**Customers Churn Predictor**

**Project Report**



**Self Certificate**

This is to certify that the dissertation/project report entitled “Customers Churn Predictor ” has been carried out solely by me as an authentic piece of work for the partial fulfilment of the requirements for the award of the Diploma in Computer Science and Technology (CST) degree under the guidance of Raihan Mistry Sir. I hereby declare that I am fully aware of the guidelines stated in the “Diploma CST Project Report.”

A Project Report

Submitted By

Name : **Sandipan Sarkar**

Roll No:

**ACKNOWLEDGEMENT**

We feel immense pleasure to introduce **“Customer Churn Predictor”** as my solo project.

I would like to express my heartfelt gratitude to my mentor, **Mr. Raihan Mistry Sir**, who has been a constant source of knowledge and inspiration, and who provided invaluable guidance throughout the development of this project.

I am deeply grateful to my beloved parents for their unwavering support, insightful feedback, and encouragement, which have been instrumental in completing this project within the given timeframe.

I would also like to extend my thanks to my teachers and friends, whose motivation and support have given me the strength to persevere and move forward.

Finally, I extend my sincere gratitude to everyone who has provided moral support and encouragement, helping me bring this project to completion successfully.

**Index**

|  |  |  |
| --- | --- | --- |
| **SL No.** | **Content** | **Page No.** |
| 1 | Scope of the project | 5 |
| 2 | Abstract | 6 |
| 3 | Introduction | 7 |
| 4 | Literature Survey | 8 |
| 5 | Data Set & Importing Libraries | 9 |
| 6 | Data Analysis | 10 |
| 7 | Model Implementation | 14 |
| 8 | Web app & Interface | 16 |
| 9 | Deployment | 18 |
| 10 | Conclusion | 19 |
| 11 | References | 20 |

**1.Scope of the Project**

The project focuses on building a machine learning system to predict which customers are likely to leave (churn) a service or subscription. It analyzes historical data—like usage, complaints, and billing behavior—to flag high-risk users, allowing businesses to intervene early.

**Objectives:**

* Predict churn likelihood accurately.
* Identify top churn drivers.
* Segment customers by risk level.
* Automate churn scoring for batch/real-time use.

**Functional Scope:**

* Data collection from multiple sources (CRM, logs, billing).
* Data cleaning, feature engineering (e.g., tenure, usage drops).
* ML model training, tuning, and evaluation.
* Simple dashboard to track churn trends and drivers.
* Modular design for easy deployment (API-ready).

**Out of Scope (MVP):**

* Real-time streaming inputs.
* Retention recommendation engine.
* Full marketing/CRM automation.

**Constraints:**

* Must comply with data privacy laws.
* Favor interpretable models.
* Keep future scalability in mind.

**Expected Impact:**

* Reduce revenue loss from churn.
* Enable data-backed retention strategies.

**2.Abstract**

In an increasingly competitive business environment, customer retention has become just as vital as customer acquisition. This project focuses on building a web-based Customer Churn Prediction System that helps businesses identify customers at high risk of leaving. By integrating modern machine learning with lightweight, scalable web technologies, the system delivers both predictive insight and user-friendly interaction.

The application is built using a three-tier architecture. The frontend is developed with Streamlit, a Python-based framework designed for quickly creating interactive data applications. It provides a clean, responsive user interface that allows users to input customer information and view churn predictions along with visual analytics in real-time. The backend is powered by FastAPI, a modern, asynchronous web framework known for its high performance. It handles incoming requests from the frontend, manages model inference, and returns predictions efficiently via RESTful APIs. This separation of concerns ensures scalability, maintainability, and rapid response times.

At the heart of the system lies an Random Forest classifier, chosen for its high accuracy, speed, and robustness when dealing with structured data and class imbalance—common challenges in churn prediction. The model is trained on historical customer behaviour, transaction patterns, and demographic features to output the likelihood of customer churn. This probability score enables businesses to prioritize retention efforts effectively.

