Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	11 February 2026
Team ID	LTVIP2026TMIDS90283
Project Name	Rising waters: a machine learning approach to flood prediction

1. Product Backlog

The product backlog represents all functional and non-functional requirements required to complete the Flood Prediction System. The backlog was created based on system features, machine learning workflow, and web deployment requirements.

The major epics identified for the project are:

Data Collection and Analysis

Data Preprocessing

Model Building

Web Application Development

User Interface Design

Testing and Deployment

Each epic was broken down into smaller user stories to ensure systematic implementation.

2. Sprint Planning

The project was divided into multiple sprints to manage development efficiently.

Sprint 1 – Data Handling & Preparation

Focus:

- Collect dataset
- Perform data cleaning
- Handle missing values
- Feature selection
- Apply scaling techniques

Goal:
Prepare a clean and structured dataset ready for machine learning model training.

Sprint 2 – Model Building & Evaluation

Focus:

- Train multiple algorithms (Decision Tree, Random Forest, KNN, XGBoost)

Evaluate model performance

Compare models

Select best-performing model

Goal:
Identify the most accurate model for flood prediction.

Sprint 3 – Model Saving & Backend Integration

Focus:

Save trained model using Joblib

Save scaler

Create Flask backend

Implement routing and prediction logic

Goal:
Integrate machine learning model with web application.

Sprint 4 – Frontend & Deployment

Focus:

- Design HTML pages (home, prediction form, result page)
- Apply CSS styling
- Connect frontend with backend
- Test end-to-end prediction flow

Goal:
Deploy a fully functional flood prediction web application.

---

3. User Stories (Example)

- As a user, I can enter rainfall and cloud data through a web form.
- As a user, I can receive instant flood prediction results.
- As a developer, I can train and evaluate multiple ML models.
- As a system, I can scale input data before prediction.
- As a system, I can load the saved model for real-time prediction.

---

4. Estimation & Story Points

Story points were assigned based on complexity:

- Data preprocessing tasks – Medium effort
- Model training and evaluation – High effort
- Flask integration – Medium effort
- UI design – Low to medium effort

The workload was distributed across sprints to ensure timely completion and balanced development effort.

---

5. Velocity and Progress Tracking

The team's progress was monitored using sprint planning and story point completion. Each sprint was designed to achieve a specific milestone of the project. Work completion was measured based on:

- Model accuracy achieved
- Successful integration
- Functional UI testing
- End-to-end system validation

This ensured continuous improvement and timely delivery of the project.

---

6. Burndown Logic

The burndown approach was used conceptually to track remaining tasks over time. As tasks such as data preprocessing, model evaluation, and Flask deployment were completed, the remaining workload decreased steadily across sprints. This helped maintain development focus and avoid delays.

Conclusion

The planning logic of the Flood Prediction System followed an agile-based sprint approach. By dividing the project into manageable phases such as data handling, model development, backend integration, and deployment, the project was completed systematically and efficiently.

This structured planning ensured better task management, improved productivity, and successful implementation of the end-to-end machine learning application.