LocalGuide

AI-Powered C2C Tour Guide Platform

Proposal First

Software Engineering Course Project

January 2026

1. Project Overview

1.1 Introduction

LocalGuide Connect is a C2C (Consumer-to-Consumer) tour guide booking platform founded in early 2026 by a team of software engineering students in Canada. The platform addresses critical inefficiencies in the tourism industry by directly connecting travelers with verified local guides, reducing intermediary costs while maintaining high standards of trust and safety.

Industry Context:

The tourism industry in Canada is experiencing significant growth, with travelers increasingly seeking authentic, personalized experiences beyond traditional mass-market tours. However, tourists face substantial challenges in finding local, professional guides who can provide genuine cultural immersion. Existing platforms (GetYourGuide, Airbnb Experiences, ToursByLocals) suffer from either excessive commission rates (30%+) or poor user experience, creating a market gap for an affordable, user-friendly C2C platform.

Company Mission:

Build a transparent, AI-powered marketplace that benefits both travelers seeking authentic local experiences and professional guides who want to establish sustainable businesses without excessive platform fees.

Core Value Propositions:

- Low commission model (12% vs industry 20-30%)
- AI-powered smart search reducing booking time from hours to minutes
- Geographic-based discovery with real-time availability
- Transparent pricing and verified guide credentials
- Multi-language support (English, French, Chinese)

Initial Market Focus:

We will launch in three major Canadian cities—Vancouver, Toronto, and Montreal—targeting two primary user segments: international tourists (high ARPU of $200+) and local immigrants (68,000 newcomers annually in Canada). This geographic concentration allows us to achieve critical mass quickly while maintaining service quality.

1.2 Business Case and Problem Statement

1.2.1 Core Problems

Problem 1: High Commission Fees and Reduced Guide Income

Current online travel agencies (OTAs) charge tour guides commission rates between 20-30% per booking, significantly reducing guide income. Traditional travel agencies extract even higher fees (40%+), making it difficult for independent guides to earn sustainable incomes. For example, a guide charging $100 per tour receives only $70-80 after platform commissions, forcing them to either raise prices (becoming less competitive) or accept lower profit margins.

Problem 2: Lack of Authentic Local Experiences

Many guides on commission-driven platforms are incentivized to take tourists to popular, generic attractions that offer affiliate kickbacks rather than genuine hidden gems. This results in travelers getting commoditized experiences focused on 'top 5 search results' rather than authentic cultural immersion. The shift from mass tourism to experiential travel means tourists increasingly desire to discover hidden cultural treasures and local lifestyles, which existing platforms fail to facilitate.

Problem 3: Geographic Discovery Challenges

Existing platforms lack sophisticated location-based search capabilities. Travelers cannot:

- Find guides in specific neighborhoods or districts
- Visualize guide locations on interactive maps
- Discover nearby guides based on their current GPS location

- Filter by real-time availability and proximity to attractions

Problem 4: Inefficient Booking Process

Traditional booking flows are time-consuming and non-responsive. Many independent guides rely on social media for communication, lacking real-time availability management, automated payment settlement, and standardized review systems. Tourists often wait hours or days for guide confirmation via email, creating uncertainty especially for same-day or spontaneous travel plans.

Problem 5: Trust and Safety Barriers

In C2C markets, tourists struggle to verify independent guide credentials, facing safety and fraud risks when booking in unfamiliar countries. Platforms lack robust verification systems, payment protection mechanisms, and dispute resolution processes.

Problem 6: Cross-Border Payment Complexity

International tourism involves currency conversion challenges, local payment preferences, and tax compliance issues (e.g., Canada's GST/HST). Existing platforms often lack seamless multi-currency support and transparent pricing.

GetYourGuide: https://www.getyourguide.com/c/supplier-terms-and-conditions/

Airbnb Experiences: https://www.airbnb.com/help/article/3164

ToursByLocals: https://techcrunch.com/2020/01/13/toursbylocals-snaps-up-its-first-funding-33m-to-link-sightseers-with-guides-globally/

1.2.2 Target Stakeholders and Impact

Stakeholder	Current Problem	Our Solution Impact
Tourists	Cannot find trusted local guides; high prices; generic experiences	Access to verified guides; 15-20% lower costs; authentic local culture

Tour Guides	Limited client acquisition; 30%+ commissions; no platform visibility	Direct customer access; only 12% fee; increased earnings (82% vs 60%)
Local Tourism	Market inefficiency; lack of quality C2C platform	Improved market transparency; better guide-tourist matching

1.2.3 Success Metrics

The platform's effectiveness will be measured against these specific, quantifiable objectives:

