Software Requirements Specifications

UNIVERSITY OF SARGODHA

DEPARTMENT OF COMPUTER SCIENCE

FACULTY OF COMPUTING & IT

Bachelor's Degree in Computer Science

Area: AI & ML

Project Code: HiGWM-22

Physics Informed LLM Based Weather Prediction System

FYP Coordinator:

Dr. Muhammad Ilyas

Supervisor:

Dr. Fahad Maqbool

Supervisor's Signature:

____

Second Supervisor:
Miss Anum Saleem

Candidates:

(TL) M Abubakar (BSCS51F22S026)

M Awais Ilyas (BSCS51F22S018)

M Zeeshan Asif (BSCS51F22S017)

BSCS SELF SUPPORT

2022/2026

Table of Contents

Document Information

Category	Information
Customer	CS & AI, UOS
Project	HI-GWM22
Document	Requirement Specifications
Document Version	1.0
Identifier	PGBH01-2003-RS
Status	Draft
Author(s)	Muhammad Abubakar, Muhammad Awais Ilyas, Muhammad Zeeshan
Approver(s)	PM
Issue Date	OCT. 18, 2005
Document Location	
Distribution	1. Advisor 2. PM 3. Project Office

Definition of Terms, Acronyms and Abbreviations

This section provides the definitions of all terms, acronyms, and abbreviations required to interpret the terms used in the document properly.

Term	Description
NWP	Numerical Weather Prediction
PINNS	Physics Informed Neural Network System
LLMs	Large Language Model
ML	Machine Learning

1. Introduction

1.1. Purpose of Document

The Purpose of this Software Requirement Specification (SRS) document is to set the functional and non functional requirements for Physics Informed LLM Based Weather prediction System. This document serves as a guideline for the development, outlining functional and non functional requirements, constraints and interfaces. The intendent audience for example project advisors, the project manager, the development team members, and the university evaluators who will evaluate the system's planed architecture and feasibility. This certifies all stakeholders have a clear idea of what the system will deliver, guiding to align expectations and implementation.

1.2. Project Overview

The main concern of this project is to create a hybrid weather prediction system that integrates Physics-Informed Neural Network (PINNS) with Machine Learning (ML) techniques and Large Language Models (LLMs). Traditional Numerical Weather Prediction (NWP) models are accurate and precise but require major computational power and time which may often lead to delays in real-time forecasting. On the other-hand, pure data driven ML models can deliver faster results but may generate outputs that breach physical laws for an instant unrealistic temperature spikes or wind pattern. In contrast, our system will aim to address these issues by integrating physical equations (e.g., Navier-Stokes for fluid dynamics) directly into the neural network training process through PINNs, allowing predictions are both data informed and physically consistent. LLMs will then process these outputs to produce natural language summaries, allowing the forecasts accessible to non experts. The main concern is to improve accuracy for short term predictions, less computational cost a compared to supercomputer based NWP and enhanced interpretability which can help in applications such as agriculture, disaster management, and policy planning.

1.3. Scope

The system will handle short-term weather forecasting (1-3 days ahead) for key variables like temperature, humidity, pressure, and wind speed, using publicly available datasets. It includes data preprocessing, PINN-based prediction with physics constraints, LLM integration for human-readable outputs, and basic evaluation against ML baselines. The system will be a prototype focused on demonstration, not production deployment. What the system will do: Process weather data, generate physics-consistent forecasts, and provide interpretable summaries. What the system will not do: Perform long-term climate modeling, high-resolution global simulations, real-time data ingestion from sensors, or serve as a public-facing operational tool. Exclusions also include training custom LLMs from scratch or using proprietary datasets.

The system will generate short term weather forecasting (1-3 days ahead) for key variables like temperature, pressure, humidity and wind speed using publicly available datasets.

2. Overall System Description

This section describes the general context of the system, including its users, environment, and limitations.It will cover data preprocessing, PINN based prediction with physical constraints, LLM embedding for human readable outputs and basic evaluation against ML baselines. The system will be a prototype that will focus on demonstration, not production deployment.

What the system will do:

- Process weather data
- Produce physics-consistent forecasts
- Provide interpretable summaries

What the system will not do

- Perform long term climate modeling
- High-resolution global simulations
- Real-time data ingestion from sensors
- Serve as a public-facing operational tool

2.1. User Characteristics

The system aims to target two main user classes:

- **Non-technical users** (primary): Users like farmers, policymakers, disaster managers, and the general public, who need simple, readable forecasts without requiring to understand underlying models, These user must have basic computer and mobile skills like operating a simple interface or inputting location data
- **Technical users** (secondary): Researchers, developers, or meteorologists who may approach raw prediction data for analysis or research. These users have knowledge of ML and physics constraints, and will evaluate system's accuracy and consistency.

