

**CAPSTONE PROJECT REPORT**

**Report 2 – Project Management Plan**

– HCM, December 2025 –

**Table of Contents**

[I. Record of Changes 3](#_heading=h.idg1wez4ntzw)

[II. Project Management Plan 4](#_heading=h.4t6n441vebep)

[1. Overview 4](#_heading=h.p3rf54ukwv36)

[1.1 Scope & Estimation 4](#_heading=h.1eecnlclla1z)

[1.2 Project Objectives 4](#_heading=h.djde01ecuc6y)

[1.3 Project Risks 5](#_heading=h.7rhltoowpv56)

[2. Management Approach 5](#_heading=h.nwf1r6eldsna)

[2.1 Project Process 5](#_heading=h.9c985s7wozkp)

[2.2 Quality Management 6](#_heading=h.jo942mb4ohnn)

[2.3 Training Plan 7](#_heading=h.bmn10t78jtyp)

[3. Project Deliverables 7](#_heading=h.dj4q1isjh6j1)

[4. Responsibility Assignments 8](#_heading=h.nrmr666yxbxs)

[5. Project Communications 9](#_heading=h.e8f2wmockmh)

[6. Configuration Management 9](#_heading=h.guwj0yf77vma)

[6.1 Document Management 9](#_heading=h.800p7mpttqq2)

[6.2 Source Code Management 9](#_heading=h.2nnq44s7km8m)

[6.3 Tools & Infrastructures 10](#_heading=h.s8dajfib6bvv)

# I. Record of Changes

| **Date** | **A\* M, D** | **In charge** | **Change Description** |
| --- | --- | --- | --- |
| 9-06-2025 | A | Nguyễn Anh Huy | Initial project proposal draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

\*A - Added M - Modified D - Deleted

# II. Project Management Plan

## 1. Overview

### 1.1 Scope & Estimation

### 

| **#** | **WBS Item** | **Complexity** | **Est. Effort (man-days)** |
| --- | --- | --- | --- |
| **1** | **Account & Identity** |  | **20** |
| 1.1 | Account registration & login | Simple | 4 |
| 1.2 | User profile management | Medium | 6 |
| 1.3 | Instructor verification (KYC) | Complex | 10 |
| **2** | **Booking & Session Management** |  | **36** |
| 2.1 | Search & book instructor | Medium | 8 |
| 2.2 | Schedule & route management | Medium | 8 |
| 2.3 | Real-time GPS tracking | Complex | 10 |
| 2.4 | Payment integration | Medium | 6 |
| 2.5 | Instructor session confirmation & notes | Simple | 4 |
| **3** | **Communication & Engagement** |  | **21** |
| 3.1 | Notifications & reminders | Medium | 6 |
| 3.2 | Chat system (novice driver ↔ instructor) | Complex | 7 |
| 3.3 | Feedback & rating system | Simple | 4 |
| 3.4 | Booking history & logs | Medium | 4 |
| **4** | **Administration & Compliance** |  | **28** |
| 4.1 | Inspector verification dashboard | Medium | 7 |
| 4.2 | Admin user & role management | Medium | 6 |
| 4.3 | Reports & monitoring dashboard | Complex | 9 |
| 4.4 | Security & authentication | Complex | 6 |
| **5** | **System Quality (Non-Functional)** |  | **10** |
| 5.1 | Responsive UI/UX design | Medium | 4 |
| 5.2 | Scalability & performance tuning | Complex | 6 |
| ***Total Estimated Effort (man-days)*** | | | 115 |

### 

### 1.2 Project Objectives

| **#** | **Testing Stage** | **Test Coverage** | **No. of Defects** | **% of Defect** | **Notes** |
| --- | --- | --- | --- | --- | --- |
| 1 | Reviewing | 100% documents | < 5 | < 5% | Peer reviews |
| 2 | Unit Test | 90% functions | < 10 | < 10% | Automated unit tests |
| 3 | Integration Test | 85% modules | < 15 | < 15% | API + UI integration |
| 4 | System Test | 80% scenarios | < 20 | < 20% | Functional + NFR tests |
| 5 | Acceptance Test | 100% features | < 5 | < 5% | UAT by supervisor |

