



CONSULTING SERVICES FOR THE UPDATING OF THE COMPREHENSIVE TRANSPARENCY SYSTEM OF MPW- BARMM THROUGH THE E-BARMM SYSTEM

INCEPTION REPORT

October 2025



**WOODFIELDS
CONSULTANTS, INC.**
A Planning and Engineering Consulting Firm



	WOODFIELDS CONSULTANTS, INC. DOCUMENT INFORMATION PAGE	Form No		
		SP-QMS-02F3		
		Rev No	Effectivity Date (mm/dd/yy)	Page/s
		1	09/04/17	1 of 1

DOCUMENT CODE:	FRP-25-001
PROJECT NAME:	Consulting Services for the Updating of the Comprehensive Transparency System of MPW-BARMM through the E-BARMM System
PROJECT CODE:	WCI-25-035
PROJECT MANAGER:	Dave Ervin Umali
DOCUMENT NAME:	Inception Report

REVISION HISTORY


REVISION STATUS	DOCUMENT REVISION DATE (DAY-MONTH-YEAR)	DESCRIPTION	AUTHORIZATION (NAME/POSITION)	SIGNATURE
0	15 October 2025	Inception Report	Dave Ervin Umali	

TABLE OF CONTENTS

TABLE OF CONTENTS	iii
EXECUTIVE SUMMARY	vii
Project Overview	vii
Project Objectives (TOR Section III)	vii
Proposed Solution	vii
Project Timeline	viii
Success Criteria	viii
CHAPTER 1 INTRODUCTION	1-1
1.1 Background	1-1
1.1.1 E-ARMM/E-BARMM System History	1-1
1.1.2 Why Update is Needed	1-1
1.2 Purpose of This Report	1-2
1.3 Project Stakeholders	1-2
1.4 Report Structure	1-3
CHAPTER 2 CURRENT SYSTEM ASSESSMENT	2-1
2.1 System Overview	2-1
2.1.1 Websites:	2-1
2.1.2 Coverage:	2-1
2.2 Technology Stack	2-1
2.2.1 Web Application	2-1
2.2.2 Mobile Application	2-1
2.2.3 Infrastructure	2-1
2.2.4 GIS Tools	2-1
2.3 Database Architecture	2-1
2.3.1 Database: earmm2019 (MySQL)	2-1
2.3.2 Structure:	2-1
2.3.3 Core Tables:	2-2
2.3.4 Data Integrity Issues:	2-2
2.4 Current Features	2-2
2.4.1 Public Portal (bmorodpwh.com)	2-2
2.4.2 Admin Portal (portal.bmorodpwh.com)	2-2
2.4.3 Mobile App (Broken)	2-3
2.5 What's Working	2-3
2.6 Critical Issues	2-3
2.6.1 Mobile App Failure	2-3
2.6.2 Technology Obsolescence	2-3
2.6.3 Data Fragmentation	2-3
2.6.4 No Data Integrity	2-3
2.6.5 Tampering Vulnerability	2-4
2.6.6 Storage Inefficiency	2-4
2.6.7 High Cloud Costs	2-4
2.6.8 Limited GIS Capabilities	2-4
2.6.9 No Automated Reporting	2-4
2.6.10 Missing Features	2-5
2.7 User Workflows	2-5

2.7.1	DEO Project Data Entry:	2-5
2.7.2	QGIS Processing (by GIS team):	2-5
2.7.3	Public Access:	2-5
2.8	Assessment Summary	2-5
2.8.1	Must Fix (High Priority):	2-5
2.8.2	Should Add (Medium Priority):	2-6
2.8.3	Can Improve (Lower Priority):	2-6
2.9	Existing System GIS Data Flow Diagram	2-6
CHAPTER 3	PROJECT OBJECTIVES	3-1
3.1	OBJECTIVE A: Integrated GIS Mapping	3-1
3.2	OBJECTIVE B: GIS Data Editing	3-1
3.3	OBJECTIVE C: GIS-Aided Progress Monitoring	3-2
3.4	OBJECTIVE D: Tamper-Proof Progress Reporting	3-3
3.5	OBJECTIVE E: Mobile Application	3-4
3.6	OBJECTIVE F: Web Application Enhancements	3-5
3.7	OBJECTIVE G: Server Migration/Optimization	3-6
3.8	Objectives Summary	3-7
CHAPTER 4	PROPOSED SOLUTION	4-1
4.1	Technology Stack	4-1
4.2	System Architecture	4-2
4.3	Database Design	4-2
4.3.1	Migration Strategy:	4-2
4.3.2	Key Schema Improvements:	4-2
4.4	How Solution Addresses Objectives	4-3
4.5	Key Features	4-4
4.5.1	Unified GIS Map	4-4
4.5.2	Web-Based GIS Editor	4-4
4.5.3	Geofencing System	4-4
4.5.4	Automated Report Generation	4-4
4.5.5	Mobile App (Android)	4-5
4.5.6	Alert System	4-5
4.6	Security Measures	4-5
4.6.1	Authentication	4-5
4.6.2	Authorization	4-5
4.6.3	Data Protection	4-5
4.6.4	Audit Trail	4-5
4.7	Performance Optimizations	4-6
4.7.1	Database	4-6
4.7.2	Application	4-6
4.7.3	Frontend	4-6
4.7.4	Media Delivery	4-6
4.8	Development Approach	4-6
4.8.1	Iterative Development	4-6
4.8.2	Version Control	4-6
4.8.3	Environments	4-6
4.8.4	Testing Strategy	4-6
4.9	Proposed System GIS Data Flow Diagram	4-7
CHAPTER 5	WORK PLAN & TIMELINE	5-1

5.1	Project Phases.....	5-1
5.2	Month-by-Month Schedule.....	5-1
5.2.1	Month 1: Inception.....	5-1
5.2.2	Month 2: Foundation Setup.....	5-1
5.2.3	Month 3: Database Migration & Backend.....	5-1
5.2.4	Month 4: GIS Development.....	5-1
5.2.5	Month 5: GIS Editing & Integration.....	5-2
5.2.6	Month 6: Progress Monitoring System.....	5-2
5.2.7	Month 7: Monitoring & Alerts & Tamper-Proof System.....	5-2
5.2.8	Month 8: Mobile App Development Start.....	5-2
5.2.9	Month 9: Mobile App & Web Enhancement.....	5-2
5.2.10	Month 10: Testing & UAT.....	5-3
5.2.11	Month 11: Server Optimization & Deployment Prep.....	5-3
5.2.12	Month 12: Production Deployment & Handover.....	5-3
5.3	Deliverables Timeline.....	5-3
5.4	Work Schedule.....	5-4
5.5	Critical Path.....	5-5
5.6	Resource Allocation.....	5-5
5.7	Milestones & Demo Sessions.....	5-5
5.8	Risk Buffers.....	5-6
5.8.1	Built-in Buffers:.....	5-6
5.8.2	Contingency Plans:.....	5-6
CHAPTER 6	BUDGET & RESOURCES.....	6-1
6.1	Payment Schedule.....	6-1
6.2	Cloud Infrastructure Costs.....	6-1
6.2.1	Current Costs (Baseline).....	6-1
6.2.2	Proposed Optimized Costs.....	6-1
6.2.3	Optimization Strategies.....	6-1
6.2.4	Budget for 12-Month Project.....	6-2
6.3	On-Premises Infrastructure Analysis.....	6-2
6.3.1	Hardware Requirements (Estimated).....	6-2
6.3.2	Ongoing Costs (Annual).....	6-2
6.3.3	Total Cost of Ownership (5-Year).....	6-2
6.3.4	Advantages/Disadvantages.....	6-2
6.4	Team Structure.....	6-3
6.5	Client Responsibilities.....	6-3
6.5.1	Access and Credentials.....	6-3
6.5.2	Infrastructure Support.....	6-3
6.5.3	Personnel Support.....	6-3
6.5.4	Timely Reviews.....	6-3
6.5.5	Decision Points.....	6-3
6.6	Development Tools & Licenses.....	6-4
6.6.1	Open Source (No Cost):.....	6-4
6.6.2	Cloud Services (Usage-Based):.....	6-4
6.6.3	Optional Paid Services:.....	6-4
6.7	Infrastructure Requirements.....	6-4
6.7.1	Development Environment:.....	6-4
6.7.2	Staging Environment:.....	6-4
6.7.3	Production Environment:.....	6-4

6.8	Training & Documentation	6-4
6.9	Budget Summary	6-5
6.10	Assumptions	6-5
CHAPTER 7	RISK MANAGEMENT	7-1
7.1	Risk Assessment Matrix	7-1
7.2	Top Risks & Mitigation	7-1
7.2.1	RISK 1: Data Migration Complexity	7-1
7.2.2	RISK 2: Legacy Data Quality Issues	7-1
7.2.3	RISK 3: Cloud Cost Overruns	7-2
7.2.4	RISK 4: Database Performance Issues	7-2
7.2.5	RISK 5: User Adoption Resistance	7-2
7.2.6	RISK 6: On-Premises Hardware Delays (if chosen)	7-2
7.2.7	RISK 7: Key Personnel Unavailability	7-2
7.2.8	RISK 8: Scope Creep	7-3
7.2.9	RISK 9: Internet Connectivity Issues (DEOs).....	7-3
7.2.10	RISK 10: Security Vulnerabilities	7-3
7.2.11	RISK 11: Third-Party API Failures	7-3
7.2.12	RISK 12: Mobile App Compatibility	7-4
7.3	Risk Monitoring	7-4
7.4	Escalation Procedures	7-4
CHAPTER 8	NEXT STEPS	8-1
8.1	Immediate Actions for Phase 2	8-1
8.1.1	Week 1-2: Access and Setup	8-1
8.1.2	Week 3-4: Database Migration Preparation	8-1
8.2	Key Decisions Required	8-1
8.2.1	Decision 2: Cloud vs. On-Premises (Month 3-4)	8-1
8.3	Deliverable Acceptance Criteria	8-2
8.3.1	Month 5: Integrated GIS Web Map	8-2
8.3.2	Month 7: Progress Monitoring System	8-2
8.3.3	Month 9: Updated Website & Mobile App	8-2
8.3.4	Month 12: Final Report & System	8-2
8.4	Communication Protocols	8-3
8.4.1	Regular Communication	8-3
8.4.2	Issue Escalation	8-3
8.5	Success Metrics	8-3
8.6	Phase 2 Kick-Off Checklist	8-4
8.7	Conclusion	8-4

EXECUTIVE SUMMARY

PROJECT OVERVIEW

MPW-BARMM has engaged Woodfields Consultants to modernize the E-BARMM infrastructure monitoring system, originally deployed as E-ARMM from 2016-2019 across five provinces (Basilan, Lanao del Sur, Maguindanao, Sulu, Tawi-Tawi) and eight District Engineering Offices (DEOs).