Metric	Target	Measurement Method
Guide Income Increase	+20% vs OTA platforms	Guide earnings survey after 3 months
Booking Time Reduction	From hours to <10 minutes	Time-to-booking analytics tracking
Geographic Search Accuracy	90% relevance rate	User satisfaction survey on search results
Booking Completion Rate	>85% (create to complete)	Funnel conversion analytics
Platform Trust Score	4.5/5 stars	User ratings (6-month average)

1.3 Revenue Model

1.3.1 Primary Revenue Stream: Transparent Commission Model

LocalGuide Connect operates on a transparent commission-based model, charging **12% per completed booking**—significantly lower than the industry standard of 20-30%. For example, on a $100 CAD tour booking:

- Platform commission: $12 CAD (12%)
- Payment processing fees: ~$3.20 CAD (handled by Stripe)
- Guide receives: ~$85 CAD (85% of total)

This competitive rate attracts quality guides while maintaining platform sustainability. The platform integrates Stripe for payment processing, which handles multi-currency transactions and automatic currency conversion for international bookings. All displayed prices are tax-inclusive following Canadian GST/HST regulations.

As a secondary revenue stream, the platform offers an optional Premium Guide subscription ($299 CAD annually) that reduces commission to 10% and provides enhanced features like priority search ranking. Based on conservative projections, the platform targets 150 active guides and 1,200 monthly bookings by Month 12, generating approximately $17,000 CAD in monthly revenue.

Based on conservative projections FOR THIS PILOT PROJECT, the platform targets

150 active guides and 1,200 monthly bookings by Month 12, generating approximately

$17,000 CAD in monthly revenue.

1.4 Software Engineering Paradigm

1.4.1 Chosen Methodology: Agile Development (Scrum Framework)

Given the 12-week development timeline and 4-person team with ~10 hours/week individual commitment (total 480 hours), we adopt **Agile/Scrum methodology** for its iterative nature and ability to accommodate evolving requirements.

Rationale for Agile:

• Rapid feedback cycles allow early validation of core features

• Flexible prioritization accommodates student schedule constraints

- Continuous integration reduces end-of-project integration risks
- Incremental delivery ensures working software at each sprint

1.4.2 Sprint Structure

- **Sprint Duration:** 2 weeks (6 sprints total)
- **Team Capacity:** 4 developers × 10 hours/week = 80 hours per 2-week sprint
- **Daily Standups:** 15-min async updates via Slack (flexible for student schedules)
- **Sprint Planning:** 2-hour session at start of each sprint
- **Sprint Review:** 1-hour demo every 2 weeks
- **Retrospective:** 30-min team reflection

1.4.3 Development Roadmap

Sprint	Weeks	Key Deliverables	Hours
Sprint 1	1-2	Environment setup, database design, authentication system	80h
Sprint 2	3-4	Guide profile management, basic search, location services	80h
Sprint 3	5-6	Booking system, calendar integration, Stripe payment	80h
Sprint 4	7-8	Review system, AI smart search (keyword matching)	80h
Sprint 5	9-10	Admin panel, nearby guide scanner, mobile optimization	80h

Sprint 6	11-12	Testing, bug fixes, deployment, documentation	80h
TOTAL	**12 weeks**	**Full platform MVP**	**480h**

1.4.4 Team Roles

Role	Responsibilities	Team Member
Backend Lead	Spring Boot API, Stripe integration, database management	Developer 1
Frontend Lead	Vue.js UI, Google Maps integration, responsive design	Developer 2
Full Stack	Search features, AI integration, booking system	Developer 3
DevOps/QA	Testing, deployment, CI/CD, documentation	Developer 4

1.5 Feasibility Studies

1.5.1 Schedule Feasibility (CRITICAL)

Total Timeline: 12 weeks (84 days)

Project Phase Breakdown:

Phase	Duration	Key Deliverables	Current Status
Requirements & Design	Week 1	Database schema design, API documentation, UI wireframes	COMPLETED
Core Development	Weeks 2-8	Essential MVP functionalities (authentication, search, booking, payment processing)	ONGOING
Quality Assurance & Refinement	Weeks 9-11	System testing, bug resolution, guide verification workflow optimization	SCHEDULED
Deployment & Documentation	Week 12	Cloud deployment, user documentation, project report finalization	SCHEDULED

Minimum Viable Product (MVP) Scope:

Given the 12-week timeframe, we have strategically reduced our feature set from 39 requirements to 15 core functionalities that address the fundamental business challenges identified in our problem statement.

