WeatherPlus: Your Personalized Weather Companion

Keyur Patel - 16010421073
Ronit Mehta - 16010421056
Batch - A2
Exp-3(SPMP)

March 2024

Aim:

To Prepare Software Project Management Plan (SPMP)

Resources:

Internet Explorer, GanttProject, LaTex Editor

1 Introduction

1.1 Project Overview:

The WeatherPlus application is designed to provide users with accurate and real-time weather information based on their location. The primary purpose is to offer a seamless and personalized weather experience, empowering users to make informed decisions based on up-to-the-minute meteorological data.

1.2 Project Deliverables:

The deliverables for your weather application website can be categorized into two main groups: Internal deliverables and External deliverables.

1.2.1 Internal Deliverables:

- **Project plan:** This document outlines the project scope, timeline, resources, and risks. It should be updated regularly to reflect any changes in the project.
- **Technical documentation:** This includes detailed information about the system architecture, code documentation, API documentation, and deployment procedures.
- **User stories and acceptance criteria**: These define the functionalities of the application and the criteria for considering those features complete.
- **Test plans and results:** This includes the test cases and their results, ensuring the application functions as intended.

1.2.2 External Deliverables:

- **Frontend Development:** This involves the actual coding of the website's frontend using HTML, CSS, and JavaScript, ensuring responsiveness and compatibility with various devices and browsers.

- **Backend Development:** Development of the server-side logic and database management for storing user preferences, location data, and weather information.
- **Weather Data Integration:** Integration with weather APIs or services to fetch real-time weather data based on user location or search queries.
- **User Registration and Authentication:** Implementation of user authentication and registration system if required, to personalize user experience and save preferences.
- **Search Functionality:** Implementation of a search feature allowing users to search for weather forecasts by location, zip code, or geographic coordinates.
- **Weather Forecast Display:** Display of weather forecasts for different locations, including current conditions, hourly forecasts, and multi-day forecasts.

2 Project Organization:

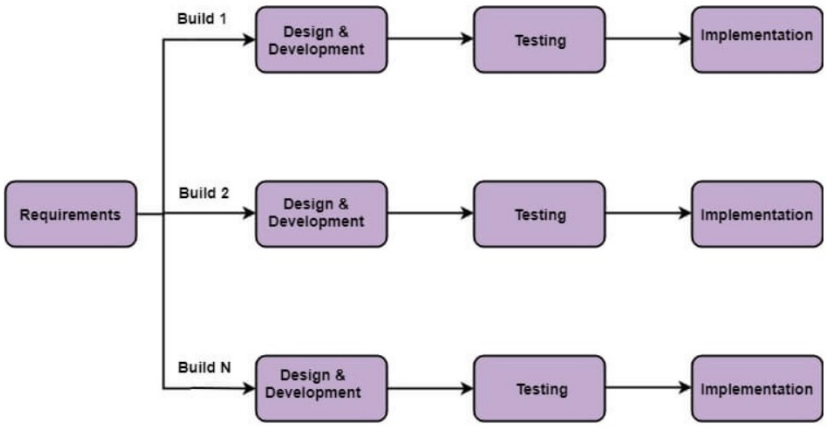

Figure 1: Incremental model

2.1 Software Process Model

For the development of the WeatherPlus application, we will utilize the Incremental Software Development Model. This model involves breaking down the software development process into smaller, manageable parts and delivering them incrementally.

2.2.1 Flow of Information and Work Products:

1. ***Requirements Analysis:*** Initial analysis to identify and prioritize core functionalities.
2. ***Incremental Development:*** Development is organized into increments, with each increment adding specific features or enhancements to the application.
3. ***Integration:*** Integration of new increments with the existing application to ensure compatibility and seamless operation.
4. ***User Feedback:*** Regular feedback from users after each increment is released, influencing subsequent development.

2.2.2 Reviews to be Conducted:

1. ***Incremental Reviews:*** After the completion of each increment, a review is conducted to assess functionality, usability, and gather feedback.
2. ***User Feedback Sessions:*** Regular sessions with users to collect feedback on the implemented features and prioritize future increments.
3. ***Code Reviews:*** Ongoing code reviews to maintain code quality and identify potential issues.

2.2 Roles and Responsibilities

Project Team Structure:

(a) **Project Manager (PM):**
- **Responsibilities:** Overall project coordination, planning, and scheduling.

(b) **Product Owner (PO):**
- **Responsibilities:** Represents end-users, provides feedback on deliverables.

(c) **Development Team:**
- **Roles:** Backend Developers, Frontend Developers
- **Responsibilities:** Implement features based on user stories and design specifications, Collaborate with QA for testing.

(d) **Quality Assurance (QA) Team:**
- **Responsibilities:** Develops test plans and test cases, Conducts functional, performance, and regression testing.

(e) **UX/UI Designer:**
- **Responsibilities:** Designs user interfaces and experiences, Provides design assets for implementation.