The system is designed with deployment and extensibility in mind, supporting containerization via Docker and allowing seamless integration with external CRMs or business intelligence tools. Additionally, the architecture accommodates real-time inference and can be extended to support batch processing or periodic model retraining. Overall, this project delivers a practical, end-to-end machine learning solution that empowers companies to make data-driven decisions and reduce churn proactively.

**3.Introduction**

In today’s data-driven economy, customer retention plays a pivotal role in ensuring long-term business success. Losing existing customers not only reduces revenue but also increases acquisition costs, making churn prediction a mission-critical task for organizations across sectors. Traditional methods for identifying potential churners often rely on manual analysis, static thresholds, or business heuristics—approaches that fail to scale or adapt in real time. With the evolution of machine learning and AI, businesses now have the ability to move beyond guesswork and adopt predictive models that provide actionable insights.

The Customer Churn Prediction project leverages this potential by using structured customer data—such as service usage patterns, account history, demographics, and support interactions—to forecast churn probability. The goal is to help businesses take proactive, data-backed decisions to retain customers more effectively. The project is built on a modern and modular tech stack: the machine learning core uses the Random Forest algorithm, known for its speed, accuracy, and ability to handle noisy or imbalanced datasets. The backend is implemented with FastAPI, enabling fast and scalable API endpoints for serving model predictions. The frontend is developed using Streamlit, offering a lightweight and intuitive interface that allows users to input data, view real-time predictions, and interact with performance visualizations without needing technical expertise.

By combining predictive analytics with production-ready components, this project offers a fully deployable solution for churn management. It enables real-time inference, supports future extensibility such as batch prediction or retraining pipelines, and integrates seamlessly with existing systems. Overall, this application serves as a powerful tool for businesses looking to reduce churn, improve customer lifetime value, and gain a competitive edge through smarter decision-making.

**4.Literature Survey**

Numerous research efforts have explored the domain of customer churn prediction, particularly in telecommunications, e-commerce, and subscription-based services, where customer retention is a primary concern. The methodologies employed in this project are informed by prior studies that apply classification algorithms to historical customer data to predict churn likelihood with high accuracy. These studies emphasize the importance of leveraging behavioural and transactional patterns to identify at-risk users effectively.

A critical component of the project is the dataset, sourced from **Kaggle**, a reputable platform offering diverse and high-quality public datasets. The dataset used contains a rich blend of customer attributes, including service usage statistics, tenure information, demographic details, and churn labels. This aligns with existing literature that highlights the role of feature-rich and well-curated datasets in training robust machine learning models. The presence of categorical and numerical variables within the dataset reinforces the need for comprehensive preprocessing, such as encoding, scaling, and handling class imbalance—steps thoroughly addressed in this project.

From a machine learning standpoint, past studies have demonstrated the superiority of **ensemble models**, especially **Random Forest**, in classification tasks involving imbalanced data and nonlinear feature interactions. Compared to traditional models like logistic regression or decision trees, Random Forest model consistently offers better performance and generalizability. This literature-backed insight directly influenced the model selection process for the churn predictor.

Further, research has shown that deploying machine learning models through scalable web technologies enhances their accessibility and practical value. Inspired by this, the project integrates the trained Random Forest model into a web stack using **FastAPI** for backend API services and **Streamlit** for interactive frontend deployment. This architecture follows the trend seen in recent studies advocating for real-time predictive systems that combine analytical power with user-friendly interfaces. The literature, thus, serves as a foundational blueprint that shapes the system design, from data handling to model deployment, ensuring both accuracy and usability in real-world scenarios.

**5.Dataset & Importing Libaries**

**Data Set**

The dataset used in this project (located in the Datasets/ folder as : WA\_Fn-UseC\_-Telco-Customer-Churn.csv) includes:

* Customer demographics: Gender, Senior citizen, Has partners, Has dependents.
* Account information: Tenure, payment method, contract, paperless billing.
* Services & billing info: Phone service, online backup, monthly charges & total charges etc.