Critical Issues Requiring Update:

- Mobile app non-functional since 2019 (no field data collection)
- Outdated, insecure technology (PHP 5.3, MySQL 5.5 from 2010-2011)
- High cloud costs (\$613/month; ~₱34,300 PHP based on August 2025 actual bill)
- Data fragmentation and tampering risks
- No geofencing or photo verification

PROJECT OBJECTIVES (TOR SECTION III)

Objective	Description	Deliverable Month
A	Integrate GIS map (photos + road inventory)	5
B	Web-based GIS data editing	5
C	Progress monitoring (design plans, alerts)	7
D	Tamper-proof reporting system	7
E	Mobile application (Android only)	9
F	Web enhancement (UI, security, performance)	9
G	Server migration/optimization analysis	11

Deliverables & Payment Schedule: See Chapter 6.1 for complete payment schedule tied to deliverables.

PROPOSED SOLUTION

Technology Migration:

Component	Current (2016 – 2019)	Proposed (2025)
Backend	PHP 5.3 / MySQL 5.5	Python 3.9 / Django 4.0 or FastAPI / PostgreSQL 14+
CMS/Framework	Joomla 2.5 / Apache 2.2	Django 4.0 or FastAPI / Nginx + Unicorn
Spatial	Google Maps API	PostGIS 3.2 / GeoDjango / Leaflet.js
Frontend	PHP templates	React.js 18
Mobile	Ionic (broken)	Android (Kotlin/React Native)
Storage	EBS (expensive)	S3 + CloudFront

Key Improvements:

- Single unified GIS map linking all data (projects, photos, videos, roads)
- Geofencing validation for photo locations
- Automated PDF reports with QR codes and digital signatures
- Image/video compression

- Web-based GIS editing
- 24-41% cloud cost reduction (~P100,000-170,000 PHP/year savings)

PROJECT TIMELINE

Phase 1: Inception & Planning (Month 1)

Phase 2: Design & Development (Months 2-8)

Phase 3: Testing & Validation (Months 8-10)

Phase 4: Deployment & Training (Months 10-12)

Phase 5: Stakeholder Engagement (Throughout project)

SUCCESS CRITERIA

The project will be successful when:

- All 7 objectives delivered on schedule
- Mobile app functional with RouteShoot working
- Data unified in single spatial database
- Geofencing validates photo locations
- Cloud costs reduced by 24-41%
- UAT completed by MPW-BARMM and DEOs
- Training completed and documentation delivered

CHAPTER 1 INTRODUCTION

1.1 BACKGROUND

1.1.1 E-ARMM/E-BARMM System History

The E-ARMM (Expanded ARMM Road Mapping and Management) system was developed 2016-2019 to promote transparency in BARMM infrastructure projects using GIS technology.

Original System (2016-2019):

1. Web Platform

- Public: bmorodpwh.com
- Admin: portal.bmorodpwh.com
- Stack: PHP/MySQL/Apache/Joomla on AWS

2. Mobile App - "On the Road"

- Android (Ionic Framework)
- RouteShoot video capture
- Geotagged photo collection
- **Status: Broken since 2019**

3. GIS Workflow

- QGIS processing of GPS tracks
- Road condition classification
- KML/KMZ file generation

Successfully used by 8 DEOs to document thousands of projects, photos, and videos from 2016-2019.

1.1.2 Why Update is Needed

Technology Issues:

- PHP 5.3.10 and MySQL 5.5.22 from 2010-2011 (security risks, no support)
- Mobile app broken (no field data collection for 5+ years)
- No responsive design

Data Problems:

- Projects, photos, videos, and road data fragmented (not linked)
- No foreign key constraints (orphaned data risk)
- Multiple separate map views

Security Gaps:

- No geofencing (contractors can submit photos from wrong locations)
- No photo verification or tamper detection
- Manual screenshot-based reports (easily faked)

Cost Issues:

- High AWS costs (\$613/month; ~₱34,300 PHP based on August 2025 actual bill)
- No storage optimization (images/videos uncompressed)

- See Chapter 6.2 for detailed cost analysis

1.2 PURPOSE OF THIS REPORT

This Inception Report is the first deliverable (Month 1, 15% payment) under TOR-WCI-25-035. According to the TOR:

*"It is the Inception Report that **shall dictate the course of the whole project until completion.**"*

This report establishes:

- Assessment of current E-ARMM system
- Detailed requirements for E-BARMM solution
- 12-month work plan with milestones
- Risk management strategies
- Budget and resource requirements

Based on:

- TOR review
- Existing system documentation (2018 reports)
- Database analysis (earmm2019 schema)
- Kickoff meeting (September 25, 2025)
- Cloud infrastructure assessment

1.3 PROJECT STAKEHOLDERS

MPW-BARMM

- Project owner and decision authority
- Project Chairman
- Technical Working Group
- System Administrator

District Engineering Offices (8 DEOs)

- Primary users and data contributors
- Province codes: bas, lds, mag, sul, taw
- Participate in UAT and training

Woodfields Consultants Team

- 3-person technical team (Team Leader/PM, Software Developer, GIS Specialist)
- See Chapter 6.4 for detailed roles and responsibilities

Other Stakeholders

- Contractors (use automated reports)
- General public (access transparency data)
- BARMM government agencies

1.4 REPORT STRUCTURE

Chapter	Contents
1	Introduction - Background, purpose, stakeholders
2	Current System Assessment - Architecture, database, issues
3	Project Objectives - Seven objectives (A-G) from TOR
4	Proposed Solution - Technology stack, architecture, features
5	Work Plan & Timeline - 12-month schedule and phases
6	Budget & Resources - Costs, team, payment schedule
7	Risk Management - Risks and mitigation strategies
8	Next Steps - Review process and Phase 2 kickoff

CHAPTER 2 CURRENT SYSTEM ASSESSMENT

2.1 SYSTEM OVERVIEW

The E-ARMM system consists of web portals, mobile app, and GIS workflow serving 8 DEOs across 5 provinces.

2.1.1 Websites:

- **Public:** bmorodpwh.com (transparency and public access)
- **Admin:** portal.bmorodpwh.com (DEO data entry and management)

2.1.2 Coverage:

- 8 District Engineering Offices
- 5 provinces
- Thousands of projects from 2012-2019
- Geotagged photos, RouteShoot videos, drone footage

2.2 TECHNOLOGY STACK

2.2.1 Web Application

- Backend: PHP 5.3.10 (2010) + MySQL 5.5.22 (2011)
- Web Server: Apache 2.2.22 + Joomla 2.5
- GIS: GIS Cloud API, Google Maps

2.2.2 Mobile Application

- Name: "On the Road"
- Platform: Android (Ionic Framework)
- Versions: Public, Admin, Offline
- **Status: Non-functional since 2019**

2.2.3 Infrastructure

- Hosting: AWS (EC2, VPC, AWS WAF, AWS Support)
- OS: Ubuntu 12.04
- Cost: \$613/month (~₱34,300 PHP) - August 2025 actual bill

2.2.4 GIS Tools

- Desktop: QGIS
- Formats: KML/KMZ, GeoJSON, Shapefile

2.3 DATABASE ARCHITECTURE

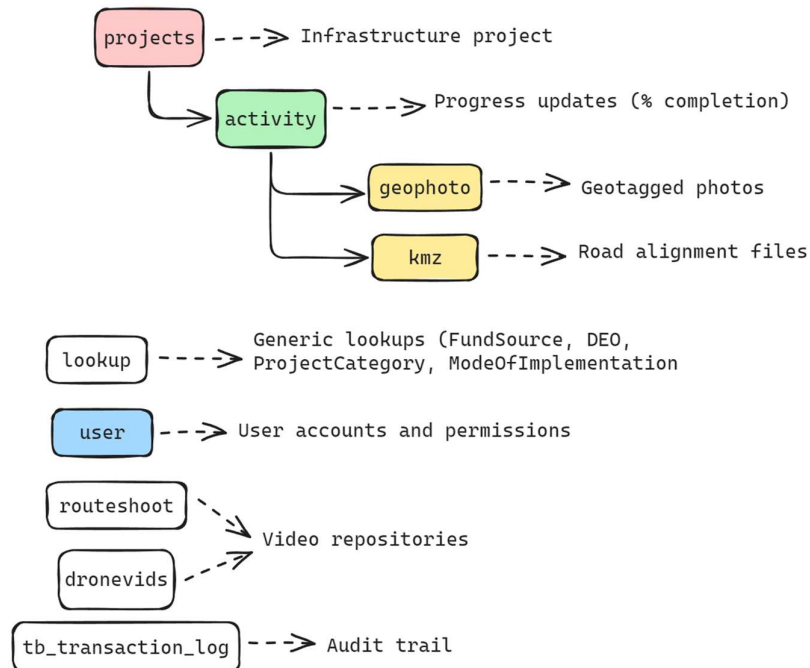
2.3.1 Database: earmm2019 (MySQL)

2.3.2 Structure:

- 18 tables
- 15 views

- **No foreign key constraints** (application-enforced only)

2.3.3 Core Tables:



- vw_projects - Projects with lookups and accomplishment
- vw_photos - All photos with project context
- v_activity_list - Projects with latest activity

2.3.4 Data Integrity Issues:

- No CASCADE operations (orphaned records possible)
- String-based foreign keys (case-sensitive matching)
- No referential integrity enforcement

