Capstone Project Proposal Report
(Individual Report)

Instructions:

This form is to be completed by each student doing Project registration to fulfill their senior design or capstone requirement. It must be completed and submitted to your Guide. Each student must complete this form individually.

This report is to be completed during the starting of the semester, while the project description report will be completed during end of the semester.

Guide Approval (initials/date):		

CAP4001– Capstone Project Proposal Report

Student Name	**Chintan Kamleshbhai Lad**
Student Register Number	**22BCB7071**
Programme	Bachelor of Technology
Semester/Year	Fall Semester 2025 - 26
Guide(s)	
Project Title	Advanced Load Forecasting for Greenfield Smart Cities Using Hybrid Machine Learning Models

Team Composition: Provide the information below for each member of the **project team**. Include **all** project team members, not just those in your discipline or those enrolled for Capstone project. Please also include yourself!

Reg. No	Name	Major	Specialization
22BCB7051	Swapnaneel Sarkar	Computer Science	Business Systems
22BCB7071	Chintan Kamleshbhai Lad	Computer Science	Business Systems
22BCB7101	Nikhil Jangid	Computer Science	Business Systems
22BCB7115	Naman Nigam	Computer Science	Business Systems

Project and Task Description: Provide a brief (one or two page) technical description of the design project and your specific tasks, as outlined below: (use a separate sheet)

(a) Provide a summary of the project, including a description of the project and its requirements, the purpose, specifications, and a summary of the approach. If this is a continuing project, you may use and/or edit the same project description.

This capstone project advances electricity load forecasting for greenfield cities, addressing the inherent data sparsity by extending a baseline conceptual framework that employed synthetic datasets derived from Gujarat's historical patterns via the Central Electricity Authority (CEA). The baseline models—Linear Regression (R^2=0.942, RMSE=45.29 kW, MAPE=5.78%), XGBoost (RMSE=53.22 kW, MAPE=5.83%), and Random Forest—served as proofs-of-concept for time-series prediction using multi-variate inputs like demographic scaling and smart city efficiency factors.

This iteration introduces a hybrid architecture integrating recurrent neural networks (RNNs) such as Long Short-Term Memory (LSTM) or Gated Recurrent Units (GRU) with ensemble boosting techniques to model non-linear temporal dependencies and stochastic variations in load profiles. Targeting enhanced metrics (e.g., MAPE<4%, RMSE reduction >10% from baseline), the system ensures computational feasibility through distributed training on platforms like Google Colab, while adhering to ethical data handling protocols for simulated smart meter streams.

The purpose is to facilitate predictive analytics for energy infrastructure in nascent urban zones, exemplified by GIFT City, where phased development introduces volatile demand curves. By optimizing grid allocation and incorporating renewables (e.g., solar irradiance-driven adjustments), the model mitigates over-provisioning risks, promoting energy efficiency and sustainability.

Requirements encompass seamless integration of heterogeneous data streams, robustness to noisy inputs via regularization techniques (e.g., L2 penalties in neural networks), and validation against real-world benchmarks from CEA APIs, all while maintaining scalability for deployment in IoT-enabled smart grids.

Specifications

- Input Data: Time-series from CEA API (hourly Gujarat consumption); simulated GIFT City meters (sectoral loads with Gaussian noise injection); meteorological APIs (e.g., OpenWeatherMap for multivariate features like temperature-humidity cross-terms).

- Output: Probabilistic forecasts with confidence intervals, evaluated via RMSE, MAPE, and MAE; accompanied by residual plots for bias detection.

- Performance: Surpass baseline through hyperparameter tuning (e.g., grid search for LSTM hidden units) and cross-validation (k=5 folds).

- Constraints: Open-source stack (TensorFlow/Keras for RNNs, scikit-learn for ensembles); data simulation via Monte Carlo methods; training epochs limited to <100 for efficiency.