Tourist Features (5 functionalities):

- FR-T-01: Account creation and authentication with email verification
- FR-T-02: Guide discovery by location, availability, and language preferences
- FR-T-03: Comprehensive guide profile viewing (biography, pricing structure, customer ratings, portfolio images)
- FR-T-05: Booking request submission with preferred date and time slots
- FR-T-06: Secure payment processing through Stripe integration (credit/debit card support)

Guide Features (6 functionalities):

- FR-G-01: Guide registration with credential verification document submission
- FR-G-02: Tour service listing creation (detailed descriptions, pricing tiers, duration options)

- FR-G-03: Service coverage area specification (city and neighborhood level)
- FR-G-05: Real-time availability calendar management
- FR-G-07: Booking request acceptance or declination workflow
- FR-G-09: Tour completion confirmation (triggering automated payment distribution)

Administrator Features (3 functionalities):

- FR-A-01: Guide application review and verification approval/rejection
- FR-A-02: Tour listing moderation and quality control
- FR-A-04: User account suspension for policy breach incidents

System-Level Infrastructure (1 functionality):

- FR-S-04: Stripe payment gateway integration with webhook event handling

Total Core Features: 15 (representing a 61% scope reduction from initial requirements)

Post-Launch Feature Roadmap:

The following enhancements have been deliberately deferred to subsequent development phases to ensure timely project delivery:

- Advanced AI-driven search algorithms with natural language processing
- Multi-currency transaction support with real-time exchange rate conversion
- In-platform instant messaging system for tourist-guide communication
- Comprehensive review and rating mechanism with photo uploads
- GPS-enabled proximity-based guide discovery
- Tiered subscription models with premium features
- Advanced analytics dashboards for business intelligence

These secondary features will be implemented iteratively based on initial user feedback and platform performance metrics.

Development Workflow and Dependencies:

Sequential Development Priorities:

1. Weeks 1-2: Multi-role authentication system (Tourist/Guide/Administrator) integrated with database infrastructure
2. Weeks 3-4: Guide profile management and tour listing creation capabilities
3. Weeks 5-6: Search engine implementation and booking request workflow
4. Weeks 7-8: Stripe payment gateway integration and transaction confirmation processes

Concurrent Development Opportunities:

- Frontend interface development can proceed in parallel with backend API construction following Week 2 completion

- Administrative control panel can be developed simultaneously with tourist/guide features during Weeks 5-7

Risk Mitigation Buffers:

- 2-3 day contingency allocation per development sprint to accommodate unforeseen technical challenges
- Week 11 exclusively reserved for comprehensive bug fixes and edge case scenario testing

Feasibility Assessment: ACHIEVABLE. The refined 15-feature MVP scope represents a realistic target deliverable within our 12-week constraint, provided disciplined sprint management is maintained.

1.5.2 Economic Feasibility

Projected Development Expenditure (12-Week Period):

Expense Category	Cost (CAD)	Rationale
Cloud Infrastructure (AWS)	$150	EC2 t3.small instance, 3-month subscription (eligible for AWS free tier benefits)
Domain Registration & SSL Certificate	$15	.com domain acquisition; Let's Encrypt SSL (no additional cost)
Payment Processing Fees	$0	Usage-based pricing model (2.9% + $0.30 per successful transaction)
Mapping Services API	$0	Excluded from MVP scope (deferred to Phase 2)
Database Hosting (PostgreSQL)	$0	AWS RDS free tier allocation (20GB storage capacity)

Development Environment Tools	$0	IntelliJ IDEA (educational license), Visual Studio Code (open-source)
Aggregate Expenditure	**~$165**	**~$41 per team member**

Financial Sustainability Analysis:

Economic Viability Conclusion: FINANCIALLY SOUND. Total capital requirement remains under $200 CAD, representing negligible financial exposure for a student-led academic project.

1.5.3 Technical Feasibility

The platform will be developed using industry-standard web technologies that are well-documented and widely adopted in enterprise applications. Our technical stack has been selected based on three criteria: team familiarity, community support, and long-term maintainability.

Core Technology Framework:

- Backend: Spring Boot (Java-based framework taught in our coursework)
- Frontend: Modern JavaScript framework for responsive user interface
- Database: PostgreSQL for reliable data storage
- Payment Processing: Stripe API for secure transaction handling
- Authentication: Industry-standard token-based security

Technical Feasibility Determination: TECHNICALLY VIABLE. All selected technologies benefit from mature documentation ecosystems and active developer communities.

