AutoCareAI

SPRINT RETROSPECTIVE REPORT

Drive Happy, Drive Safe.

IS2108 Final Project - Group 17

Project Overview

- Project Name: AutoCareAI
- Sprint Duration: Sprint 1: February 03 – February 14, Sprint 2: February 15 – February 27
- Scrum Team: Group 17
- Sprint Objectives:
  - Sprint 1: Develop core functionalities (User Registration and Login, Car Profile Management).
  - Sprint 2: Implement AI-driven recommendations, Service Booking, Notifications, and Admin Dashboard.

Sprint 1 Retrospective (Feb 03 – Feb 14)

1. Sprint Goals & Key Deliverables

Planned:

- Implement User Registration & Login (Email Verification, Password Management).
- Develop Car Profile Management (Adding, Editing, and Viewing Vehicles).
- Conduct initial unit testing & integration of core features.

Delivered:

- Successfully implemented User Registration & Login with email verification.
- Car Profile Management features completed & tested.
- Integrated basic security measures (authentication, session management).

2. What Went Well?

- Task Allocation & Responsibility: Team members had clearly assigned tasks, ensuring ownership.
- Successful Core Feature Development: The authentication system and profile management were implemented with minimal bugs.
- Effective Communication: Daily stand-up meetings allowed quick problem resolution.

3. What Didn’t Go Well?

- UI Finalization Delays: Due to misalignment in requirements, the UI took longer than expected.
- Testing Constraints: Limited time for end-to-end testing, leading to minor defects.
- Dependency Issues: The backend and frontend teams had synchronization issues, which delayed API testing.

4. Key Challenges & Risk Analysis

Challenge	Impact	Mitigation Strategy
UI design misalignment	Delayed front-end development	Ensure early design approvals before implementation
Lack of test coverage	Minor defects in login & profile management	Allocate buffer time for QA & system testing
Backend & frontend sync issues	Slowed down API testing	Implement mid-sprint check-ins for dependencies

5. Action Plan for Improvement

- Conduct mid-sprint reviews to align expectations early.
- Improve time estimation for testing and debugging phases.
- Implement a collaborative sprint planning approach for better synchronization.

Sprint 2 Retrospective (Feb 15 – Feb 27)

1. Sprint Goals & Key Deliverables

Planned:

- Implement AI-driven Service Recommendations.
- Develop a Service Booking System with cancellation & rescheduling.
- Integrate Real-time Notifications & Admin Dashboard.
- Conduct system-wide testing and optimize workflows.

Delivered:

- AI-powered recommendations are successfully integrated into the system.
- Service Booking System completed, including cancellation & rescheduling.
- Notifications and real-time booking updates were implemented & tested.

2. What Went Well?

- Smooth AI Feature Integration: The recommendation engine performed well in early tests.
- Successful Implementation of Booking & Payment Modules: The system handled various booking scenarios effectively.
- Cross-team Collaboration Improved: Developers and testers worked closely, reducing integration issues.

3. What Didn’t Go Well?

- Increased Complexity in AI Training: Some recommendations required fine-tuning based on real-world data.
- Feature Dependencies Delayed Testing: The booking system relied on notifications, leading to late-stage changes.
- Performance Optimization Challenges: Initial testing showed slow response times for AI-driven queries.

4. Key Challenges & Risk Analysis

Challenge	Impact	Mitigation Strategy
AI recommendation inconsistencies	Reduced accuracy of service suggestions	Implement AI model refinement cycle post-sprint
Dependency on booking module for notifications	Late-stage debugging issues	Introduce parallel testing phases for interdependent features
Performance optimization	Slower load times for AI queries	Optimize query handling & data caching

5. Action Plan for Improvement

- Introduce buffer time for AI feature refinements after deployment.
- Improve parallel testing for modules with high interdependencies.
- Conduct performance benchmarking before finalizing AI queries.
- Enhance data-driven decision-making by collecting real-time feedback from test users.

6. Lessons Learned

- Prioritize AI fine-tuning earlier in sprints to allow adjustments.
- Improve dependency tracking to avoid delays in linked features.
- Allocate dedicated testing slots for AI-driven components to ensure accuracy.