9.1 Project Management Methodology

Unisoftwares uses **Agile methodology** for software development and **Waterfall/Hybrid** for fixed-scope projects like website design.

9.2 Agile Development Process

9.2.1 Sprint Structure

- **Sprint Duration**: 2 weeks (10 working days)
- **Sprint Planning**: Monday morning (2 hours)
- **Daily Standup**: Every morning (15 minutes)
- **Sprint Review**: Friday afternoon (1 hour)
- **Sprint Retrospective**: Friday afternoon (30 minutes)

9.2.2 Roles

- **Product Owner**: Client or internal stakeholder
- **Scrum Master**: Project Manager
- **Development Team**: Developers, designers, QA

9.2.3 Artifacts

- **Product Backlog**: All features and requirements
- **Sprint Backlog**: Work selected for current sprint
- **Burndown Chart**: Progress tracking

9.3 Project Lifecycle (Waterfall Projects)

Phase 1: Initiation

- Project charter created
- Stakeholders identified
- Initial requirements gathered
- Feasibility assessment

Phase 2: Planning

- Detailed scope document
- Work breakdown structure (WBS)
- Timeline with milestones
- Resource allocation
- Risk assessment

Phase 3: Execution

- Design and development
- Regular client check-ins
- Progress tracking

Phase 4: Monitoring & Control

- Track against timeline and budget
- Change request management
- Quality assurance
- Status reports

Phase 5: Closure

- Final delivery and handoff
- Client training
- Documentation
- Post-project review
- Lessons learned

9.4 Communication & Reporting

9.4.1 Client Communication Schedule

- **Kickoff Meeting**: Day 1 (1-2 hours)
- **Weekly Status Calls**: Every Friday (30 minutes)
- **Ad-hoc Updates**: Via email or Slack as needed
- **Monthly Executive Summary**: Emailed 1st of month (for retainer clients)

9.4.2 Status Report Contents

- Completed work this period
- Work planned for next period
- Risks and issues
- Budget status (if applicable)
- Next milestone date

9.4.3 Escalation Path

- **Level 1**: Project Manager (response within 4 hours)
- **Level 2**: Department Head (response within 24 hours)
- **Level 3**: Director/VP (response within 48 hours)

9.5 Change Request Process

9.5.1 When is a Change Request Needed?

- Adding new features outside original scope
- Significant design changes after approval
- Changing technology stack mid-project
- Major timeline shifts

9.5.2 Change Request Form

Change Request #: _______

Project Name: ___________

Requested By: ___________

Date: __________

Description of Change:

[Detailed explanation]

Reason for Change:

[Business justification]

Impact Assessment:

- Timeline Impact: [+X weeks]

- Budget Impact: [+PKR X]

- Resource Impact: [Additional developer needed]

Priority: [ ] Critical [ ] High [ ] Medium [ ] Low

Approval:

Project Manager: ___________ Date: ______

Client: ___________ Date: ______

9.5.3 Change Request Workflow

1. Client or team member submits change request
2. PM assesses impact (time, cost, resources)
3. PM discusses with client
4. Client approves or rejects
5. If approved: Update project plan and contract addendum
6. Implement change in next sprint/phase

9.6 Risk Management

9.6.1 Common Project Risks

Technical Risks:

- Third-party API downtime or changes
- Technology compatibility issues
- Security vulnerabilities discovered
- Performance bottlenecks

Resource Risks:

- Team member unavailability (sick leave, resignation)
- Skill gaps in new technology
- Freelancer delays

Client Risks:

- Delayed feedback or approvals
- Scope creep
- Content not provided on time
- Changing requirements

External Risks:

- Payment gateway regulatory changes
- Currency fluctuations (international projects)
- Internet/infrastructure outages

9.6.2 Risk Mitigation Strategies

- **Buffer Time**: Add 20% buffer to estimates
- **Backup Resources**: Cross-train team members
- **Clear Contracts**: Define scope, deliverables, client responsibilities
- **Regular Backups**: Code repo + database backups
- **Dependency Tracking**: Identify critical path items early

---

9.7 Quality Assurance Process

9.7.1 Code Review

- All code peer-reviewed before merge
- Automated tests (unit tests, integration tests) where applicable
- Code quality tools (ESLint, Prettier, SonarQube)

9.7.2 Testing Phases

1. **Unit Testing**: Individual component testing (developers)

2. **Integration Testing**: Component interaction testing
3. **System Testing**: End-to-end testing (QA team)
4. **User Acceptance Testing (UAT)**: Client testing
5. **Regression Testing**: After bug fixes or changes

9.7.3 Bug Tracking

- Use Jira or ClickUp for bug logging
- Bug severity levels:
  - **Critical**: Site down, data loss, security breach
  - **High**: Major feature broken
  - **Medium**: Minor feature issue, workaround exists
  - **Low**: Cosmetic issues, nice-to-have fixes

FAQs (Project Management)

Q1: How often do we communicate with clients?
A: Weekly status calls (Fridays), daily during critical phases, plus ad-hoc via email/Slack. (Source: Project_Management_Guide.pdf, Section 9.4.1)

Q2: What happens if project scope changes?
A: Submit a change request form, PM assesses impact, client approves, then we update timeline/budget. (Source: Project_Management_Guide.pdf, Section 9.5)

Q3: How do you handle bugs found after launch?
A: Bugs within warranty period (1-3 months) fixed free. Post-warranty bugs fixed under maintenance contract. (Source: Project_Management_Guide.pdf, Section 9.7.3)