### 1.3 Project Risks

| **#** | **Risk Description** | **Impact** | **Possibility** | **Response Plans** |
| --- | --- | --- | --- | --- |
| 1 | Limited technical knowledge of React Native & frameworks | High | Medium | Allocate training sessions, assign mentor, allow buffer time |
| 2 | Integration issues (microservices, payment gateway) | High | Medium | Early integration testing, sandbox environment, CI/CD pipeline |
| 3 | Data privacy & security vulnerabilities | High | Low | Apply secure coding practices, perform regular security reviews & audits |
| 4 | Solution does not fully meet real-world requirements | High | Medium | Conduct thorough business analysis, gather user feedback, benchmark against existing apps |
| 5 | Team-related risks (conflicts, unavailability) | Medium | Medium | Regular team meetings, task redistribution, conflict resolution practices |
| 6 | Tight deadlines & project delays | Medium | Medium | Prioritize critical tasks, redistribute workload, adjust timeline when necessary |

## 2. Management Approach

The project will be managed using the **Waterfall model**, where each phase is completed sequentially before moving to the next. This ensures that all requirements are clearly defined and agreed upon before implementation begins, reducing scope creep and uncertainty.

### 2.1 Project Process

The team applies the **Waterfall development model**, following these sequential phases:

* **Requirement Analysis**
  + Conducted initial meetings to gather all customer needs and define detailed system requirements.
  + Documented requirements clearly and obtained stakeholder approval before moving to design.
* **System Design**
  + Designed the overall system architecture, database schema, and UI/UX mockups based on approved requirements.
  + Prepared technical specifications for development teams to follow.
* **Implementation**
  + With a fixed design, coding was carried out according to the original plan.
  + Developers followed predefined specifications, minimizing scope change during this phase.
* **Testing**
  + Performed unit tests, integration tests, and full system tests to ensure quality.
  + Verified that all functional and non-functional requirements were met.
* **Deployment**
  + Delivered the completed product to staging and production environments.
  + Ensured smooth installation and configuration.
* **Maintenance**
  + Provided ongoing support to fix defects, optimize performance, and incorporate user feedback.
  + Ensured long-term system stability and scalability.

### 2.2 Quality Management

To ensure that the project meets its quality objectives, the DriveMate team will adopt a comprehensive quality management approach. This approach emphasizes defect prevention, thorough reviews, structured testing, and continuous documentation to maintain high product reliability and usability throughout the software development lifecycle.

**Defect Prevention:**

* **Peer Reviews:** Regular peer reviews of code, APIs, and UI components will be conducted to detect defects early, ensure adherence to coding standards, and facilitate knowledge sharing among team members.
* **Coding Standards:** Developers will follow defined conventions and best practices (e.g., naming rules, secure coding practices, consistent error handling) to minimize common errors and vulnerabilities.

**Reviewing:**

* **Requirements Reviews:** User and functional requirements will be reviewed with the supervisor and team to ensure alignment with business needs, especially regarding booking flows, instructor verification, and payment integration.
* **Design Reviews:** System architecture and design diagrams (UML, database schema, API design) will be collaboratively reviewed to validate scalability, security, and feasibility before moving into implementation.

**Testing:**

* **Unit Testing:** Each component (e.g., booking module, payment gateway, GPS tracking) will be tested individually using automated unit tests (xUnit for backend, Jest for frontend) to ensure functional correctness.
* **Integration Testing:** Interactions between services (e.g., mobile app ↔ backend API ↔ payment gateway) will be tested to confirm smooth communication across modules. CI/CD pipelines will help automate integration checks.
* **System Testing:** Full end-to-end testing will be carried out in an environment that mirrors production. Scenarios include booking a session, verifying instructors, making payments, and live GPS tracking to ensure system reliability.

**Documentation:**

* **Test Documentation:** Test plans, cases, and execution reports will be maintained and updated to support transparency, reproducibility, and future maintenance.
* **Defect Tracking:** All issues identified during testing will be logged into a defect tracking system (e.g., GitHub Issues, Jira). Regular triage meetings will prioritize defects based on severity and business impact to ensure timely resolution.

### 2.3 Training Plan

| **Training Area** | **Participants** | **When, Duration** | **Waiver Criteria** |
| --- | --- | --- | --- |
| React Native (Mobile App) | CuongTM, NganTT, TriLM | 15/09/2025 – 16/09/2025 (2 days) | Waived if prior experience in mobile dev |
| ASP.NET Core Microservices (API) | TriLM, HuyNA, GiangPL | 17/09/2025 – 19/09/2025 (3 days) | Waived if prior .NET project experience |
| Git & GitHub Workflow | TriLM, HuyNA, GiangPL, CuongTM, NganTT | 20/09/2025 (1 day) | Waived if proven Git usage in past projects |
| Payment Integration (VNPay, ZaloPay, PayOS) | TriLM, HuyNA, GiangPL | 05/10/2025 (2 day) | Waived if experience in fintech APIs |