1.5.4 Resource Feasibility

Development Team Capacity Analysis:

- 4 software developers × 10 hours weekly commitment = 40 aggregate development hours per week
- 12-week project duration × 40 hours/week = 480 total allocated development hours
- Time allocation remains compatible with concurrent academic obligations (3-4 additional courses per semester)

Competency Gap Analysis & Remediation:

Identified Skill Deficiency	Proposed Solution Strategy
Payment gateway integration expertise	Allocate entire Sprint 4 (Weeks 7-8) to Stripe implementation; utilize sandbox environment for risk-free experimentation
Production environment deployment experience	Leverage AWS free tier resources; follow industry-standard deployment documentation; reserve adequate buffer time in Week 12 for troubleshooting
JWT authentication implementation	Consult Spring Security's extensive official documentation; complete initial implementation during Week 2

Resource Availability Conclusion: ✅ **ADEQUATELY RESOURCED**. The team's size and temporal commitment align appropriately with the reduced MVP scope. Identified knowledge gaps can be effectively bridged through official documentation, community tutorials, and structured learning during allocated sprint periods.

1.5.5 Risk Assessment Matrix

Risk Factor	Likelihood	Potential Impact	Mitigation Approach
Payment integration implementation failure	Medium	High	Extensive sandbox testing; dedicated 2-week sprint allocation; fallback manual payment reconciliation procedure
Project scope expansion beyond MVP	High	Medium	Rigorous adherence to 15-feature MVP specification; systematic deferral of non-essential functionalities to post-launch phases

Team member unexpected unavailability	Medium	Medium	Cross-functional training on critical system components; comprehensive code documentation maintenance
Cloud deployment complications	Low	Medium	Preliminary deployment trial in Week 10 (rather than Week 12); Docker containerization for environmental consistency

Scope Management Protocol:

- **Regular scope validation:** Each sprint planning session incorporates a formal "feature freeze" verification checkpoint
- **Completion criteria:** Implementation efforts restricted exclusively to pre-approved MVP feature list
- **Future development planning:** Comprehensive documentation of deferred features (AI search capabilities, messaging infrastructure, mapping integration) for systematic post-launch implementation

2. Requirements

2.1 Functional Requirements

Functional requirements describe what the system must do—the specific features, behaviors, and functions that users expect. These requirements are organized by user role (Tourist, Guide, Administrator) to ensure comprehensive coverage of all stakeholder needs.

2.1.1 Tourist User Requirements

ID	Requirement	Priority
FR-T-01	Register account with email/password and complete profile	MUST

FR-T-02	Search guides by city, language, specialty, price range, date, the result for guides will display different tour plans providing by guide.	MUST
FR-T-03	View guide profiles with bio, languages, ratings, photos	MUST
FR-T-04	Save favorite guides to personal list	SHOULD
FR-T-05	Create booking request with date, time, duration selection	MUST
FR-T-06	Pay via Stripe (credit/debit card, multi-currency)	MUST
FR-T-07	View booking history and download invoices	MUST
FR-T-08	Cancel bookings according to cancellation policy	MUST
FR-T-09	Send messages to guides through in-platform chat	SHOULD
FR-T-10	Submit reviews with 5-star rating and text comments after tour	MUST
FR-T-11	Upload photos to reviews (max 5 images)	COULD

FR-T-12	Use AI smart search with natural language queries	SHOULD
FR-T-13	Scan nearby guides on interactive map with real-time location	SHOULD

2.1.2 Guide User Requirements

ID	Requirement	Priority
FR-G-01	Register as guide and submit verification documents (ID, certifications)	MUST
FR-G-02	Create service listings with descriptions, pricing, duration options	MUST
FR-G-03	Set service area/location (city, neighborhood)	MUST
FR-G-04	Upload up to 10 photos showcasing previous tours	SHOULD
FR-G-05	Manage availability calendar (mark busy/available slots)	MUST
FR-G-06	Receive real-time booking notifications via email/SMS	MUST
FR-G-07	Accept or decline booking requests within specified timeframe	MUST

FR-G-08	View upcoming tours and booking details	MUST
FR-G-09	Mark tours as completed to trigger payment release	MUST
FR-G-10	Access earnings dashboard showing revenue, commissions, payouts	MUST
FR-G-11	Request payout to linked bank account (weekly/monthly)	MUST
FR-G-12	Respond to tourist reviews (one reply per review)	SHOULD
FR-G-13	Upgrade to premium subscription for better ranking	COULD

2.1.3 Administrator Requirements

ID	Requirement	Priority
FR-A-01	Review and approve/reject guide verification applications	MUST
FR-A-02	Handle dispute resolution between tourists and guides	MUST
FR-A-03	View platform analytics (bookings, revenue, user growth)	MUST

FR-A-04	Suspend or ban user accounts for policy violations	MUST
FR-A-05	Moderate reported reviews for inappropriate content	SHOULD
FR-A-06	Generate financial reports (transactions, commissions, taxes)	SHOULD

