scope of the project and focus only on the contrat churn vector. In our upcoming work, we plan to segment customers using clustering algorithms while continuing to utilise contract data. Additionally, our goal is to predict the probability of acquiring new customers using classification algorithms, leveraging proposal data for these predictions. This extension of our analysis will provide a more comprehensive understanding of our energy service contracts and customer behaviours.

From a technology readiness level perspective, we assess that our solution is at a level 6 for the churn probability vector. We managed to develop a DVC by analysing, processing and exploiting the data collected by our data provider, but also from other sources like energy market prices. As a result, our data provider has access to predictive information about contracts that might churn and act upon those in advance.

4. **DATA GOVERNANCE AND LEGAL COMPLIANCE:** Describe the security level of the solution, i.e. how authentication, authorization policies, encryption or other approaches are used to keep data secure. Explain how the solution is compliant with the current data legislations concerning security and privacy (e.g. GDPR). Describe in a convincing way how your solution realizes a secure DVC, e.g. through usage of specific tools.

AWS is compliant with several security policies, including ISO 27017, 27018, and 9001. Only 2 services are exposed to the Internet (i.e. CloudFront and Elastic Load Balancer) and have certificates and HTTPS. CloudFront only replies to HTTP GET/OPTIONS requests and HTTP traffic if redirected to the HTTPS port. The application load balancer, which is the system's only entry point, has several security policies in place to handle threats such as cross-site scripting and SQL injection. Authentication is managed by Cognito, which has several policies in place (e.g. password strength and rotation policies, 2-factor authentication, etc.), although some are still to be integrated in the system. Authenticated requests are checked against the user's authorised permissions which right now are performed based on user roles. The data layer is not exposed to the Internet and is only accessible to backend service and the containers with ML models. Datasets do not include Personally Identifiable Information (PII). We do not use session cookies or keep users' activity data for more than a few days (rolling file policies). User accounts may be deleted anytime, and their data redacted from the system while maintaining full auditing traces.

5. **QUALITY ASSURANCE AND RISK MANAGEMENT:** Describe the quality process followed for the final product. Technologically, which problems have you encountered and how you have solved them, and any processes followed that guarantee that the solution fulfils the challenge/Theme Challenges and data provider requirements.

Our Quality Assurance (QA) procedures include storing the code in versioned repositories with formatting standards, peer reviews, static analysis (with SonarQube) and a mandatory 80% coverage of automated tests for the backend to ensure only quality code is submitted. Additionally, we implement automated tests for Frontend and microservice and plug them to our CI/CD pipeline. Finally, we script manual test cases that are executed by our QA engineers. Identified defects are tracked using Linear boards. Security tests were performed by testing the top 10 OWASP API security risks. We continuously measure and track the performance of our models, which will also provide optimistic and pessimistic prediction intervals. QA of the classification solutions was achieved by assessing the model's performance on unseen data, evaluating its ability to make accurate predictions.

Risk 1: mismatch between requirements we captured, and the data provider's needs. Was mitigated with a joint endeavour with the data provider by defining user-stories, use-case diagrams and UI/UX mockups, and weekly meetings. **Risk 2:** no information about which optimization metrics are more relevant. Was mitigated with meetings with the data provider to better understand their optimization goals and which criteria is more relevant. **Risk 3:** the full datasets uncover patterns in the data that we have not considered. Was mitigated by repeating all the dataset analyses we performed on the samples during the Explore phase, on the full datasets provided in the Experiment phase. **Risk 4:** the quality of the real datasets may not allow some of the algorithms we have chosen to be applicable. Was mitigated by analysing the full dataset's quality and working together with the data provider to tweak and improve them. Open-data sources were also identified and prepared, including Portugal's electricity market data. **Risk 5:** there might be delays until the real datasets are provided. Was mitigated by reducing the scope of the project and focusing on the data provider's needs.