Information Flow:

- **Design to Development:**
  (a) The UX/UI designer provides design assets to the development team.
  (b) Design specifications and user stories are discussed in sprint planning meetings.
- **Development to QA:**
  (a) Developers share their completed increments with the QA team during sprint reviews.
  (b) Continuous communication during daily stand-ups regarding the development progress.
- **QA to Development:**
  (a) QA team reports bugs and issues to the development team.
  (b) Collaboration during sprint reviews to address and resolve identified issues.

2.3 Tools and Techniques

2.3.1 Development Methodology: Incremental Model:

A method where the project is divided into small, manageable parts or increments. Each increment builds upon the previous one, allowing for the delivery of partial functionality.

2.3.2 Notations:

UML diagrams (Class diagrams, Sequence diagrams) for system architecture.

2.3.3 Programming Languages:

- HTML
- CSS
- JavaScript
- Firebase

2.3.4 Techniques:

Object-Oriented Programming (OOP): OOP principles will be applied throughout the development.

2.3.5 Tools:

- Lucidchart or draw.io for creating visual diagrams.
- Figma or Adobe XD for UI/UX design.
- Latex Overleaf for textual documentation.
- Visual Studio Code: VS Code is chosen for its versatility, robust features, and strong community support, contributing to a seamless development experience.
- Git for version control.
- GitHub for repository hosting.

3 Project Management Plan:

3.1 Tasks:

3.1.1 Task-1: Requirements Gathering

3.1.1.1 Description: Identify and document functional and non-functional requirements for the weather app.

3.1.1.2 Deliverables and Milestones:

- Requirements Document (Milestone 1): A comprehensive document outlining both functional and non-functional requirements.
- Includes details such as user expectations, system capabilities, and any regulatory or compliance requirements.

3.1.1.3 Resources Needed: Google

3.1.1.4 Dependencies and Constraints: Availability of subject matter experts.

3.1.1.5 Risks and Contingencies:

- **Risk:** Misunderstanding requirements.
- **Contingency:** Regular feedback and validation with users.

3.1.2 Task-2: Design and Architecture

3.1.2.1 Description: Create high-level and low-level design for the weather app. Design the database schema.

3.1.2.2 Deliverables and Milestones:

- Structured representation of the database, including tables, relationships, and constraints.(Milestone 2)
- A blueprint for organizing and storing data related to weather information.

3.1.2.3 Resources Needed:

- **Database Designers:** Experts in designing efficient and effective database structures. Responsible for creating a well-organized database schema.
- **Design Tools (e.g., Figma, draw.io):** Software tools for creating visual representations of the system design. Facilitate collaboration and documentation of design artifacts.

3.1.2.4 Dependencies and Constraints: Availability of technical experts.

3.1.2.5 Risks and Contingencies:

- **Risk:** Changes in requirements affecting design.
- **Contingency:** Adopt an iterative design approach.

3.1.3 Task-3: Development

3.1.3.1 Description: Implement front-end and back-end functionalities of the weather app. Set up the database and integrate APIs for weather data. This task involves the actual coding and development phase, focusing on building the core functionalities of the weather app.

3.1.3.2 Deliverables and Milestones:

- The actual codebase that implements the specified features and functionalities.(Milestone-3)
- Version-controlled and regularly updated with new developments.(Milestone-4)
- Allows for early testing and validation of implemented features.(Milestone-5)

3.1.3.3 Resources Needed:

- Skilled individuals responsible for writing the front-end and back-end code.
- Collaborate to ensure consistency across different components.
- Manage and optimize the database setup. Platforms or environments for developers to write, test, and deploy code.

3.1.3.4 Dependencies and Constraints:

- **Dependency:** The development team relies on the availability of suitable environments for coding, testing, and deployment.
- **Constraint:** Delays or limitations in accessing development environments may hinder progress.

3.1.3.5 Risks and Contingencies:

- **Risk:** Technical challenges in integrating APIs.
- **Contingency:** Conduct thorough API testing and have backup sources for weather data.

3.1.4 Task-4: Testing

3.1.4.1 Description: Perform unit testing, integration testing, and system testing. Conduct user acceptance testing (UAT).

3.1.4.2 Deliverables and Milestones:

- Test plans and cases (Milestone 6)
- Test results
- UAT documentation

3.1.4.3 Resources Needed:

- QA engineers
- End-users for UAT
- Testing environments

3.1.4.4 Dependencies and Constraints: Availability of testing environments.

3.1.4.5 Risks and Contingencies:

- **Risk:** Critical bugs discovered late in the testing phase.
- **Contingency:** Regular and thorough testing throughout the development process.

3.2 Assignments:

3.2.1 Assignment for Task-1: SPMP-01

i. **Team member-1**: Lead the requirements gathering, document functional requirements and the creation of the requirements document.

ii. **Team member-2**: Assist in gathering non-functional requirements.