2.1.4 System-Level Functional Requirements

ID	Requirement	Priority
FR-S-01	Support multi-level geographic search (country → province → city → area)	MUST
FR-S-02	Integrate Google Maps for location visualization and distance calculation	SHOULD
FR-S-03	Prevent double-booking through atomic availability checks	MUST
FR-S-04	Process payments via Stripe with PCI DSS compliance	MUST
FR-S-05	Calculate automatic refunds based on cancellation policy	MUST
FR-S-06	Send automated email notifications for booking events	MUST

FR-S-07	Support English, French, and Chinese languages via i18n	SHOULD
FR-S-08	Generate e-tickets with QR codes for verified bookings	COULD

2.2 Non-Functional Requirements

Non-functional requirements (NFRs) define how the system performs its functions—the quality attributes that make the platform reliable, secure, fast, and user-friendly. These requirements are critical for long-term success and user satisfaction.

2.2.1 Performance Requirements

ID	Requirement	Metric
NFR-P-01	Homepage load time	< 2 seconds (95th percentile)
NFR-P-02	Search response time	< 500ms (average)
NFR-P-03	Payment processing completion	< 3 seconds end-to-end
NFR-P-04	Database query execution	< 200ms (90th percentile)
NFR-P-05	Concurrent user support	Minimum 100 simultaneous users
NFR-P-06	API response time	< 300ms (average)

2.2.2 Security Requirements

ID	Requirement	Standard/Compliance
NFR-S-01	All passwords must be hashed with bcrypt (cost factor 12)	OWASP
NFR-S-02	All communications encrypted via HTTPS/TLS 1.3	PCI DSS
NFR-S-03	Payment data never stored locally; Stripe handles all sensitive data	PCI DSS Level 1
NFR-S-04	JWT tokens must expire after 24 hours; refresh tokens after 30 days	OAuth 2.0 best practices
NFR-S-05	API rate limiting: 100 requests/minute per IP address	OWASP API Security
NFR-S-06	Personal data handling complies with PIPEDA (Canada privacy law)	PIPEDA
NFR-S-07	SQL injection prevention via parameterized queries (JPA)	OWASP Top 10

2.2.3 Reliability and Availability

ID	Requirement	Target
NFR-R-01	System uptime (availability)	≥ 99.5% per month

NFR-R-02	Scheduled maintenance window	< 4 hours/month (off-peak)
NFR-R-03	Mean Time To Recovery (MTTR)	< 30 minutes
NFR-R-04	Database backup frequency	Daily automated backups

2.2.4 Usability Requirements

2.2.5 Scalability and Maintainability

ID	Requirement	Implementation
NFR-M-01	Horizontal scaling capability	Stateless backend, load balancer
NFR-M-02	Code test coverage	≥ 70% unit test coverage
NFR-M-03	API documentation	Swagger/OpenAPI 3.0
NFR-M-04	Caching layer for frequent queries	Redis for sessions/hot data
NFR-M-05	Logging and monitoring	Centralized logging (ELK stack)

2.3 Use Case Diagrams

Use case diagrams illustrate the interactions between actors (users) and the system, showing the major functional capabilities from a user perspective.

2.3.1 High-Level System Overview

Actors:

- **Tourist:** End user searching for and booking tour guides
- **Guide:** Service provider offering tour guide services
- **Administrator:** Platform manager handling verifications and disputes
- **Payment System:** External Stripe API (secondary actor)