2.4 CURRENT FEATURES

2.4.1 Public Portal (bmorodpwh.com)

- Dashboard with project statistics
- Project listing by fund source and mode
- RouteShoot video viewer with map overlay
- Drone video viewer
- Separate GIS maps: one for geotagged photos, another for road networks (not integrated)

2.4.2 Admin Portal (portal.bmorodpwh.com)

- All public features plus:
- Project data entry and editing
- Activity management
- Geotagged photo upload (drag-and-drop)

- User management
- Transaction logging

2.4.3 Mobile App (Broken)

- RouteShoot video capture
- Geotagged photo collection
- Offline data collection
- Last working: 2019

2.5 WHAT'S WORKING

1. **Active User Base** - 8 DEOs familiar with workflows
2. **Historical Data** - Comprehensive 2012-2019 repository intact
3. **Working Portals** - Public and admin websites functional
4. **Audit Trail** - Transaction logging captures all changes
5. **Cloud Infrastructure** - AWS already operational
6. **Established Workflows** - DEOs know data entry processes

2.6 CRITICAL ISSUES

2.6.1 Mobile App Failure

Impact: No field data collection for 5+ years

- "On the Road" app broken since 2019
- No RouteShoot video capture
- Cannot publish updates (outdated Ionic Framework)

2.6.2 Technology Obsolescence

Impact: Security risks, cannot use modern libraries

- PHP 5.3.10 from 2010 (vulnerabilities, no support)
- MySQL 5.5.22 from 2011 (lacks modern features)
- Ubuntu 12.04 (end-of-life)
- Joomla 2.5 (deprecated)

2.6.3 Data Fragmentation

Impact: Cannot perform holistic analysis

- Projects, photos, RouteShoot, drone videos stored separately
- Geotagged photos and road networks displayed on separate, unintegrated maps
- No unified spatial database
- Manual linking required

2.6.4 No Data Integrity

Impact: Data inconsistencies, cleanup challenges

- Zero foreign key constraints
- Application-enforced relationships only
- Orphaned records possible
- String-based lookups (case-sensitive)

2.6.5 Tampering Vulnerability

Impact: Risk of fraudulent progress reporting

- No geofencing to validate photo locations
- No photo hash verification
- EXIF metadata easily altered
- Manual screenshot-based contractor reports

2.6.6 Storage Inefficiency

Impact: High cloud expenses

- EBS storage instead of S3 (3-5x more expensive)
- Uncompressed photos and videos
- No CDN for media delivery

2.6.7 High Cloud Costs

Impact: Budget strain (~₱412,000 PHP/year)

- Current cost: \$613/month (~₱34,300 PHP at ₱56/\$1)
- Annual cost: ~\$7,357 (~₱412,000 PHP)
- Potential savings: \$150-250/month (24-41% reduction) through optimization
- See Chapter 6.2 for detailed breakdown and optimization strategies

2.6.8 Limited GIS Capabilities

Impact: Requires GIS expertise, slow workflows

- No web-based GIS editing
- Desktop QGIS required for all spatial work
- No CRS management
- Cannot overlay georeferenced design plans

2.6.9 No Automated Reporting

Impact: Low trust, manual verification required, fraudulent reporting risk

- **Current practice:** Contractors manually screenshot geotagged photo pages for progress reports
- Easy to fake, alter, or manipulate screenshots
- No automated PDF generation from database
- Static PDF upload feature exists but not used for progress reporting
- No QR codes, watermarks, or digital signatures for verification
- No standardized tamper-proof format

2.6.10 Missing Features

- No alert system for data changes
- Drone videos not georeferenced
- No real-time collaboration
- No mobile-responsive design
- No API for third-party integration

2.7 USER WORKFLOWS

2.7.1 DEO Project Data Entry:

1. Login to portal.bmorodpwh.com
2. Add/edit project details
3. Add activity with % accomplishment
4. Upload geotagged photos (drag-and-drop)
5. Submit for public display

2.7.2 QGIS Processing (by GIS team):

1. Download RouteShoot GPS tracks
2. Import to QGIS Desktop
3. Classify road segments (condition, year, type)
4. Export as shapefile/GeoJSON
5. Upload to web portal
6. Manually link to projects

2.7.3 Public Access:

1. Visit bmorodpwh.com
2. Browse projects by DEO, fund source, or mode
3. View project details and photos on map
4. Watch RouteShoot/drone videos
5. Download pre-uploaded PDF documentation files

2.8 ASSESSMENT SUMMARY

Verdict: The E-ARMM system successfully established GIS-based infrastructure monitoring from 2016-2019 but now suffers from critical failures requiring immediate modernization.

2.8.1 Must Fix (High Priority):

- Mobile app restoration
- Data unification and linking
- Geofencing and tamper-proofing
- Cloud cost optimization
- Technology security upgrade

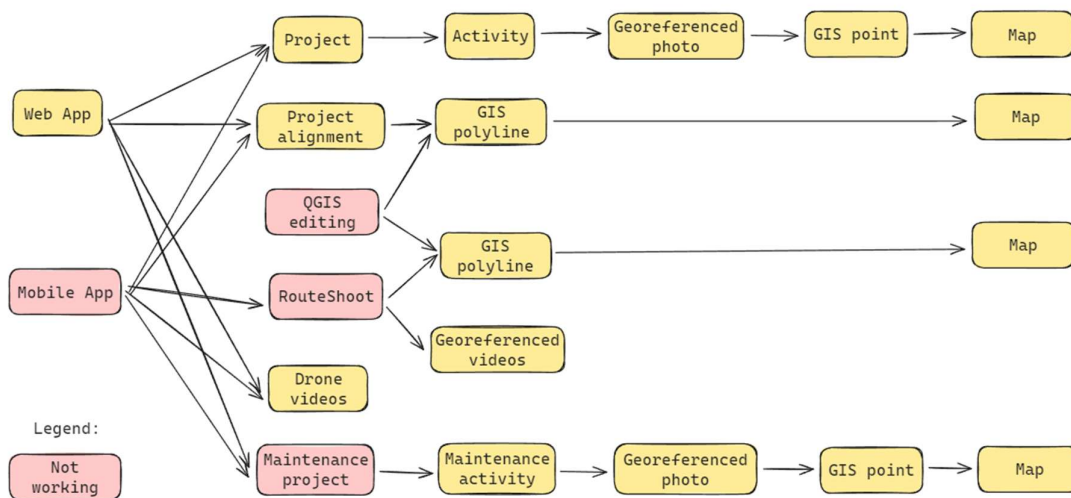
2.8.2 Should Add (Medium Priority):

- Web-based GIS editing
- Automated report generation
- Georeferenced design plan overlays
- Alert system
- API for integrations

2.8.3 Can Improve (Lower Priority):

- User interface modernization
- Performance optimization
- Better documentation
- Enhanced training materials

2.9 EXISTING SYSTEM GIS DATA FLOW DIAGRAM



CHAPTER 3 PROJECT OBJECTIVES

This chapter details the seven objectives (A-G) from TOR-WCI-25-035 Section III (Scope of Work).

3.1 OBJECTIVE A: INTEGRATED GIS MAPPING

Goal: Consolidate geotagged photos and road inventory into a single unified GIS map.

Current Problem: Photos stored per-project (fragmented), separate map views, cannot see full infrastructure picture

Requirements:

1. Single interactive map showing:

- Road inventory dataset (all roads in BARMM)
- Geotagged construction progress photos
- Project locations and boundaries
- RouteShoot tracks
- Drone video coverage areas

2. Retain existing features:

- Layer controls (toggle on/off)
- Zoom/pan, info popups, search/filter

3. New features:

- Photo clustering for dense areas
- Timeline slider (filter by year)
- Unified legend
- Export map as image/PDF

Deliverable: Integrated GIS web map (Month 5, 25% payment)

3.2 OBJECTIVE B: GIS DATA EDITING

Goal: Enable web-based editing and modification of GIS data.

Current Problem: All spatial editing requires desktop QGIS, slow turnaround, limited to GIS specialists

Requirements:

1. Database upgrade:

- Migrate to PostgreSQL/PostGIS
- Support spatial data types (geometry, geography)
- Enable spatial indexing

2. Web editing interface:

- Add/edit/delete points, lines, polygons
- Drawing tools with snap-to features
- Attribute modification

3. Data validation:

- Required field checks
- Coordinate validation
- Duplicate detection
- Geometry validation

4. Access controls:

- Role-based permissions (admin, editor, viewer)
- DEO-specific data access
- Change tracking and version history

Deliverable: GIS web map with editing (Month 5, combined with Objective A)

3.3 OBJECTIVE C: GIS-AIDED PROGRESS MONITORING

Goal: Comprehensive progress monitoring using georeferenced design plans, geotagged photos, drone videos, and alerts.

Current Problem: No design plan integration, photos not validated, drone videos not georeferenced, manual monitoring

Requirements:

1. Upload georeferenced design plans:

- Accept raster formats (GeoTIFF, PNG with world file, JPEG with world file)
- PDF inputs will be rasterized and georeferenced
- Display as overlay on map
- Align with satellite imagery

2. Photo-over-plan overlay:

- Compare planned vs. actual
- Measure distances/areas

3. CRS handling:

- Support WGS84 (EPSG:4326), PRS92, UTM zones
- On-the-fly reprojection
- CRS conflict warnings

4. Project-specific GIS features:

- Create project boundary polygons
- Update % accomplishment spatially
- Link to budget/cost data

5. Drone video integration:

- Upload with GPS tracks
- Georeference video frames

- Display flight path on map

6. Georeferenced videos (RouteShoot):

- GPS-synchronized video playback
- Use for final billing verification

7. Alert system:

- Email/SMS notifications on data changes
- Alerts for new uploads and critical edits
- Configurable per user/DEO

Deliverable: GIS-aided progress monitoring system (Month 7, 25% payment)

3.4 OBJECTIVE D: TAMPER-PROOF PROGRESS REPORTING

Goal: Prevent contractors from altering or faking progress reports.