Summary of the Approach

The methodology refines baseline pipelines by emphasizing feature extraction and model fusion: preprocess via imputation (e.g., Kalman filtering for missing values) and normalization (z-score), engineer domain-specific covariates (e.g., cooling degree days as HDD = max(0, T_base - T_avg)), then train hybrid ensembles with weighted averaging for prediction fusion

(b) Describe the specific role and tasks that **you individually** will be completing as part of the design of the project. What **specific deliverables** will you produce?

My individual role involves laying the foundational data infrastructure to support the hybrid load forecasting model. This includes conducting targeted research on data sources, collecting and integrating multi-modal datasets, and performing advanced preprocessing and feature extraction to ensure high-quality inputs for downstream modeling. My contributions are critical in the design phase, bridging raw data challenges (e.g., scarcity in greenfield contexts) with actionable features that enhance model accuracy and robustness.

Specific tasks I will complete:

- **Research and Data Sourcing:** Review literature and identify optimal datasets, such as pulling real-time consumption data from the CEA API, simulating GIFT City smart meter readings using Monte Carlo methods, and integrating meteorological variables from OpenWeatherMap API via scripted queries.

- **Data Preprocessing:** Clean and transform datasets by handling outliers (e.g., via IQR method), imputing missing values with techniques like forward-fill for time series, and normalizing features (e.g., min-max scaling for weather covariates) using Python libraries like Pandas and Scikit-learn.

- **Feature Engineering:** Design and implement domain-specific features, including temporal lags, interaction terms (e.g., temperature-humidity products for cooling load estimation), and derived metrics like heating/cooling degree days, followed by selection via correlation analysis and dimensionality reduction (e.g., PCA).

- **Pipeline Development:** Build a modular ETL (Extract, Transform, Load) pipeline in Python, testable on Google Colab, to automate data flow and ensure compatibility with LSTM/GRU and ensemble models.

- **Validation and Documentation:** Conduct exploratory data analysis (e.g., generating correlation heatmaps) to validate feature efficacy and document processes for team handoff.

Specific deliverables I will produce include:

- A comprehensive dataset repository (e.g., unified CSV/Parquet files of preprocessed data) hosted on GitHub.

- Feature engineering report with code notebooks, including visualizations like feature importance plots and correlation matrices.

- Automated data pipeline scripts with unit tests.

- Interim validation metrics (e.g., data quality summaries) to support model iteration.

(c) Discuss in detail the specific approach that will be used to complete **your** portion of the design.

My approach to completing this portion of the design is systematic and iterative, emphasizing data quality and model compatibility while addressing greenfield forecasting challenges like sparsity and variability. This ensures the creation of a scalable data foundation that directly supports the hybrid model's accuracy goals, like achieving MAPE below 4%.

- Conduct literature review on datasets and script API integrations for CEA and OpenWeatherMap, while developing Monte Carlo simulations for GIFT City data to source a comprehensive repository.
- Clean datasets by removing outliers via IQR, imputing missing values with time-series methods like interpolation, and normalizing features using z-score for consistency.
- Merge multi-source data into a unified DataFrame, performing diagnostics such as correlation matrices to detect and mitigate multicollinearity.
- Engineer temporal and domain-specific features, including lags, rolling averages, cooling degree days, and socio-economic indices with polynomial interactions.
- Select high-impact features using Recursive Feature Elimination (RFE) and Variance Inflation Factor (VIF) checks to ensure robustness without redundancy.
- Build a modular ETL pipeline with unit tests and visualizations (e.g., feature importance plots), documenting in Jupyter notebooks for team handover and iterative refinements.

(d) Describe the phases of the design process that will be incorporated and what work will be accomplished during those phases. (you may attach a Gantt Chart)

The design process for this capstone project on enhanced electricity load forecasting for greenfield cities follows a structured, iterative methodology inspired by standard data science and engineering workflows (e.g., CRISP-DM). It incorporates six key phases to ensure systematic progression from conceptualization to deployment, building on the baseline conceptual report. Each phase includes specific accomplishments, with milestones for validation and collaboration.