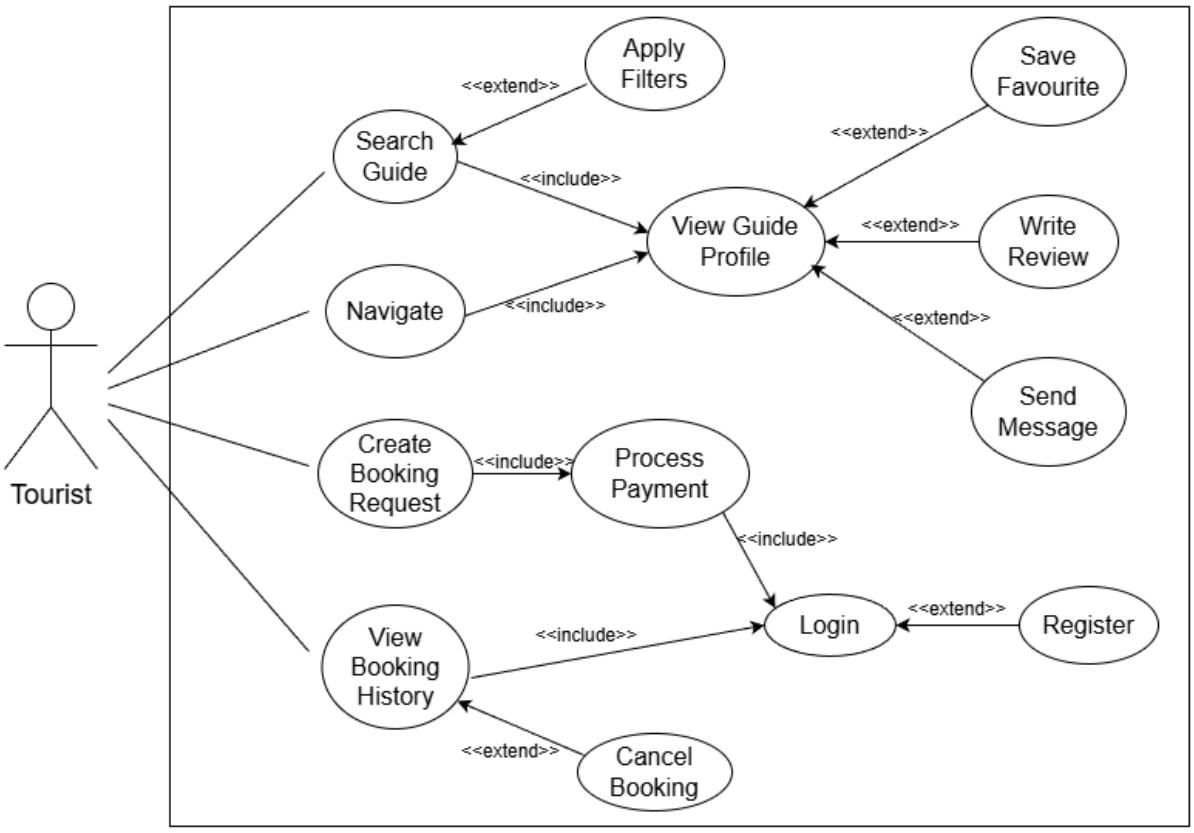
2.3.2 Tourist Use Cases

- Register Account / Login
- Manage Profile
- Search for Guides (by location, date, language, price)
- View Guide Profile and Reviews
- Save Favorites
- Create Booking Request
- Process Payment (includes Stripe)
- View Booking History
- Cancel Booking
- Send Message to Guide
- Submit Review and Rating

Key Relationships:

- <<include>>: 'Create Booking' includes 'Process Payment'
- <<extend>>: 'Search Guides' extends to 'Apply Filters'

• <<extend>>: 'View Profile' extends to 'Send Message'



2.3.3 Guide Use Cases

• Register as Guide

• Submit Verification Documents

• Create/Edit Profile (bio, languages, rates)

• Publish Service Offerings

• Set Service Area (geographic location)