Current Problem: Contractors screenshot pages (easily faked), photos can be from wrong locations, no verification

Requirements:

1. Industry standards research:

- Watermarking, barcode/QR codes, digital signatures, blockchain

2. Geofencing implementation:

- Define project boundary polygons
- Validate uploaded photo GPS coordinates
- Reject photos outside boundary (with tolerance buffer)
- Flag suspicious locations for review

3. Photo integrity verification:

- Calculate cryptographic hash (SHA-256) on upload
- Verify hash on access (detect tampering)
- Check EXIF metadata consistency

4. QR code authentication:

- Generate unique QR code per report
- Link to immutable database record
- Scannable for instant verification

5. Digital watermarks:

- Visible watermark on report PDFs
- Invisible watermark (steganography) on photos

6. Digital signatures:

- PKI-based signing of reports
- Timestamp authority for non-repudiation

7. Automated PDF report generation:

- Replace manual screenshots
- System-generated reports from database
- Includes QR code, watermark, digital signature
- Export/print/email functionality

Deliverable: Tamper-proof progress reporting (Month 7)

3.5 OBJECTIVE E: MOBILE APPLICATION

Goal: Develop E-BARMM mobile application for Android.

Current Problem: "On the Road" app broken since 2019, no field data collection

Requirements:

1. **Platform:** Android only (iOS not required)

2. **Android app minimum:**

- Target Android 8.0+ (API level 26+)
- Publish to Google Play Store

3. **Priority features (must have):**

RouteShoot integration:

- GPS-synchronized video recording
- Real-time GPS track overlay
- Pause/resume, save video + GPS track (KML)
- Upload to server

Secure geotagging:

- Camera integration
- Embed GPS coordinates in EXIF
- Tamper-proof metadata with hash verification
- Timestamp and device ID

Offline mode:

- Download project list for offline access
- Collect photos/videos offline
- Sync when internet available
- Queue uploads

Project assignment:

- Assign users to specific projects
- Filter by DEO
- Track progress offline

4. **Ideal functionality:**

- All web app features (if feasible)
- Project listing and details

- Photo gallery viewer
- Map viewer

Deliverable: E-BARMM mobile app (Month 9, combined with Objective F)

3.6 OBJECTIVE F: WEB APPLICATION ENHANCEMENTS

Goal: Modernize and improve the E-BARMM web application.

Current Problem: Outdated UI, security vulnerabilities, poor performance, no responsive design

Requirements:

1. User-friendly interface:

- Modern, clean design with intuitive navigation
- Accessibility (WCAG 2.1)

2. Security enhancements:

- HTTPS everywhere (SSL/TLS)
- CSRF, SQL injection, XSS protection
- Secure password hashing (bcrypt)
- Rate limiting (prevent brute force)

3. Testing and debugging:

- Unit tests (backend)
- Integration tests (API)
- End-to-end tests (user workflows)
- Performance testing (load testing)
- Security audit

4. Training and support:

- User manuals (web and mobile)
- Video tutorials
- Training sessions for DEOs
- Admin documentation

5. Responsive design:

- Mobile-friendly (phones, tablets)
- Adaptive layouts, touch-optimized
- Progressive Web App (PWA) features

6. Performance optimization:

- Page load < 3 seconds
- API response < 500ms
- Image lazy loading, code minification
- CDN for static assets, caching (Redis)

7. Analytics and reporting:

- Dashboard with key metrics
- Project cost summaries, accomplishment charts
- Export to Excel/PDF

8. Automated project reports:

- Generate PDF reports from database
- Include QR code, watermark, signature
- Replace contractor screenshot practice

Deliverable: Updated E-BARMM website (Month 9, 25% payment)

3.7 OBJECTIVE G: SERVER MIGRATION/OPTIMIZATION

Goal: Analyze cloud vs. on-premises hosting and optimize infrastructure.

Current Problem: High AWS costs (\$613/month), inefficient storage, no optimization

Requirements:

1. Maintain cloud during project:

- Keep AWS operational for 12 months
- No service interruption

2. On-premises feasibility study:

- Hardware requirements, network infrastructure
- Physical security, maintenance costs
- Backup/disaster recovery, scalability
- 5-year TCO comparison

3. Inception discussion (Month 3):

- Present cloud vs. on-premises analysis
- Advantages, disadvantages, cost comparison
- Risk assessment and recommendation

4. If on-premises chosen:

- Procure, install, configure hardware
- Migrate from cloud, test and document

5. If cloud chosen:

- Use saved budget for future cloud subscription
- Extend AWS reserved instances

6. Cloud optimization (regardless):

EBS to S3 migration:

- Move media to S3 Standard/Intelligent-Tiering
- Up to 70% storage cost reduction

Image/video compression:

- Convert to WebP format and H.265 (HEVC)

- 50-60% size reduction

RDS Reserved Instances:

- Purchase 1-year or 3-year RI
- Up to 40% discount

Right-sizing EC2:

- Downsize over-provisioned instances
- Use burstable instances (t3)

CloudFront CDN:

- Reduce data transfer costs
- Improve global access speed

Auto-scaling:

- Scale down during low usage

Cost monitoring:

- Setup AWS Budget alerts
- Monthly cost review

Target: Reduce cloud costs by 24-41% (~₱100,000-170,000 PHP/year savings based on August 2025 actual costs)

Deliverable: Migrated server or optimized cloud setup (Month 11)

3.8 OBJECTIVES SUMMARY

Objective	Focus	Month	Payment
A	Integrated GIS Map	5	25%
B	GIS Editing	5	(with A)
C	Progress Monitoring	7	25%
D	Tamper-Proof Reports	7	(with C)
E	Mobile App	9	25%
F	Web Enhancement	9	(with E)
G	Server Optimization	11	(with Final)

CHAPTER 4 PROPOSED SOLUTION

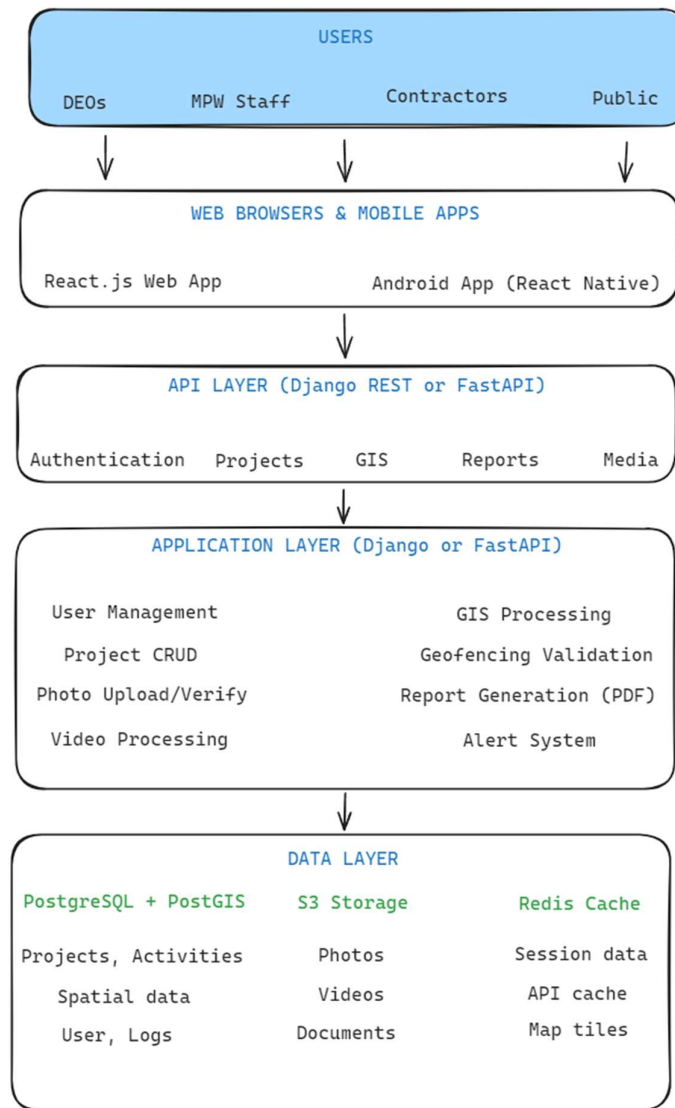
4.1 TECHNOLOGY STACK

Component	Technology	Rationale
Backend	Python 3.9 / Django 4.0 or FastAPI	Modern, secure, excellent GIS support (GeoDjango)
Web Server	Nginx + Gunicorn (Django) or Uvicorn (FastAPI)	High performance, production-ready
API	Django REST Framework / fastAPI	Standards-based, auto-documentation
Task Queue	Celery	Async tasks (video processing, emails)
Frontend	React.js 18	Fast, component-based, large ecosystem
UI Library	Material-UI or Ant Design	Professional, accessible components
Maps	Leaflet.js	Open source, feature-rich
State Management	Redux or Context API	Predictable state updates
Database	PostgreSQL 14+	Industry standard, reliable
Spatial Extension	PostGIS 3.2	Best-in-class spatial database
Mobile	Android (Kotlin or React Native)	Kotlin = native performance, React Native = faster dev
Mobile Maps	Google Maps SDK or Mapbox	Native integration
Offline Storage	SQLite	Local data persistence
GIS Tools	QGIS, GeoDjango, GDAL/OGR	Spatial processing and conversions
Cloud	AWS (EC2, S3, RDS, CloudFront)	Already in use, familiar
Monitoring	CloudWatch, Sentry	Error tracking and metrics

Implementation Flexibility:

The technologies and approaches outlined in this report represent our recommended solution based on current assessment and industry best practices. However, if during development any component or approach is found to be infeasible, incompatible with existing systems, or unsuitable for project requirements, alternative solutions will be evaluated and discussed with MPW-BARMM. Such changes will be documented through the project's change control process and communicated in monthly progress reports.