2.2. Operational Environment

The software will work on standard desktop or laptop computers, essentially in a development and testing setup. It will use Python 3.x as the core programming language, with compatibility across Windows, Linux, macOS operating systems. Key components of this software are with open-source libraries for data handling (e.g., xarray for multidimensional arrays, numpy and pandas for processing) and ML frameworks (e.g., PyTorch for PINNs). The system must coexist with other applications like Jupyter Notebooks for prototyping. There will be no need of any specialized servers but may requires moderate GPU support for training efficiency.

2.3. System Constraints

The system faces several constraints that limit its design and implementation:

- **Software constraints**: It will only depend on open source tools. There is no need for proprietary software or custom LLM fine-tuning beyond pre-trained models. To maintain compatibility will Python libraries avoid use deprecated versions.
- **Hardware constraints:** There is no need of supercomputers, can be used on consumer grade hardware which may increase training time.
- **Cultural/Legal Constraints:** Outputs will be shown in Natural Language; may support multilingual. Compliance with data usage terms from public datasets like ERA5.
- **Environmental Constraints:** Assumes a quiet, stable indoor setup for development; no audio alerts or features for noisy environments.
- **User constraints**: There will be a simple interface for non experts which will use graphical elements.
- **Component constraints:** Model sizes can vary due to libraries like Hugging Face Transformers.

3. External Interface Requirement

This specifies how the external elements interact and are related to the system. At a high level , data is fetched or taken from online repositories and is processed by system. System avoids any complex real time integrations.

3.1. Hardware Interface

Standard hardware interfaces are required: keyboard and mouse for user input, a display monitor or a screen for viewing outputs (e.g., forecasts and summaries), and optional GPU for faster computations. Specialized devices such as Weather sensors etc. are not required.

3.2. Software Interface

Interfaces include:

- **Datasets**: For datasets a connection to ERA5 and WeatherBench is required. Which can be via file downloads or APIs (e.g., using Python requests library). Data formats can be: NetCDF for multidimensional weather data such as data with latitude and longitude.
- **Models**: Pre-trained LLMs from Hugging Face (e.g., GPT-2) will be integrated where PINN outputs (numerical arrays) are passed as text prompts to generate summaries. Shared data will include predictions in JSON format.

- **Libraries**: PyTorch will be used for neural network training. Tensor operation services will also be used.

3.3. Communication Interface

It is only limited to initial data collection.

- HTTPS for secure downloads from different dataset resources or providers.
- Large files (up to several GBs) transfer rate will be supported.
- No real time network requirements such as email notifications etc.

4. Functional Requirements

The system must support the following business-related functions:

- **Data Collection and Preprocessing**: Data will be automatically downloaded and normalized from ERA. Grid resolution will be reduced to 1°x1° and divided in separate training, validation and test sets.

- **PINN Forecasting**: A Neural Network will be trained that uses physical equations to predict short-term weather variables. It will minimize both data loss and physics residuals.

- **LLM Interpretation**: PINN outputs will be fed into pre-trained LLM which will generate natural language descriptions.

- **Evaluation**: Predictions will be compared with standards or baselines like LSTM/CNN using metrics such as RMSE, MAE and physics consistency checks will also be performed.

- **User Interface**: A basic command line or web based interface will be provided so that the use can input parameter and then the results will be displayed.

5. Non-Functional Requirements

5.1. Performance Requirements

- Forecasts should be produced by the system under 5 minutes on moderate hardware.
- Should be able to handle datasets up to 10GB.
- Precision targets such as RMSE < 2°C for temperature, MAE < 5% for humidity.
- 95% uptime during testing
- Error handling for data issues

5.2. Safety Requirements

As it will be a non-critical prototype so formal safety certifications are not needed. However, safeguards include:

- Outputs validation to prevent physically impossible results.
- A warning from system for the users that forecasts are experimental and are not for life safety decisions.

5.3. Security Requirements

Basic measures:

- User authentication is not required but the downloaded data will be protected from tampering via checksums.
- User or personal data will not be collected.

5.4. User Documentation

Deliverable includes:

- A user manual in form of pdf as an installation guide, usage steps, and troubleshooting of any possible errors. A brief tutorial video or GitHub READ ME online help.

6. Assumptions and Dependencies

6.1. Assumptions:

- Datasets like ERA5 are freely accessible.
- Minimum hardware requirements for Python/ML libraries.
- Pre-trained LLMs offer reliable explanations.

6.2. Dependencies:

- External Libraries e.g., PyTorch, Hugging Face Transformers etc.
- Availability of team resources.

7. References

Ref. No.	Document Title	Date	Source
[1]	Richardson, L. F. (1922). Weather Prediction by Numerical Process	1922	Cambridge University Press
[2]	Pathak, J. et al. (2022). FourCastNet	2022	NVIDIA Research
[3]	Lam, R. et al. (2023). GraphCast	2023	Google DeepMind
[4]	Karniadakis, G. E. et al. (2021). Physics-informed Machine Learning	2021	Nature Reviews Physics
[5]	Hugging Face (2024). Transformers Library	2024	https://huggingface.co