**Importing Libraries**

The project uses several essential Python libraries:

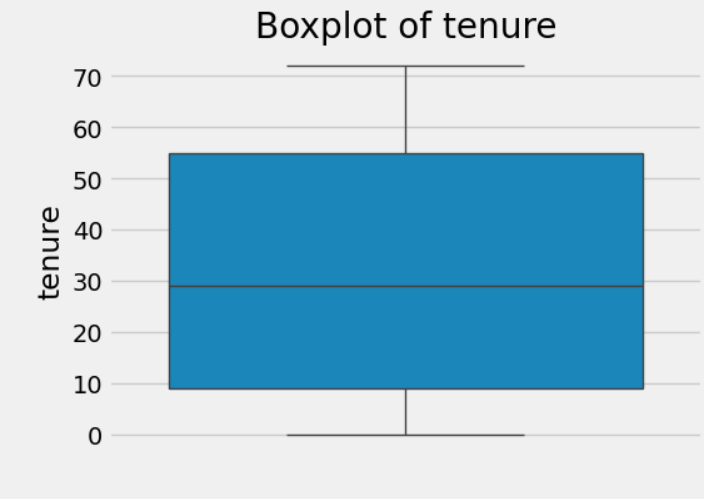
* NumPy: A fundamental package for numerical computations, offering powerful multidimensional array objects and routines for fast operations on arrays.
* Pandas: A versatile library for data manipulation and analysis that provides data structures (e.g., DataFrame) for handling structured data.
* Matplotlib: A comprehensive library for creating static, interactive, and animated visualizations in Python.
* Seaborn: Built on top of Matplotlib, it simplifies the creation of statistical graphics with attractive and informative visualizations.
* Scikit-learn: A robust machine learning library that offers efficient tools for data mining, data analysis, and implementing classification, regression, and clustering algorithms.
* Joblib: A utility for object serialization, particularly useful for saving and loading trained machine learning models.

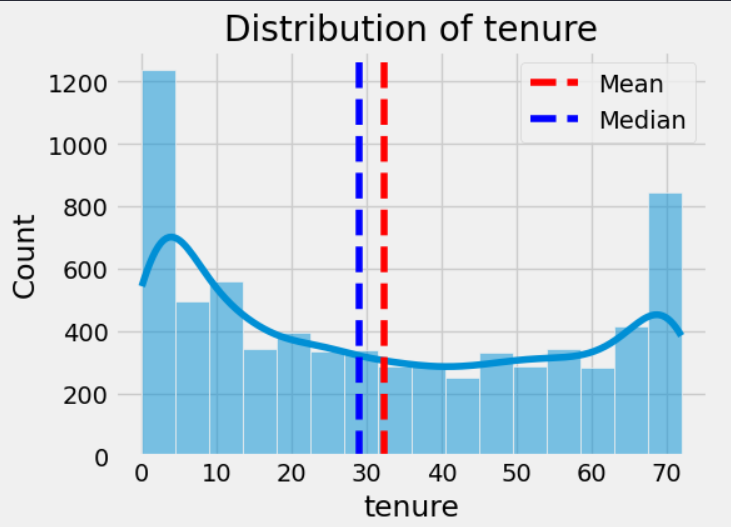
**6.Data Analysis**

Data visualization is a key step in understanding the dataset and selecting relevant features. The following plots were generated to assist in data exploration and to provide insights into customer behaviour.

6.1 **Distribution of Tenure & boxplot** :

This histogram shows the distribution of Tenure alongside Count. Also have a boxplot This **boxplot of tenure** gives a statistical summary of customer tenure (i.e., how long a customer has been with the company).