4.2 SYSTEM ARCHITECTURE



4.3 DATABASE DESIGN

4.3.1 Migration Strategy:

1. Export earmm2019 MySQL to SQL dump
2. Convert schema to PostgreSQL syntax
3. Add foreign key constraints
4. Convert spatial data (lat/lon → PostGIS geometry)
5. Import with validation
6. Create spatial indexes

4.3.2 Key Schema Improvements:

1. Current vs. New Database Structure

Feature	Current (MySQL)	New (PostgreSQL + PostGIS)
Foreign Keys	None (application-enforced)	All relationships enforced with CASCADE
Spatial Data	Separate lon/lat columns (numbers)	Native GEOMETRY type (Point, Polygon)
Spatial Indexes	Regular indexes only	GIST indexes for fast spatial queries
Data Integrity	Manual validation	Database-level constraints
Geofencing	Not supported	Project boundary polygons
Tamper Detection	None	SHA-256 file hashes stored
S3 Integration	Local paths only	S3 object keys stored
Audit Trail	Manual logging	Automatic created_by/updated_by tracking

2. Enhanced Tables

PROJECTS Table

Old: title, deo, location (text), lat, long, fund_source (string)
 New: + location_geom (Point), + project_boundary (Polygon),
 + fund_source_id (FK), + contractor_id (FK),
 + created_at, + created_by (FK to users)

GEOPHOTO Table

Old: file_name, activity_id, lon, lat, img_date
 New: + location (Point geometry), + file_hash (SHA -256),
 + s3_key, + within_boundary (boolean),
 + uploaded_at, + uploaded_by (FK to users)

3. New Tables Added

Table	Purpose
<i>design_plans</i>	Store georeferenced design overlays (raster images: GeoTIFF, PNG, JPEG)
<i>alerts</i>	Track alert system notifications
<i>report_generations</i>	Log generated PDFs with QR codes
<i>geofence_validations</i>	Record all geofencing checks (pass/fail)
<i>api_tokens</i>	Manage mobile app authentication tokens

4.4 HOW SOLUTION ADDRESSES OBJECTIVES

Objective	Implementation Approach
A: Integrated GIS Map	Single Leaflet map with PostGIS spatial queries, layer toggles, clustering
B: GIS Editing	Leaflet.draw plugin + API (Django or FastAPI) + PostGIS validation
C: Progress Monitoring	Upload georeferenced rasters (GeoTIFF/PNG/JPEG), PDF-to-raster conversion, overlay photos using spatial queries, Celery alert system
D: Tamper-Proof Reports	ST_Within() for geofencing, SHA-256 hashing, Python qrcode library, ReportLab PDFs

Objective	Implementation Approach
E: Mobile App	Android Kotlin/React Native, camera + GPS EXIF, SQLite offline sync
F: Web Enhancements	React responsive components, framework security middleware, Redis caching
G: Server Optimization	boto3 S3 migration, Pillow/FFmpeg compression, CloudWatch monitoring

4.5 KEY FEATURES

4.5.1 Unified GIS Map

- Base layers: OpenStreetMap, Satellite
- Data layers: Projects, photos, roads, videos (toggle on/off)
- Clustering: Group nearby photos
- Popups: Click for details
- Search: Find by name, DEO, year
- Export: Download as image or GeoJSON

4.5.2 Web-Based GIS Editor

- Draw tools: Point, line, polygon, rectangle
- Edit vertices: Drag to reshape
- Attribute editor: Form fields
- Validation: Real-time checks
- Save: POST to API → PostGIS

4.5.3 Geofencing System

Geofencing ensures photos are taken at the actual project site by validating GPS coordinates against project boundaries

- **How It Works:**

Geofencing ensures photos are taken at the actual project site by validating GPS coordinates against project boundaries.
- **Benefits:**
 - Prevents contractors from submitting photos taken elsewhere
 - Reduces fraudulent progress reporting
 - Provides audit trail of all validation attempts
 - Allows reasonable tolerance (100m buffer) for GPS accuracy issues
- **Technical Implementation:**
 - PostGIS spatial function `ST_Within()` checks if point is inside polygon
 - PostGIS function `ST_DWithin()` checks distance for near-misses
 - All validation results logged in `geofence_validations` table
 - Failed validations trigger alerts to project managers

4.5.4 Automated Report Generation

- Template: Django template with project data
- PDF: ReportLab (for complex layouts and signatures)
- QR code: Link to public verification page

- Watermark: Project name, date, "Official MPW-BARMM"
- Digital signature: Sign PDF with private key
- Delivery: Email to contractor, download link

4.5.5 Mobile App (Android)

- RouteShoot: GPS-synchronized video recording + KML track generation
- Geotagged photos: Camera integration with EXIF metadata and SHA-256 hashing
- Geofencing validation: Check photo/video location against project boundaries
- Offline mode: SQLite storage, queue uploads, auto-sync when connected
- Project assignment: View DEO-specific projects, map view, details
- Authentication: JWT tokens, role-based access
- Target: Android 8.0+ (API 26+), Google Play Store

4.5.6 Alert System

- Triggers: New photo upload, project edit, % accomplishment change
- Recipients: Configurable per DEO/project
- Delivery: Email (SendGrid) and/or SMS (Twilio)
- Queue: Celery async tasks
- Log: All alerts recorded

4.6 SECURITY MEASURES

4.6.1 Authentication

- JWT tokens for API (stateless)
- Django session auth for web
- Password: bcrypt hashing

4.6.2 Authorization

- Role-based: Superadmin, Admin, Editor, Viewer
- DEO-based: Users only see/edit their DEO data
- Object-level: Check permissions per project/photo

4.6.3 Data Protection

- HTTPS only (redirect HTTP → HTTPS)
- CSRF tokens on forms
- SQL injection: Django ORM (parameterized queries)
- XSS: React auto-escaping
- File upload: Validate types, scan for malware

4.6.4 Audit Trail

- Log all CRUD operations
- Capture: user, timestamp, IP, before/after values
- Store in `audit_log` table

4.7 PERFORMANCE OPTIMIZATIONS

4.7.1 Database

- Indexes on foreign keys, spatial columns
- Connection pooling (pgBouncer)
- Read replicas for reporting

4.7.2 Application

- Redis cache for API responses
- Celery for async tasks (video processing)
- Pagination (100 records per page)

4.7.3 Frontend

- Code splitting (lazy load routes)
- Image lazy loading
- Minification (Webpack/React build tools)

4.7.4 Media Delivery

- S3 for storage
- CloudFront CDN (fast global access)
- Image thumbnails (generate on upload)

4.8 DEVELOPMENT APPROACH

4.8.1 Iterative Development

- Work organized in 2-week cycles
- Regular internal reviews and adjustments
- Monthly progress reporting to client

4.8.2 Version Control

- Git repository (GitHub/GitLab)
- Branching: main, develop, feature/*
- Code review process for quality assurance
- Automated deployment pipelines (CI/CD)

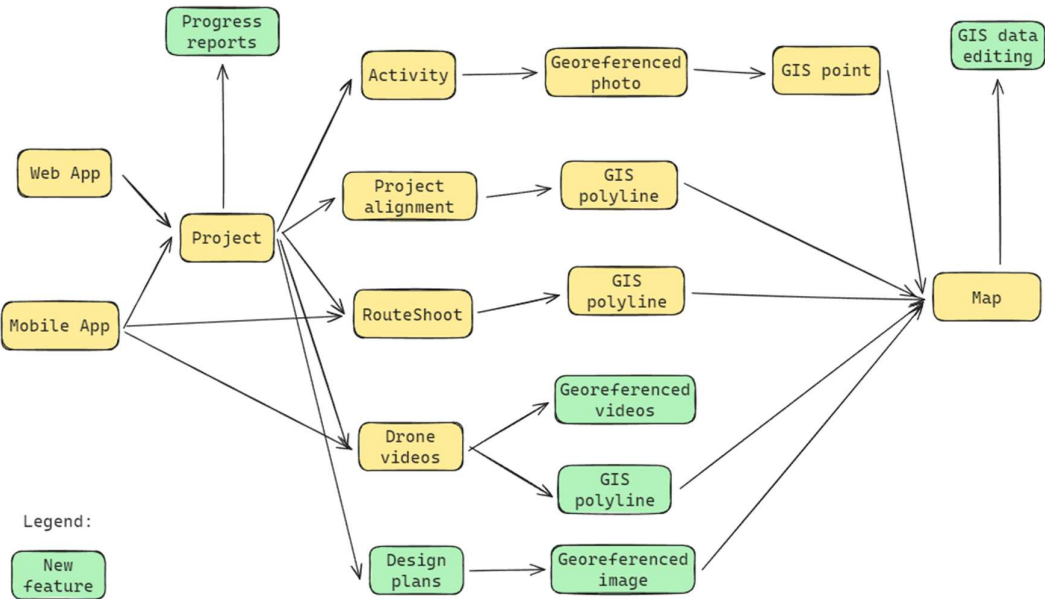
4.8.3 Environments

- Development: Local machines
- Staging: AWS (mirror of production)
- Production: AWS (live system)

4.8.4 Testing Strategy

- Unit tests: Django (pytest), React (Jest)
- Integration tests: API endpoints
- E2E tests: Cypress (user workflows)
- Manual UAT: DEOs test in staging

4.9 PROPOSED SYSTEM GIS DATA FLOW DIAGRAM



CHAPTER 5 WORK PLAN & TIMELINE

5.1 PROJECT PHASES

Phase	Duration	Focus
1: Inception & Planning	Month 1	Assessment, requirements, Inception Report
2: Design & Development	Months 2-8	Database migration, backend/frontend/mobile development
3: Testing & Validation	Months 8-10	Unit/integration/E2E testing, UAT, security audit
4: Deployment & Training	Months 10-12	Production deployment, data migration, training
5: Stakeholder Engagement	Throughout	Monthly reports, monthly meetings, demos

5.2 MONTH-BY-MONTH SCHEDULE

5.2.1 Month 1: Inception

- ☒ System assessment
- ☒ Database schema analysis
- ☒ Kickoff meeting
- ☒ Inception Report

Deliverable: Inception Report → 15% payment

5.2.2 Month 2: Foundation Setup

- ☐ Get system access (AWS, database, servers)
- ☐ Setup development environment
- ☐ Create PostgreSQL/PostGIS schema
- ☐ Write MySQL → PostgreSQL migration scripts
- ☐ Setup Django project
- ☐ Begin S3 migration and image compression