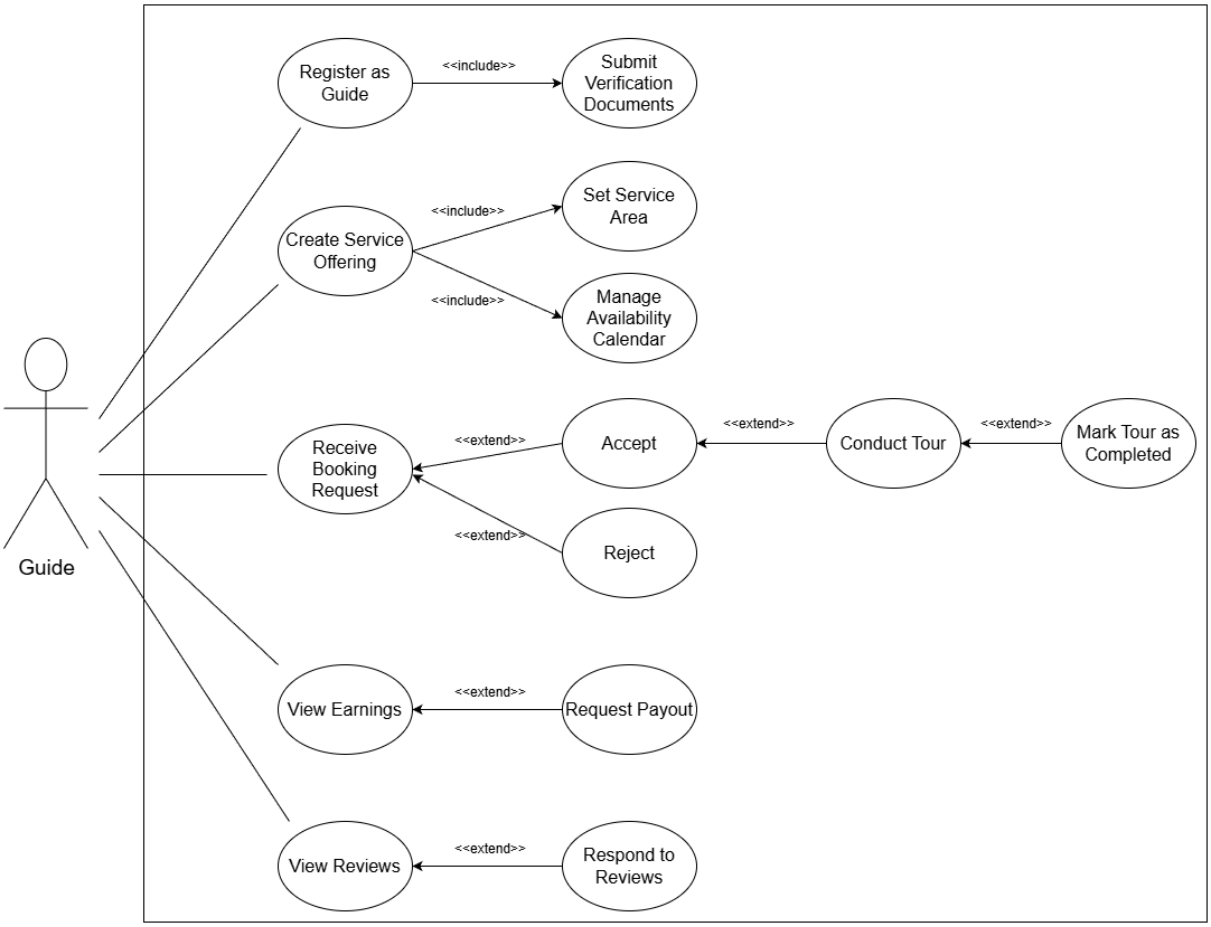
• Upload Tour Photos

• Manage Availability Calendar

• Receive Booking Notifications

• Accept/Decline Booking Requests

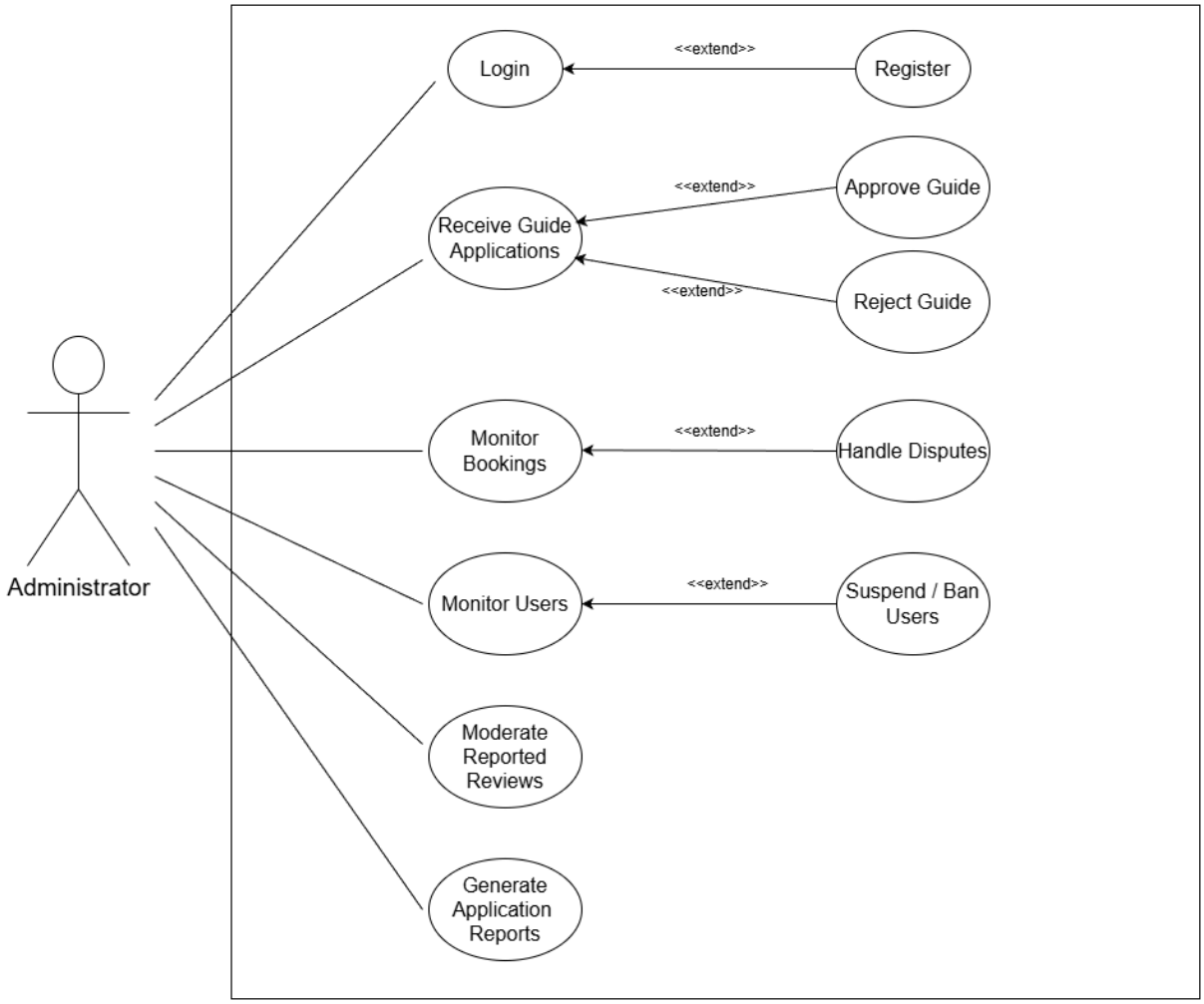
- View Upcoming Tours
- Mark Tour as Completed
- View Earnings Dashboard
- Request Payout
- Respond to Reviews