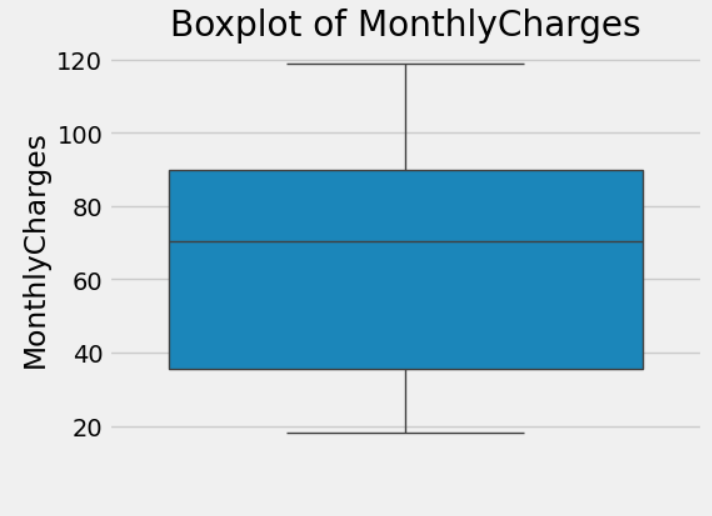


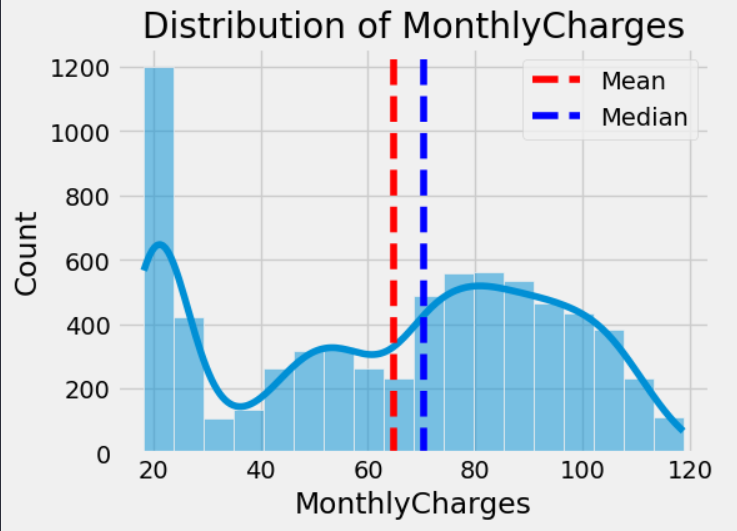


**Usage :** This plot demonstrates the Tenure distribution tailoring strategies & personalizing user expericnce

6.2 **Distribution of Monthly charges**

This is an another distribution plot of a important feature “Monthly Charges” which helps us to understand the distribution of the monthly charges