5.2.3 Month 3: Database Migration & Backend

- ☐ Run database migration
- ☐ Validate migrated data
- ☐ Create Django models
- ☐ Build REST API (projects, activities, photos)
- ☐ Implement JWT authentication
- ☐ Setup Redis caching

Key Decision: Cloud vs. On-Premises Meeting

5.2.4 Month 4: GIS Development

- ☐ Implement PostGIS spatial queries
- ☐ Build map API endpoints
- ☐ Integrate Leaflet.js frontend
- ☐ Develop unified map view (Objective A)

- ☐ Photo clustering algorithm

5.2.5 Month 5: GIS Editing & Integration

- ☐ Build GIS editing interface (Objective B)
- ☐ Implement draw/edit tools (Leaflet.draw)
- ☐ Create validation rules
- ☐ Role-based permissions
- ☐ Testing and refinement

Deliverable: Integrated GIS Web Map (A & B) → 25% payment

5.2.6 Month 6: Progress Monitoring System

- ☐ Design plan upload (raster formats: GeoTIFF, PNG, JPEG)
- ☐ PDF-to-raster conversion and georeferencing
- ☐ Photo-over-plan overlay
- ☐ CRS conversion handling
- ☐ Project boundary creation
- ☐ Geofencing validation logic (Objective D)

5.2.7 Month 7: Monitoring & Alerts & Tamper-Proof System

- ☐ Drone video upload and georeferencing
- ☐ RouteShoot video integration
- ☐ Alert system (Celery + email/SMS)
- ☐ Photo hashing implementation
- ☐ QR code generation
- ☐ PDF report generator with watermarks
- ☐ Digital signature integration
- ☐ Automated report templates
- ☐ Testing and refinement

Deliverable: Progress Monitoring System (C & D) → 25% payment

5.2.8 Month 8: Mobile App Development Start

- ☐ Mobile app project setup (Android)
- ☐ Mobile API integration
- ☐ Begin camera and geotagging features
- ☐ RouteShoot planning and design

5.2.9 Month 9: Mobile App & Web Enhancement

- ☐ Mobile app development (camera, geotagging)
- ☐ RouteShoot implementation (video + GPS)
- ☐ Offline mode and sync
- ☐ React UI/UX redesign
- ☐ Security hardening
- ☐ Performance optimization
- ☐ Testing

Deliverable: Updated E-BARMM Website (E & F) → 25% payment

5.2.10 Month 10: Testing & UAT

- ☐ Comprehensive testing (unit, integration, E2E)
- ☐ User Acceptance Testing with DEOs
- ☐ Security audit and penetration testing
- ☐ Performance/load testing
- ☐ Bug fixing
- ☐ Mobile app refinement

5.2.11 Month 11: Server Optimization & Deployment Prep

- ☐ Complete cloud cost optimizations (S3, compression, RDS RI)
- ☐ OR on-premises setup (if chosen)
- ☐ Staging environment deployment
- ☐ Data migration dry run
- ☐ Prepare production deployment plan
- ☐ Draft Final Report

Milestone: Server Optimization (G) complete

5.2.12 Month 12: Production Deployment & Handover

- ☐ Production deployment
- ☐ Data migration (old → new system)
- ☐ DNS cutover
- ☐ Post-deployment monitoring
- ☐ User training sessions (DEOs, MPW staff)
- ☐ Documentation handover
- ☐ Final Report

Deliverable: Final Report → 10% payment

5.3 DELIVERABLES TIMELINE

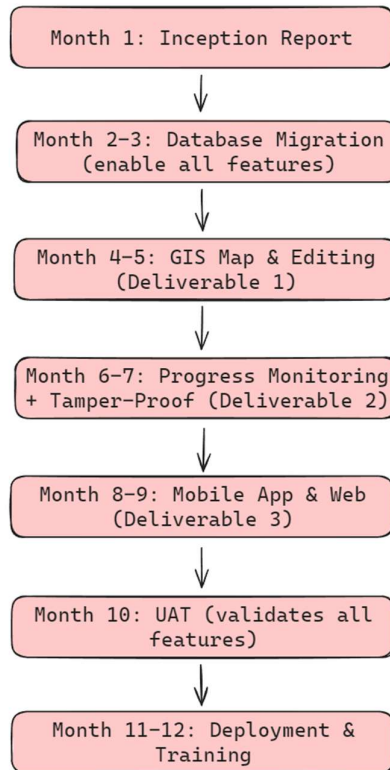
Month	Deliverable	Objectives	Payment
1	Inception Report	-	15%
5	Integrated GIS Web Map	A, B	25%
7	Progress Monitoring System	C, D	25%
9	Updated E-BARMM Website	E, F	25%
11	Draft Final Report	-	-
12	Final Report	G	10%

Monthly: Progress Reports

NO.	ACTIVITIES	MONTH 1		MONTH 2				MONTH 3				MONTH 4				MONTH 5				MONTH 6				MONTH 7				MONTH 8				MONTH 9				MONTH 10				MONTH 11				MONTH 12																	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48												
1.0	Inception and planning																																																												
	Inception																																																												
2.0	Design and development																																																												
	Foundation setup																																																												
	Database migration and backend																																																												
	GIS development																																																												
	GIS editing and integration																																																												
	Progress monitoring system																																																												
	Monitoring, alerts, & tamper-proof system																																																												
	Mobile app development start																																																												
	Mobile app and web enhancement																																																												
3.0	Testing and validation																																																												
	Testing and UAT																																																												
4.0	Deployment and training																																																												
	Server optimization & deployment prep																																																												
	Production deployment & handover																																																												
5.0	Stakeholder engagement																																																												
4.2.3	Knowledge transfer and training																																																												

[illegible]

5.5 CRITICAL PATH



Critical Dependencies:

- Database migration must complete by Month 3
- GIS map must work before adding monitoring features
- Mobile app needs API ready (Month 8)
- UAT must pass before production deployment

5.6 RESOURCE ALLOCATION

Team Member	Months 1-3	Months 4-7	Months 8-10	Months 11-12
Project Manager	Architecture, infrastructure	Project management, coordination	Deployment planning	Training, handover
Software Developer	Backend (Django, API, DB)	Frontend (React, mobile)	Testing	Deployment support
GIS Specialist	PostGIS, spatial queries	GIS features, georeferencing	QGIS workflows	Training, docs

5.7 MILESTONES & DEMO SESSIONS

- **Month 1:** Kickoff meeting
- **Month 3:** Cloud vs. on-premises decision meeting
- **Month 5:** Demo 1 - Integrated GIS map with editing

- **Month 7:** Demo 2 - Progress monitoring with geofencing
- **Month 9:** Demo 3 - Mobile app and updated website
- **Month 10:** UAT sessions with DEOs
- **Month 12:** Final demonstration and handover

5.8 RISK BUFFERS

5.8.1 Built-in Buffers:

- Testing phase (Months 8-10) allows 2 months for fixes
- Overlap between development and testing
- Month 12 reserved for production issues

5.8.2 Contingency Plans:

- Database migration delayed: Extend to Month 4 (1-month buffer)
- Mobile app delayed: Push to Month 10 (focus on web first)
- UAT fails: Month 11 for additional fixes

CHAPTER 6 BUDGET & RESOURCES

6.1 PAYMENT SCHEDULE

Payment Item	Schedule	%
Inception Report	Month 1	15%
Integrated GIS web map	Month 5	25%
Progress monitoring system	Month 7	25%
Updated EBARMM website	Month 9	25%
Final Report	Month 12	10%

6.2 CLOUD INFRASTRUCTURE COSTS

6.2.1 Current Costs (Baseline)

Monthly: \$613.12 USD (~₱34,300 PHP at ₱56/\$1)

Annual: \$7,357 USD (~₱412,000 PHP)

Breakdown (August 2025 AWS Bill):

- EC2 - Other: \$240.80 (39%)
- Others (storage, data transfer, misc): \$163.33 (27%)
- Tax: \$65.70 (11%)
- AWS WAF: \$58.00 (9%)
- Amazon Virtual Private Cloud: \$44.74 (7%)
- AWS Support (Business): \$40.55 (7%)

Note: "Others" category likely includes EBS storage, S3, data transfer, snapshots, and miscellaneous AWS services.

6.2.2 Proposed Optimized Costs

Target Monthly: \$363-470 USD (~₱20,300-26,300 PHP)

Target Annual: \$4,356-5,640 USD (~₱244,000-316,000 PHP)

Cost Reduction: 24-41%

Annual Savings: \$1,717-3,001 USD (~₱96,000-168,000 PHP)

6.2.3 Optimization Strategies

Strategy	Estimated Savings	Notes
Image/video compression	\$50-80/mo	Reduce storage in "Others" category
S3 migration from EBS	\$40-60/mo	Move from expensive EBS to S3 Standard/Intelligent-Tiering
EC2 right-sizing	\$30-50/mo	Use t3 burstable instances, Reserved Instances
Review "Others" services	\$20-40/mo	Identify and eliminate unused resources
AWS WAF optimization	\$10-20/mo	Review rules, reduce unnecessary protections

Total Estimated Savings: \$150-250/month (24-41% reduction)

6.2.4 Budget for 12-Month Project

- Months 1-2: \$1,226 USD (full cost @ \$613/mo)
- Months 3-12: \$4,170 USD (optimized @ \$417/mo average)
- **Total Year 1:** ~\$5,396 USD (~₱302,000 PHP)
- **Savings in Year 1:** ~\$1,961 USD (~₱110,000 PHP)

6.3 ON-PREMISES INFRASTRUCTURE ANALYSIS

6.3.1 Hardware Requirements (Estimated)

Item	Specifications	Cost
Application Server	Intel Xeon 8-core, 32GB RAM, 2TB SSD	~₱150,000-200,000
Database Server	Intel Xeon 8-core, 64GB RAM, 4TB SSD RAID 10	~₱200,000-250,000
Network Storage	10-20TB NAS/SAN, RAID 5/6	~₱150,000-200,000
Networking	Firewall, switch, router, UPS	~₱165,000-240,000

Total Hardware: ~₱665,000-890,000 (one-time)