3.2.2 Assignment for Task-2: SPMP-02

i. **Team member-1**: Lead the creation of the high-level design document, focusing on the overall system architecture.

ii. **Team member-2**: Lead the development of the low-level design document, with a focus on structure of a database, including tables, relationships, and data types.

3.2.3 Assignment for Task-3: SPMP-03

i. **Team member-1**: Take the lead in implementing front-end functionalities and setting up the development environment.

ii. **Team member-2**: Lead the implementation of back-end functionalities, including database setup and integration of weather data APIs.

3.2.4 Assignment for Task-4: SPMP-04

i. **Team member-1**: Lead the creation of test plans and cases for unit testing and system testing.

ii. **Team member-2**: Focus on conducting user acceptance testing (UAT) and providing end-user perspectives.

3.3 Timetable:

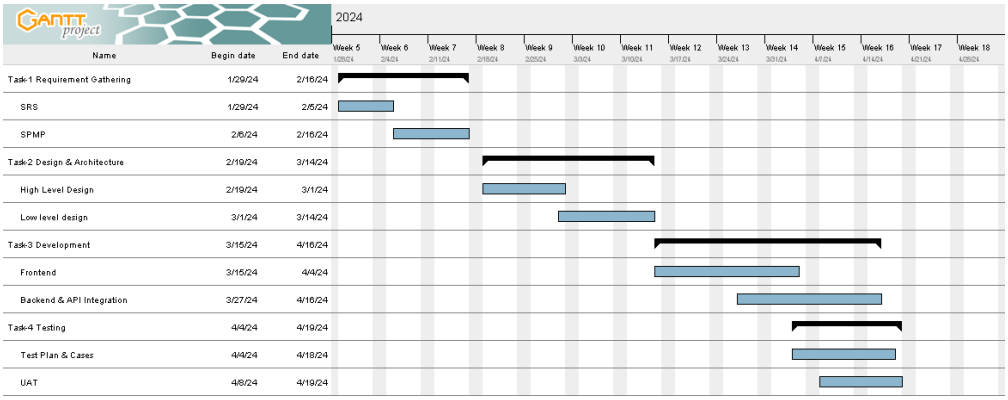

Figure 2: Gantt Chart

Questions:

1. Can a project be undertaken without a plan? What are the possible consequences?

Undertaking a project without a plan is generally not recommended, as it introduces a variety of risks and challenges that can lead to project failure or inefficiencies. Here are some possible consequences of initiating a project without a proper plan:

1. **Unclear Objectives:**Without a plan, the project's objectives may be unclear or not well-defined. This can lead to confusion among team members, stakeholders, and other involved parties about the project's goals and desired outcomes.

2. **Scope Creep:** Lack of planning may result in uncontrolled changes to the project scope. This phenomenon, known as scope creep, can lead to an increase in project complexity, cost, and duration, making it difficult to meet the original project goals.

3. **Budget Overruns:** Without a detailed budget plan, there is a higher risk of overspending. Unforeseen costs may arise during the project, leading to budget overruns and financial strain on the organization.

4. **Inadequate Resource Allocation:** Without proper planning, it may be challenging to allocate resources efficiently. This includes human resources, equipment, and materials. Poor resource allocation can result in delays, bottlenecks, or a lack of necessary expertise.

5. **Unrealistic Timelines:** Projects without a plan may lack realistic and well-defined timelines. This can lead to missed deadlines, project delays, and an overall inability to deliver the project within the expected timeframe.

6. **Quality Issues:** Insufficient planning may result in a lack of focus on quality management. Without defined quality assurance processes, there is a higher likelihood of delivering a product or service with defects or issues.

7. **Communication Breakdown:** A project plan is a critical communication tool. Without it, there may be a breakdown in communication among team members, stakeholders, and other involved parties. This lack of communication can lead to misunderstandings and conflicts.

8. **Risk Management Challenges:** Planning is essential for identifying and managing project risks. Without a risk management plan, unexpected issues may arise, and the project team may be unprepared to handle them effectively.

9. **Stakeholder Dissatisfaction:** Insufficient planning can lead to misalignment between project deliverables and stakeholder expectations. This can result in dissatisfaction among stakeholders, damaging relationships and the project's overall success.

10. **Difficulty in Monitoring and Controlling:** A project plan serves as a baseline for monitoring and controlling project activities. Without it, project managers may struggle to track progress, identify deviations, and implement corrective actions.

Outcomes

CO2: Describe software planning and management.

Conclusion

After completing this experiment we have successfully created a SPMP for our project.

References

Books:

1. Roger S. Pressman, Software Engineering: A practitioners Approach, 7th Edition, McGraw Hill, 2010.

2. Ian Somerville, Software Engineering, 9th edition, Addison Wesley, 2011.

3. John M. Nicholas, "Project Management for Business and Technology", 2nd edition, Pearson Education