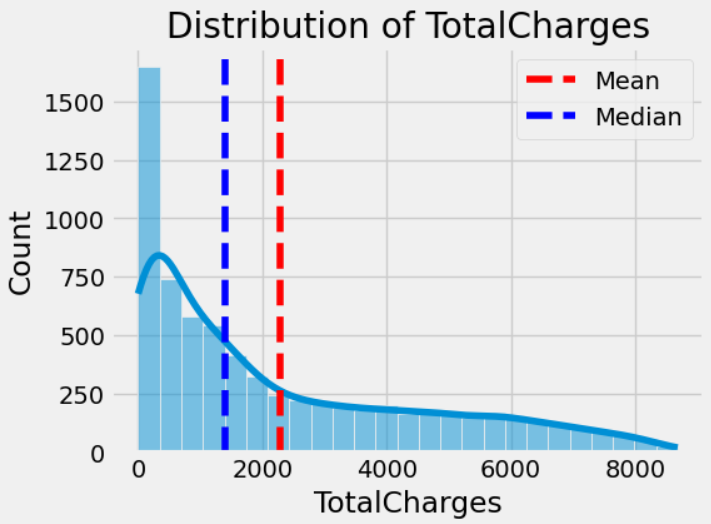


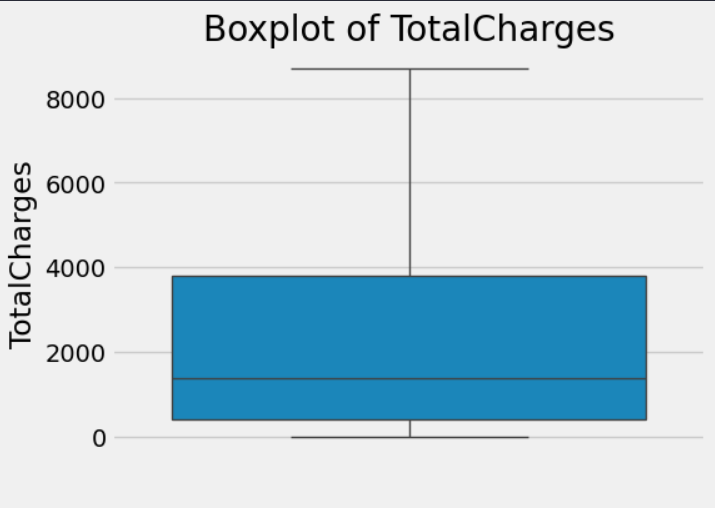


**Usage :** The distribution plot helps to understands how the Monthly charges deviates from its mean and median + it’s distribution where the boxplot provide us an important statistical summary about the spending of the users

6.3 **Distribution & boxplot of Total charges**

**This is the last distribution plot of an feature in the datset. This time it is about the “Total Charges” feature.**

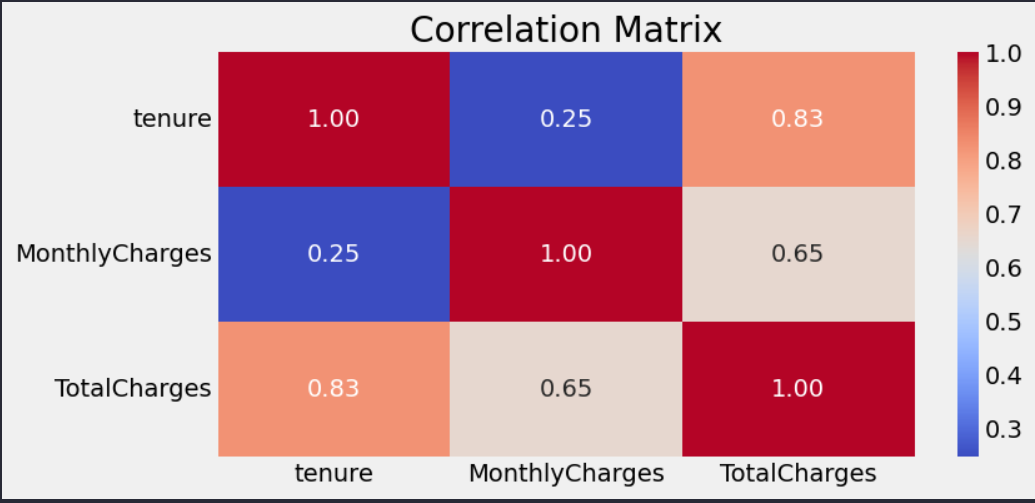




**Usage** : Same, It helps us to understand the distribution of the TotalCount feature also how it deviates from the mean & median as well where the boxplot provides a summary like how many people spends around 2000, how many spends more than 2000 but less than 4000 etc.

