MILESTONE – 3

Exoplanet Habitability Prediction System

MILSTONE-3 THEORETICAL DOCUMENTATION (BACKEND & FRONTEND)

INTRODUCTION

The Exoplanet Habitability Prediction System is a web-based application designed to predict whether a given exoplanet is habitable or not. The system uses Machine Learning techniques to analyze planetary and stellar parameters and provide a habitability score. The project consists of two major modules: Backend Development and Frontend Development. Both modules work together to provide an interactive, user-friendly prediction platform.

SYSTEM ARCHITECTURE

The system follows a client-server architecture where the user interacts with the frontend interface. The frontend sends user data to the backend server, the machine learning model processes the input, and the result is returned to the user interface.

BACKEND DOCUMENTATION

The objective of the backend module is to host the trained machine learning model, provide an API for predictions, process user input, and return habitability results. The backend is developed using Python, Flask Framework, and machine learning libraries such as Scikit-learn, Pandas, and NumPy. It implements a REST API using JSON for communication.

The main API endpoint is POST /predict which accepts planetary parameters in JSON format and returns prediction results. The backend workflow includes receiving input, validating data, preprocessing it, passing it to the ML model, generating predictions, and returning the result as JSON.

FRONTEND DOCUMENTATION

The objective of the frontend module is to provide a user-friendly interface, accept user input, communicate with the backend API, and display prediction results. The frontend is built using HTML, CSS, JavaScript, and Bootstrap Framework.

The frontend consists of three main files: index.html, style.css, and script.js. The UI is designed to be clean, responsive, and easy to use. Input forms are provided for entering planetary and stellar parameters with proper validation.

The frontend communicates with the backend using JavaScript Fetch API and asynchronous requests. User input is collected, converted to JSON, and sent to the backend. Prediction results are received and displayed using Bootstrap cards and alerts.

ERROR HANDLING AND TESTING

The system handles invalid inputs and backend connection errors gracefully. User-friendly messages are displayed without page reloads. The complete integration flow from user input to prediction result has been tested successfully.

CONCLUSION

The Exoplanet Habitability Prediction System successfully integrates machine learning, backend API, and frontend interface to provide a functional and user-friendly prediction platform.

FUTURE SCOPE

The system can be enhanced by adding more parameters, improving model accuracy, deploying online, adding visualizations, and supporting multiple prediction models.