6.3.2 Ongoing Costs (Annual)

- Electricity: ~₱100,000-150,000
- Internet: ~₱60,000-100,000 (dedicated line)
- Maintenance: ~₱50,000-100,000
- Cooling/HVAC: ~₱30,000-50,000
- Physical security: ~₱20,000-40,000

Total Annual: ~₱260,000-440,000/year

6.3.3 Total Cost of Ownership (5-Year)

Cloud (Optimized):

- 5 years @ \$417/month (avg): \$25,020 (~₱1,401,000)

On-Premises:

- Hardware (Year 0): ₱775,000
- Annual costs: ₱350,000/year × 5 = ₱1,750,000
- **Total:** ~₱2,525,000

Cloud is ~₱1,124,000 cheaper over 5 years (44% savings)

6.3.4 Advantages/Disadvantages

Factor	Cloud	On-Premises
Upfront cost	Low	High (~₱775,000)
Maintenance	None (managed)	Requires IT staff
Scaling	Easy	Limited
Redundancy	Global	Limited
Control	Medium	Full
Internet dependency	Yes	For remote access only
5-year TCO	~₱1,350,000	~₱2,525,000

Recommendation: Stay on cloud (optimized AWS)

6.4 TEAM STRUCTURE

Woodfields Consultants Team

Team Leader / Project Manager

- Responsibilities: Overall project coordination, client liaison, cloud infrastructure management, budget and timeline management, risk management, deployment oversight

Software Developer

- Responsibilities: Backend development (Django, PostgreSQL), frontend development (React.js), mobile application development (Android), API development and integration, security implementation, database migration

GIS Specialist

- Responsibilities: GIS integration (PostGIS, GeoDjango), spatial analysis and mapping, QGIS workflow optimization, RouteShoot and drone video georeferencing, coordinate system management, GIS training for users

6.5 CLIENT RESPONSIBILITIES

MPW-BARMM must provide:

6.5.1 Access and Credentials

- AWS console access (IAM user)
- Database credentials (MySQL earmm2019)
- Server SSH access
- Domain name management (DNS)

6.5.2 Infrastructure Support

- AWS account maintained for 12 months

6.5.3 Personnel Support

- Project Chairman availability
- Technical Working Group for requirements
- System Administrator for coordination
- DEO representatives for UAT (Month 10)

6.5.4 Timely Reviews

- Inception Report (Month 1): 2 weeks
- Deliverables (Months 5, 7, 9): 1 week each
- Draft Final Report (Month 11): 2 weeks
- UAT participation (Month 10): 2 weeks

6.5.5 Decision Points

- Cloud vs. on-premises (Month 3-4)
- Technology stack discussion (Month 2)
- Acceptance of deliverables
- Production deployment readiness (Month 12)

6.6 DEVELOPMENT TOOLS & LICENSES

6.6.1 Open Source (No Cost):

- Python, Django, PostgreSQL, PostGIS
- React.js, Leaflet.js, Node.js
- Git, VS Code, QGIS
- Nginx, Redis, Celery
- Android Studio, Kotlin

6.6.2 Cloud Services (Usage-Based):

- AWS (EC2, S3, RDS, CloudFront)
- Let's Encrypt (free SSL)

6.6.3 Optional Paid Services:

- Sentry (error tracking): ~\$26/month
- SendGrid (email): ~\$20/month
- Twilio (SMS alerts): ~\$10-30/month

Estimated Tools Cost: \$50-80/month (~₱3,000-4,500 PHP)

6.7 INFRASTRUCTURE REQUIREMENTS

6.7.1 Development Environment:

- 3 laptops (Woodfields)
- Internet connection (Woodfields)

6.7.2 Staging Environment:

- AWS EC2 (t3.medium): ~\$30-40/month
- RDS (db.t3.small): ~\$25-35/month
- S3 storage: ~\$10-20/month
- **Total:** ~\$65-95/month

6.7.3 Production Environment:

- AWS EC2 (t3.large): ~\$70-90/month
- RDS (db.t3.medium): ~\$60-80/month
- S3 storage: ~\$75-90/month
- CloudFront: ~\$15-25/month
- **Total:** ~\$220-285/month (optimized)

Total Cloud (Dev + Staging + Prod): ~\$285-380/month

6.8 TRAINING & DOCUMENTATION

Training Sessions (Month 12):

- 3 sessions × 2 hours = 6 hours
- DEO users, MPW admins, GIS team

Documentation Deliverables:

- User Manual (web app)
- User Manual (mobile app)
- Administrator Guide
- API Documentation
- Database Schema Documentation
- Deployment Guide
- Troubleshooting Guide

Format: PDF and online (hosted on system)

Video Tutorials:

- 5-10 short videos (5-10 minutes each)
- Cover common tasks
- Hosted on YouTube or system

6.9 BUDGET SUMMARY

Item	Cost	Period	Notes
Consulting Services	[Per Contract]	12 months	Woodfields team
Cloud Infrastructure	~\$5,396 (~P302,000)	Year 1	Optimized costs (paid by WCI)
Development Tools	~\$50-80/month	12 months	Sentry, SendGrid, etc.
Staging Environment	~\$65-95/month	12 months	Separate from prod

AWS Cloud Costs - Paid by WCI:

Woodfields Consultants, Inc. (WCI) will shoulder the AWS cloud infrastructure costs for the 12-month project duration (~\$5,240 USD / ~P295,000 PHP). This ensures uninterrupted cloud operations during system development and deployment.

Note: WCI already prepaid 1 month of AWS costs during the kickoff meeting (September 2025) to avoid AWS account restrictions and ensure immediate project access.

On-Premises Option (not recommended):

- Hardware: ~P775,000 (one-time)
- Annual: ~P350,000/year

6.10 ASSUMPTIONS

1. Exchange rate: P56 = \$1 USD
2. AWS costs based on US East region pricing
3. Current usage patterns continue
4. No major AWS price changes
5. MPW-BARMM provides existing AWS account access (WCI pays for cloud costs)
6. DEOs available for UAT in Month 10
7. No major scope changes

CHAPTER 7 RISK MANAGEMENT

7.1 RISK ASSESSMENT MATRIX

#	Risk	Probability	Impact	Severity
1	Data migration complexity	M	H	HIGH
2	Legacy data quality issues	H	M	HIGH
3	Cloud cost overruns	M	M	MEDIUM
4	Database performance issues	L	H	MEDIUM
5	User adoption resistance	L	M	LOW
6	On-premises hardware delays	M	H	HIGH (if chosen)
7	Key personnel unavailability	L	H	MEDIUM
8	Scope creep	M	M	MEDIUM
9	Internet connectivity issues	M	M	MEDIUM
10	Security vulnerabilities	L	C	MEDIUM
11	Third-party API failures	L	M	LOW
12	Mobile app compatibility	M	M	MEDIUM

Probability: Low (L), Medium (M), High (H)

Impact: Low (L), Medium (M), High (H), Critical (C)

7.2 TOP RISKS & MITIGATION

7.2.1 RISK 1: Data Migration Complexity

Description: Migrating earmm2019 MySQL to PostgreSQL/PostGIS - 18 tables, years of data, no foreign keys, spatial conversion

Mitigation:

1. **Comprehensive testing** - Test on copy first, validate all records, compare counts
2. **Staged migration** - Lookup tables → projects → media, validate each stage
3. **Rollback plan** - Keep MySQL intact, document rollback procedures
4. **Validation scripts** - Check orphaned records, verify foreign keys, validate coordinates

Contingency: Allocate extra 2 weeks in Month 3, have database expert on standby

7.2.2 RISK 2: Legacy Data Quality Issues

Description: Manual data entry since 2016, no validation rules, possible duplicates, inconsistent formatting, missing fields

Mitigation:

1. **Data profiling** - Analyze quality before migration (Month 2)
2. **Cleaning scripts** - Remove duplicates, standardize formats, fill missing values
3. **Validation rules** - Implement strict validation in new system
4. **Manual review** - MPW-BARMM reviews flagged records

Contingency: Accept some issues as "historical", focus on preventing future problems, provide admin tools

7.2.3 RISK 3: Cloud Cost Overruns

Description: AWS costs exceed budget, unexpected spikes, inefficient allocation

Mitigation:

1. **Cost monitoring** - AWS Budget alerts (\$500/month threshold), weekly reviews
2. **Early optimization** - S3 migration in Month 2, right-size from start
3. **Resource limits** - Auto-shutdown dev/staging at night, delete unused resources
4. **Cost allocation** - Tag resources by project/environment

Contingency: Scale down non-critical environments, delay non-essential optimizations

7.2.4 RISK 4: Database Performance Issues

Description: Slow queries with large datasets, expensive PostGIS spatial queries, concurrent load

Mitigation:

1. **Proper indexing** - Index FKs, GIST on spatial columns, B-tree on query fields
2. **Query optimization** - Analyze slow queries (pg_stat_statements), use EXPLAIN
3. **Caching** - Redis for API responses, cache spatial queries, CDN for assets
4. **Load testing** - Test with realistic volumes, simulate concurrent users (Month 8)

Contingency: Upgrade RDS instance, add read replicas, implement aggressive pagination

7.2.5 RISK 5: User Adoption Resistance

Description: DEOs resist change, prefer old system, training insufficient

Mitigation:

1. **User involvement** - Include DEOs in UAT (Month 10), gather feedback
2. **Change management** - Communicate benefits, show problem-solving
3. **Comprehensive training** - Multiple sessions (Month 12), video tutorials, manuals
4. **Ongoing support** - Help desk, quick response, regular check-ins

Contingency: Extended training, one-on-one coaching, incentivize usage

7.2.6 RISK 6: On-Premises Hardware Delays (if chosen)

Description: Procurement takes months, vendor delays, customs issues, installation problems

Mitigation:

1. **Early decision** - Cloud vs. on-premises by Month 4, start procurement immediately
2. **Vendor guarantees** - Clear timelines, penalties for delays, backup vendor
3. **Parallel cloud work** - Continue development, deploy to cloud first (staging)
4. **Simplified hardware** - Start minimal, expand later, use commodity hardware

Contingency: Stay on cloud if delayed beyond Month 10, use 2-server setup, hybrid (S3 + on-premises compute)