6.4 **Corelation matrix**

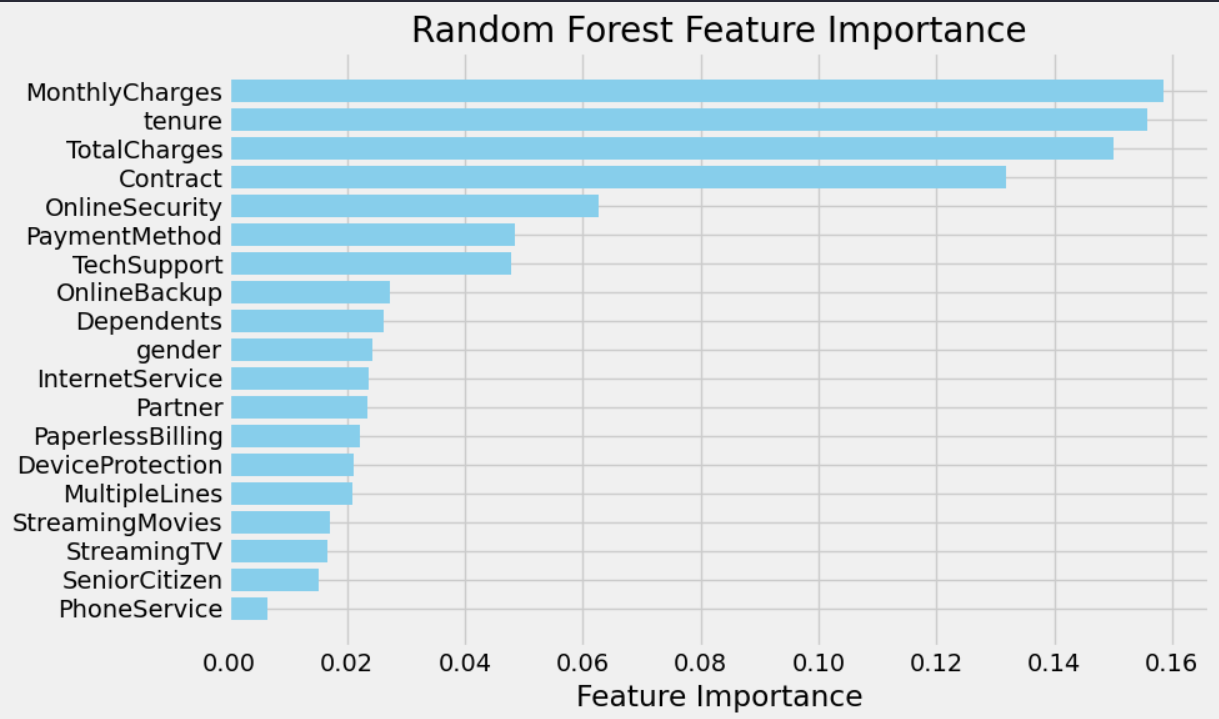
It is heatmap of how each feature of the datset is coorelated to other



**Usage** : Deeper the colour of the cell stronger the corelation of that feature with that another feature. It helps to identify what features have the highest influence in the dataset

6.5 **Feature Importance**

Showcases the Importance of each featues in the preproecessed dataset for model training



**Usage** : Helps us to understand which features are exactly needed for model training helps us to trim down the feature set increasing model accuracy while reducing the computational complexity

**7.Model Implementaion**

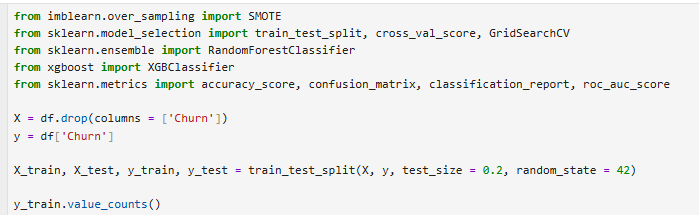
The machine learning pipeline is implemented using the **scikit-learn** library, with a focus on evaluating multiple ensemble classifiers. The workflow begins with **data preprocessing**, which includes cleaning missing or inconsistent values, encoding categorical variables using appropriate encoding strategies (like one-hot or label encoding), and scaling numerical features to ensure uniformity.

Following preprocessing, the dataset is split into training and test sets to facilitate fair evaluation of model performance. Both Random Forest and XGBoost classifiers were trained on the processed data. Hyperparameters for each model were tuned to optimize predictive accuracy and generalization.

After training, models were assessed using a set of standard classification metrics: accuracy, precision, recall, and F1-score. In addition, a confusion matrix was plotted to visualize the model’s performance across predicted classes. Based on comparative evaluation, the Random Forest classifier outperformed XGBoost in terms of balanced metric performance and interpretability, and was therefore selected as the final model for deployment.