## 3. Project Deliverables

| **#** | **Deliverable** | **Due Date** | **Notes** |
| --- | --- | --- | --- |
| 1 | Project Introduction (Report 1) | 20/08/2025 | Project introduction document |
| 2 | Project Management Plan (Report 2) | 21/08/2025 | Project management plan document |
| 3 | Software Requirement Specification (R3) | 22/08/2025 | Context diagram, use case diagram,... |
| 4 | Software Design Document (Report 4) | 22/08/2025 | System architecture, class diagram, sequence diagram,... |
| 5 | Backend Code Package | 05/09/2025 | Core services, APIs, payment integration |
| 6 | Frontend Web Code Package | 05/09/2025 | User dashboards, booking management |
| 7 | Mobile App Code Package | 05/09/2025 | React Native app with GPS, booking, chat |
| 8 | Software Test Documentation (Report 5) | 17/11/2025 | Test strategy, test plan, test case |
| 9 | Software User Guides (Report 6) | 20/11/2025 | Installation guides, user manual |
| 10 | Final Project Report + Defense | 30/11/2025 | Consolidated report + presentation |

## 4. Responsibility Assignments

| **Responsibility** | **Trí (Leader)** | **Giang** | **Huy** | **Cường** | **Ngân** |
| --- | --- | --- | --- | --- | --- |
| Project Planning & Tracking | S | S | D | I | I |
| Project Introduction (Report 1) | S | S | D | I | I |
| Project Management Plan (Report 2) | S | R | D | S | I |
| Software Requirement Specification (R3) | D | S | D | S | S |
| Software Design Document (R4) | S | S | D | S | S |
| Backend Code Package | D | D | D | I | I |
| Frontend Web Code Package | D | S | S | D | S |
| Mobile App Code Package | D | S | S | D | S |
| Software Test Documentation (R5) | S | S | D | R | I |
| Software User Guides (R6) | D | S | D | I | R |
| Final Project Report + Defense | D | R | R | S | S |
| UI/UX design prototype | S | S | D | D | D |
| Code Package Back-end | D | D | D | R | R |
| Code Package Front-end | S | R | R | D | D |
| Code Package Mobile | S | R | R | D | S |

## 5. Project Communications

## 

| **Communication Item** | **Who/Target** | **Purpose** | **When, Frequency** | **Type, Tool, Method(s)** |
| --- | --- | --- | --- | --- |
| Supervisor (Lecturer) | Mr. Nguyễn Ngọc Lâm | - Provide document template- Instruct the project team- Review deliverables- Supervise project status- Receive project report | Weekly | Zalo |
| Meeting | Team members | - Collaborate on mutual tasks- Planning- Discuss project risks and issues | Weekly | Google Meet |
| Messages | Team members | - Share daily progress made on project tasks- Share current and potential roadblocks | Daily | Messenger |
| Git Repository | Development team | - Manage source code- Track code changes- Collaborate via pull requests/issues | Continuous | GitHub |
| Project Management Tool | Team members | - Track project tasks- Assign responsibilities- Monitor deadlines and progress | Continuous | GitHub |
| Documentation | Team | - Store project documents- Ensure version control- Facilitate easy access | Continuous | Google Drive |

## 

## 6. Configuration Management

### 6.1 Document Management

* **Document tools**: Microsoft Office (Word, Excel, PowerPoint), Google Docs, Google Sheets, Google Slides.
* **File Management**: Google Drive (used for storing, sharing, and maintaining version history of all project documents).
* **Change/Version Control**: Each document version is labeled with version numbers (v1.0, v1.1, …) and date of update. Major updates will be reviewed by the team and confirmed before official release.

### 6.2 Source Code Management

Our team uses **GitHub** to manage the source code version.

**Branching strategy**:

* **main** branch for stable releases.
* **develop** branch for integration.
* Feature branches (**feature/\***) for new features.
* Bugfix branches (**bugfix/\***) for issue resolution.

**CI/CD**: GitHub CI/CD pipelines are applied for continuous integration and deployment.

**Backup**: Codebase is mirrored weekly and stored on Google Drive as an extra backup.

### 6.3 Tools & Infrastructures

| **Category** | **Tools / Infrastructure** |
| --- | --- |
| Technology | NextJS (FrontEnd), React Native (Mobile App), .NET (BackEnd) |
| Database | Firebase, PostgreSQL |
| IDEs/Editors | Visual Studio Code, Visual Studio |
| Diagramming | LucidChart, DrawIO |
| Documentation | Microsoft Office, Google Docs/Sheets/Slides |
| Version Control | GitHub (Source Codes), Google Drive (Documents) |
| Deployment server | Amazon Web Services (AWS) |
| Project management | GitHub (Tasks, Defects Tracking) |