My focus will be prominent in the early phases, providing preprocessed data for later stages. The process is designed to span 4-6 months, allowing for iterations based on feedback.

- **Phase 1: Planning and Research (Weeks 1-2):**
  Define project scope, review baseline models (e.g., Linear Regression, XGBoost from the conceptual report), and conduct literature surveys on datasets and techniques. Accomplishments include a detailed project plan, requirements document, and identification of data sources like CEA API and GIFT City plans.

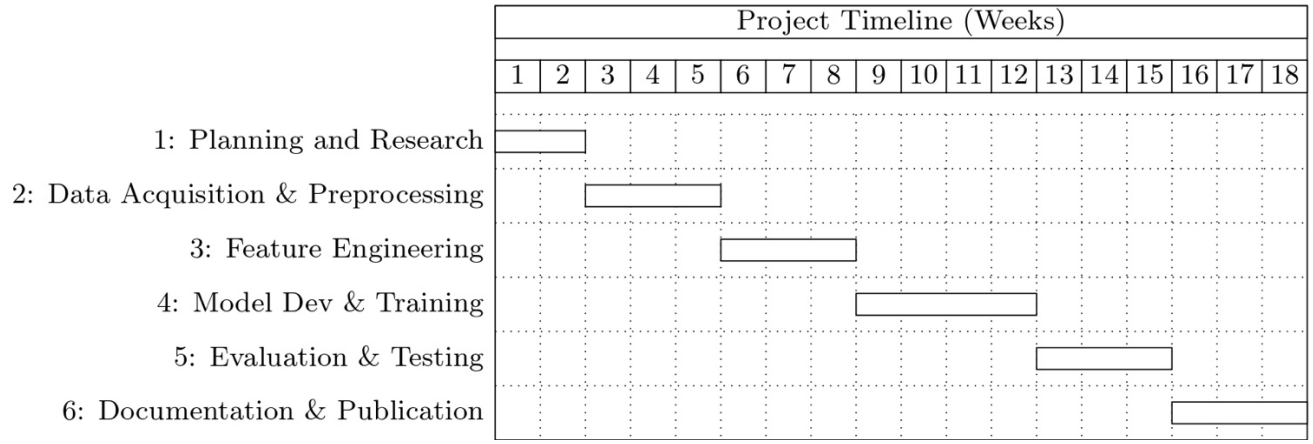
- **Phase 2: Data Acquisition and Preprocessing (Weeks 3-5):**
  Collect real and simulated data via APIs and Monte Carlo methods, followed by cleaning (e.g., outlier removal, imputation). Accomplishments: A unified, normalized dataset repository ready for analysis, with diagnostics like correlation matrices.

- **Phase 3: Feature Engineering (Weeks 6-8):**
  Design and implement advanced features (e.g., cooling degree days, temporal lags, socio-economic indices). Accomplishments: Engineered feature set with selection via RFE and VIF, documented in notebooks for model compatibility.

- **Phase 4: Model Development and Training (Weeks 9-12):**
  Build hybrid LSTM/GRU-ensemble models, tune hyperparameters, and train on prepared data. Accomplishments: Functional hybrid model with initial performance metrics (e.g., RMSE, MAPE).

- **Phase 5: Evaluation and Testing (Weeks 13-15):**
  Validate through cross-validation, compare against baseline, and test on simulated scenarios. Accomplishments: Optimized model with metrics exceeding targets (e.g., MAPE <4%), plus visualizations like residual plots.

- **Phase 6: Documentation and Publication (Weeks 16-18):**
  Compile results, prepare reports, and draft a research paper. Accomplishments: Final deliverables including code repository, technical report, and publication-ready manuscript.



Outcome Matrix: Describe your plan to demonstrate each of the outcomes below.