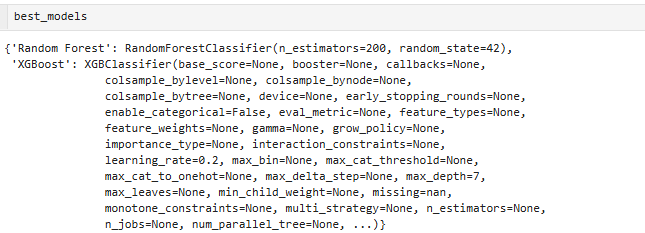
The selected Random Forest model was then serialized for reuse using pickle, along with the associated preprocessing pipeline, enabling real-time inference and smooth integration into the web-based prediction system.

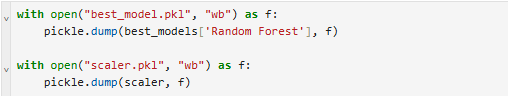
**Development code samples**











This snippet showcases how the model training is done & the model is choosen for the final predictions

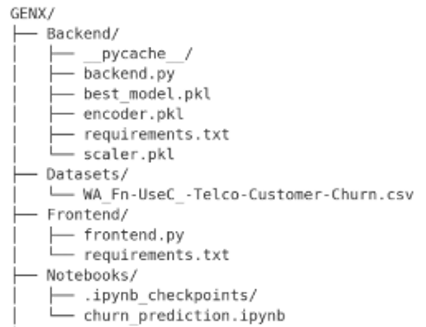
**8.Web Application & Interface**

This web application provides an interactive platform to predict customer churn using a trained Random Forest model, served via a FastAPI backend and visualized through a Streamlit frontend.

**Key Features:**

* **Input Interface**:  
  Organized into structured sections such as Customer Demographics, Account Information, Services, and Billing Information. Users can input both categorical and numerical data through dropdowns and sliders.
* **Real-Time Prediction Engine**:  
  The app connects to a deployed Random Forest classifier that returns churn predictions instantly via API.
* **Churn Probability Visualization**:  
  Displays the predicted churn risk percentage using a radial gauge with color-coded segments for risk interpretation. Results are labeled clearly (e.g., "Prediction Complete: No Churn").
* **Customer Summary Panel**:  
  Presents a concise summary of the input values and prediction outcome for quick review.
* **Key Risk Indicators**:  
  Highlights influential factors contributing to churn risk, such as contract type, payment method, and billing preferences.
* **Backend Integration**:  
  The FastAPI backend handles model inference and system health checks. API connection status is displayed in real time.

Here is the Project directory structure :