7.2.7 RISK 7: Key Personnel Unavailability

Description: Team member sick/injured, leaves company, personal emergency

Mitigation:

1. **Cross-training** - All team members understand all components, pair programming
2. **Good documentation** - Code comments, architecture docs, decision log
3. **Backup resources** - Identify backup developers, contractor network
4. **Knowledge sharing** - Weekly syncs, code reviews, shared Git repository

Contingency: Bring in contractor, reduce scope, extend timeline 2-4 weeks

7.2.8 RISK 8: Scope Creep

Description: New requirements mid-project, "small" changes accumulate, unclear requirements

Mitigation:

1. **Clear scope** - Inception Report defines baseline
2. **Change control** - All changes require formal communication, impact assessment, discussion with MPW-BARMM
3. **Prioritization** - Must-have vs. nice-to-have, core features first
4. **Regular reviews** - Monthly meetings, monthly progress reports

Contingency: Defer non-critical to Phase 2, propose additional budget, negotiate timeline

7.2.9 RISK 9: Internet Connectivity Issues (DEOs)

Description: DEOs in remote areas, unstable/slow internet, cannot upload videos

Mitigation:

1. **Mobile app offline mode** - Store data locally, sync when available, queue uploads
2. **Chunked uploads** - Break files into chunks, upload piece by piece, resume if interrupted
3. **Compression** - Compress photos/videos before upload, reduce bandwidth
4. **Progress indicators** - Show upload progress, estimate time

Contingency: Provide USB drives for data transfer (worst case), MPW-BARMM collects monthly, upload centrally

7.2.10 RISK 10: Security Vulnerabilities

Description: SQL injection, XSS, CSRF attacks, unauthorized access, data breach

Mitigation:

1. **Secure coding** - Django ORM (prevents SQL injection), React (auto-escapes XSS), CSRF tokens
2. **Authentication** - Strong passwords, JWT tokens, session timeout, rate limiting
3. **Authorization** - Role-based access, DEO-based isolation, least privilege
4. **Security audit** - Penetration testing (Month 10), OWASP Top 10 checks, fix before production
5. **HTTPS everywhere** - Let's Encrypt SSL, redirect HTTP → HTTPS, HSTS headers

Contingency: Emergency security patch process, incident response plan, backups for recovery

7.2.11 RISK 11: Third-Party API Failures

Description: Google Maps down, SendGrid/Twilio down, S3 region outage

Mitigation:

1. **Graceful degradation** - Show cached map, queue emails/SMS for retry, fallback features
2. **Monitoring** - Health checks, alert team if down
3. **Alternatives** - OpenStreetMap fallback, alternative email service ready

Contingency: Manually send critical alerts, use local map tiles if extended outage

7.2.12 RISK 12: Mobile App Compatibility

Description: App crashes on some devices, Android fragmentation, different screen sizes

Mitigation:

1. **Broad testing** - Test on multiple devices (old/new), Android 8-13, different manufacturers
2. **Target widely** - Support Android 8.0+ (covers 95%+ devices), responsive layouts
3. **Permissions handling** - Request camera/GPS properly, handle denials gracefully
4. **Beta testing** - Deploy beta to 2-3 DEOs (Month 8), collect crash logs

Contingency: Identify minimum supported devices, provide fallback web app, direct support

7.3 RISK MONITORING

Weekly: Team reviews active risks, update probability/impact, document new risks

Monthly: Report risks in progress report, highlight critical risks

Quarterly: Deep dive review with MPW-BARMM

Risk Register: Maintained in shared document, updated real-time

7.4 ESCALATION PROCEDURES

Risk Level	Action
Low	Team handles internally, mention in weekly sync
Medium	Email project chairman, discuss in monthly meeting
High	Immediate call/email, emergency meeting if needed, written escalation
Critical	Same-day escalation, all-hands meeting, formal plan, daily updates

CHAPTER 8 NEXT STEPS

8.1 IMMEDIATE ACTIONS FOR PHASE 2

8.1.1 Week 1-2: Access and Setup

System Access:

- ☐ AWS console (IAM credentials)
- ☐ MySQL database (earmm2019)
- ☐ SSH access to EC2 instances
- ☐ Domain management
- ☐ Google Play Console (mobile app publishing)

Development Environment:

- ☐ Clone/export existing code
- ☐ Setup local dev environments
- ☐ Install PostgreSQL/PostGIS
- ☐ Install Django, React, dependencies
- ☐ Setup Git repository
- ☐ Configure CI/CD pipeline

Documentation Gathering:

- ☐ Database dump
- ☐ Current AWS architecture details
- ☐ User accounts list
- ☐ DEO contact information

8.1.2 Week 3-4: Database Migration Preparation

Database Analysis:

- ☐ Run data profiling scripts
- ☐ Identify data quality issues
- ☐ Document current schema
- ☐ Design PostgreSQL schema
- ☐ Write migration scripts

Cloud Optimization Planning:

- ☐ Audit AWS resources
- ☐ Identify optimization opportunities
- ☐ Plan S3 migration strategy
- ☐ Setup cost monitoring alerts

8.2 KEY DECISIONS REQUIRED

8.2.1 Decision 2: Cloud vs. On-Premises (Month 3-4)

Question: Stay on AWS (optimized) or migrate to on-premises?

Options:

- **Option A:** Stay on AWS - Lower 5-year TCO (~P1,350,000), easier management
- **Option B:** On-premises - Higher TCO (~P2,525,000), full control, high upfront

Analysis: Chapter 6.3

Required From: MPW-BARMM Project Chairman

Recommendation: Stay on cloud (optimized AWS)

8.3 DELIVERABLE ACCEPTANCE CRITERIA

8.3.1 Month 5: Integrated GIS Web Map

- ☐ Single map shows all data (projects, photos, roads)
- ☐ Layer toggle controls functional
- ☐ Photo clustering works
- ☐ Map loads in < 3 seconds
- ☐ GIS editing tools functional
- ☐ Validation rules work
- ☐ Role-based permissions work
- ☐ No critical bugs
- ☐ Demo successful

8.3.2 Month 7: Progress Monitoring System

- ☐ Georeferenced design plans upload works
- ☐ Photos overlay on plans
- ☐ Geofencing validates locations
- ☐ Drone videos upload and display
- ☐ Alert system sends notifications
- ☐ Performance acceptable
- ☐ Demo successful

8.3.3 Month 9: Updated Website & Mobile App

- ☐ React UI redesign complete
- ☐ Responsive design (mobile/tablet/desktop)
- ☐ Security audit passed
- ☐ Performance optimized
- ☐ Android app installs and runs
- ☐ RouteShoot captures video + GPS
- ☐ Geotagging works
- ☐ Offline sync functional
- ☐ Demo successful

8.3.4 Month 12: Final Report & System

- ☐ Production system deployed
- ☐ Data migration complete
- ☐ All objectives (A-G) delivered
- ☐ Training completed
- ☐ Documentation delivered
- ☐ System stable (99% uptime)
- ☐ Final Report accepted

8.4 COMMUNICATION PROTOCOLS

8.4.1 Regular Communication

Monthly Progress Meetings

- Frequency: Once per month
- Duration: 1-2 hours
- Attendees: Woodfields team + MPW-BARMM Project Chairman
- Agenda: Progress review, deliverable status, issues, next steps, decisions

Monthly Progress Reports

- Due: Last week of each month (Month 2+)
- Format: Written report (PDF)
- Contents: Completed tasks, upcoming tasks, deliverable status, budget, risks, screenshots

Deliverable Demo Sessions

- Month 5, 7, 9
- Live demonstration + Q&A

8.4.2 Issue Escalation

Normal Issues: Email, 2-day response, resolve in monthly meeting

Urgent Issues: Phone + email, same-day response, emergency meeting if needed

Critical Issues: Immediate call, urgent meeting within 24 hours, daily status updates

8.5 SUCCESS METRICS

On-Time Delivery:

- ☐ All 5 deliverables on schedule
- ☐ Monthly progress reports submitted on time

Quality:

- ☐ All acceptance criteria met
- ☐ Security audit passed
- ☐ Performance benchmarks achieved
- ☐ UAT passed

Budget:

- ☐ Cloud costs reduced by 24-41%
- ☐ Project within budget
- ☐ No cost overruns

User Satisfaction:

- ☐ DEOs trained successfully
- ☐ Positive UAT feedback
- ☐ System actively used post-deployment

Technical:

- ☐ 99% uptime in first 3 months
- ☐ Mobile app functional
- ☐ All 7 objectives delivered

- ☐ Documentation complete

8.6 PHASE 2 KICK-OFF CHECKLIST

MPW-BARMM:

- ☐ Review Inception Report
- ☐ Process first payment (15%)
- ☐ Provide system access credentials
- ☐ Introduce Woodfields to System Administrator
- ☐ Schedule cloud decision meeting (Month 3)
- ☐ Assign DEO representatives for UAT

Woodfields:

- ☐ Address any feedback on Inception Report
- ☐ Receive system access
- ☐ Begin Phase 2 activities
- ☐ Setup development environment
- ☐ Submit Month 2 progress report
- ☐ Schedule monthly meetings
- ☐ Begin database migration work

8.7 CONCLUSION

The E-BARMM System Update Project is ready to move to Phase 2 (Design & Development) following submission approval of this Inception Report.

This report has established:

- Clear understanding of current system and issues
- Detailed requirements for all 7 objectives
- Modern technology solution (Python/Django/PostgreSQL 14+/React/Android)
- 12-month work plan with concrete milestones
- Budget analysis with cloud optimization strategy
- Risk management approach
- Team structure and responsibilities

Woodfields Consultants is committed to:

- Delivering all objectives on time and within budget
- Maintaining regular communication with MPW-BARMM
- Providing high-quality, secure, and performant system
- Training users and transferring knowledge
- Supporting MPW-BARMM's infrastructure transparency goals

We look forward to:

- MPW-BARMM's review and feedback
- Beginning Phase 2 development
- Collaborating with DEOs during UAT
- Delivering a modern E-BARMM system that serves BARMM for years to come

Next Immediate Step: Begin Phase 2 development activities