Outcomes:	**Plan for demonstrating outcome:**
a) an ability to apply knowledge of mathematics, science, and engineering	Apply mathematical and statistical techniques such as z-score normalization, correlation analysis, and polynomial feature transformations during data preprocessing and feature engineering; demonstrate through documented examples in the technical report, including calculations for metrics like cooling degree days derived from weather data.
b) an ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability	Design a scalable data pipeline that integrates sustainable features (e.g., renewable energy indicators like solar radiation) while addressing economic constraints via free tools and ethical data simulation; demonstrate with a flowchart, prototype scripts, and a report showing how it optimizes forecasts for eco-friendly grid planning in greenfield cities.
c) an ability to function on multidisciplinary teams	Collaborate with team members in areas like machine learning and urban planning by sharing preprocessed datasets and features; demonstrate through meeting minutes, shared GitHub repositories, and logs of contributions to hybrid model integration.
d) an ability to identify, formulate, and solve engineering problems	Identify challenges like data scarcity in greenfield forecasting, formulate solutions through simulated datasets and advanced features, and solve via engineering techniques like Monte Carlo simulations; demonstrate in the final report with problem statements, methodologies, and comparative metrics (e.g., before/after RMSE improvements).
e) an ability to communicate effectively	Develop clear documentation, including Jupyter notebooks with visualizations and a publication draft explaining data processes; demonstrate through presentations, team updates, and structured reports that convey technical details accessibly.
f) an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice	Employ tools such as Python (Pandas, NumPy, Scikit-learn), APIs (CEA, OpenWeatherMap), and Google Colab for data sourcing, preprocessing, and pipeline building; demonstrate with code snippets, ETL scripts, and demos in the project repository showing practical application.

Realistic Constraints:

Discuss on the Realistic Constraints taken in to account for the Project:

- Scope Constraints: We're keeping the project focused on GIFT City as our main case study and sticking to hybrid models like LSTM/GRU with ensembles, without diving into every possible urban scenario or extra features that could blow up the complexity. To handle this, we've designed everything modularly—so if we want to expand later, it's easy—but for now, it ensures we deliver a solid proof-of-concept without overreaching.

- Data Availability Constraints: Getting real-time data from places like GIFT City smart meters isn't always straightforward due to access restrictions, which could lead to inaccuracies if we're not careful. We're mitigating this by using robust simulations with Monte Carlo methods, cross-checked against reliable baselines from the CEA API, to make sure our data is as close to real as possible without stalling the project.

- Computational Resource Constraints: We don't have access to fancy supercomputers, so training deep learning models could be slow or limited. To work around this, we're relying on free tools like Google Colab with GPU options, optimizing our code to reduce training times (like limiting epochs), and testing on smaller datasets first before scaling up.

- Technical Expertise Constraints: Being students, we might not be experts in every area, like advanced neural network tuning or urban data specifics. We're addressing this by drawing from open-source tutorials, the baseline GitHub code, and splitting tasks across the team—I'm handling data engineering, which plays to my strengths, while others tackle modeling.

- Integration and Dependency Constraints: The whole hybrid system relies on smooth connections between data pipelines and models, and mismatches could cause bugs or delays. We're managing this with standardized formats in our ETL scripts, early integration tests, and regular team check-ins to catch issues before they become big problems.

Engineering Standards:

Discuss the Engineering Standards that will be followed and maintained in the Project:

- IEEE 12207 and 1233 (Software and Systems Engineering): Structure workflow with requirements, design reviews, and verification for our data pipeline and hybrid model, ensuring specs like MAPE <4% are met systematically.
- ISO/IEC 27001 (Information Security): Implement access controls, data anonymization, and risk assessments for sources like CEA API to protect privacy and comply with regulations.
- IEC 61850 (Power Systems Communication): Design API interfaces compatible with smart grid protocols for potential real-world integration in places like GIFT City.
- IEEE Ethically Aligned Design and GDPR Principles: Conduct bias checks in feature engineering and ensure transparent, fair AI practices in our LSTM/GRU models.
- ISO 9001 (Quality Assurance): Use code reviews, unit testing (e.g., pytest), and documentation in notebooks for consistency and continuous improvement.