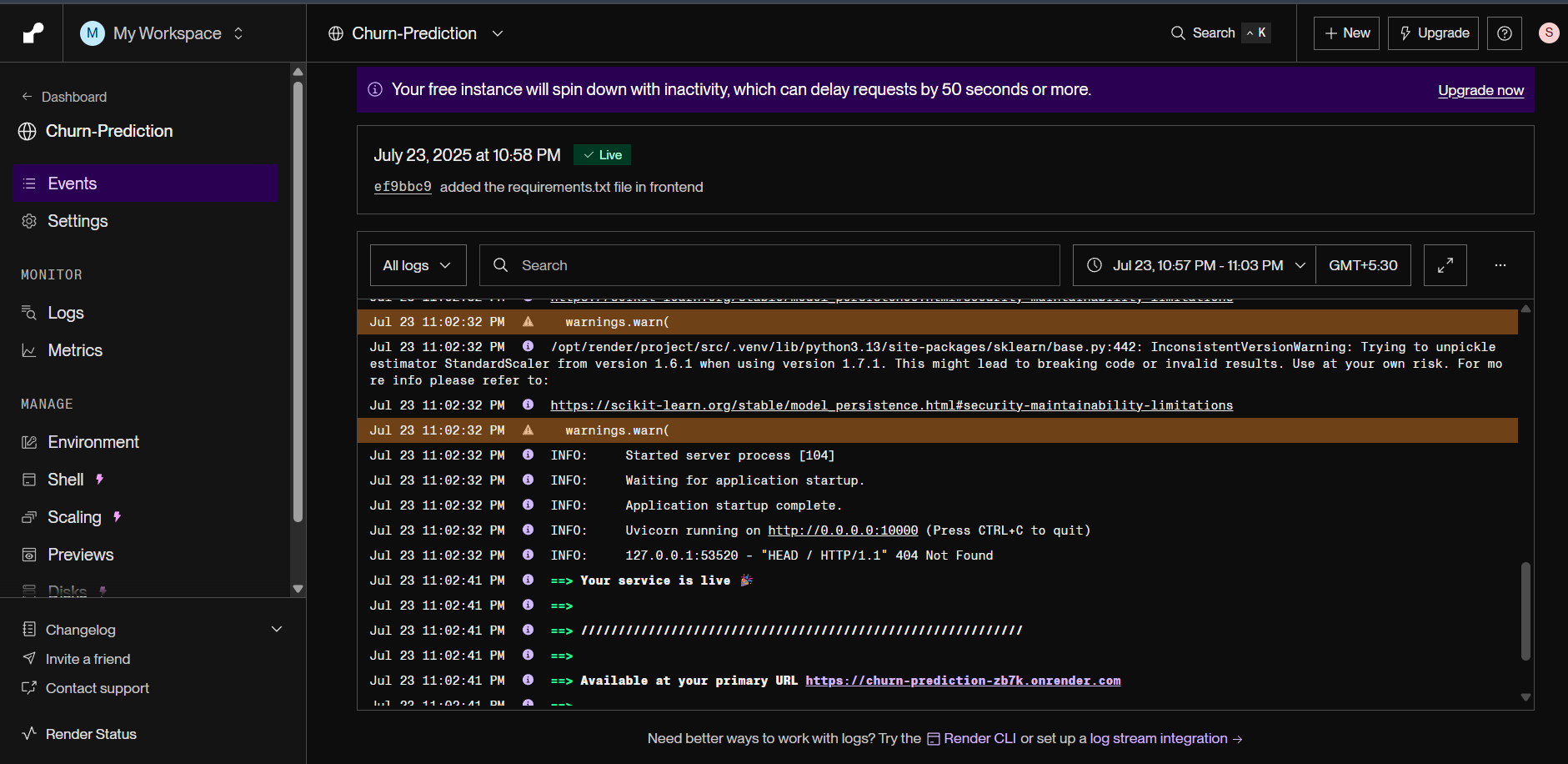


It showcases how the whole project is organized. The “Backend” folder is for the backend sever files & the “Frontend” folder is for the files that manages the frontend part of the webapp. “Datasets” folde contains the datsets used for model training where the “Notebooks” folder is where the jupyter notebooks that I used for preprocessing, model training & validation lies.

**9.Deployment**

Deploying the **Customer Churn Predictor** involves setting up the environment, running migrations, collecting static files, and ensuring the model is properly referenced in the FastAPI app.

**Screenshot of Backend Deployment Process :**



**Screenshot of Fronend Deployment Process :**as can be seen that the frontend part of the webapp is successfully deployed on the streamlit cloud platform

****

**10.Conclusion:**

The Customer Churn Prediction project demonstrates a complete end-to-end machine learning pipeline, from data preprocessing to model deployment. It leverages supervised learning techniques, with both Random Forest and Random Forest models being trained and evaluated. After careful comparison, the Random Forest model was selected based on its superior performance across key evaluation metrics. The project emphasizes practical implementation by deploying the model as a fully functional web application using FastAPI and Streamlit, enabling users to make real-time predictions based on customer input data.

The application is designed with user experience and business utility in mind. It provides detailed churn probability, risk categorization, and highlights key contributing factors to customer attrition. This makes it an effective tool for customer retention strategy development. With clean modular separation between frontend and backend, the system is also easily scalable and maintainable.

In summary, this project not only applies machine learning to a real-world problem but also showcases effective model selection, API integration, and UI development—providing a robust solution for businesses to anticipate and mitigate customer churn.

**11.References**

* GitHub Repo : https://github.com/Annoymous-ss/Churn-Prediction/tree/main
* Streamlit Documentation : https://docs.streamlit.io/
* FastAPI documentation: https://fastapi.tiangolo.com/
* Sckit-Learn Documentaion: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html