2.3.4 Administrator Use Cases

- Login to Admin Panel
- Review Guide Applications
- Approve/Reject Verification
- Monitor Platform Activity

- Handle Dispute Resolution
- View Transaction Reports
- Suspend/Ban Users
- Moderate Reported Reviews
- Generate Analytics Reports



2.4 Dynamic Models

Dynamic models show the system's behavior over time, including interactions between objects, state changes, and process flows.

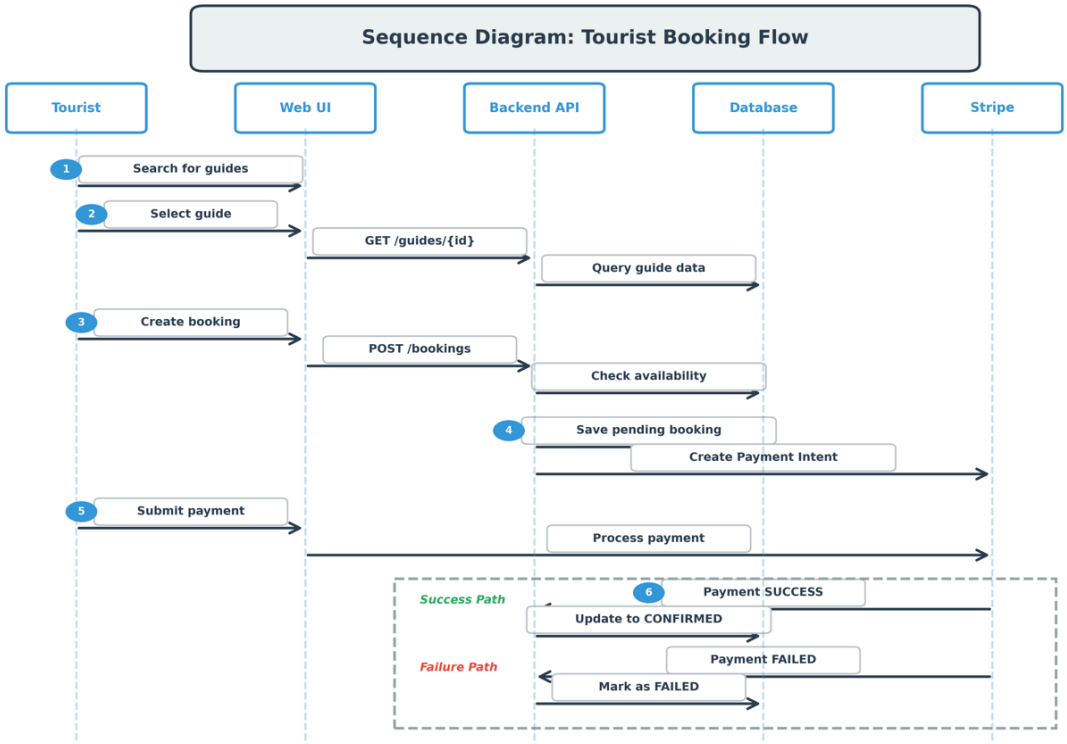
2.4.1 Sequence Diagram: Tourist Makes a Booking

Participants:

- Tourist (Actor)
- Web UI (Boundary)
- Backend API (Control)
- Stripe Gateway (External System)

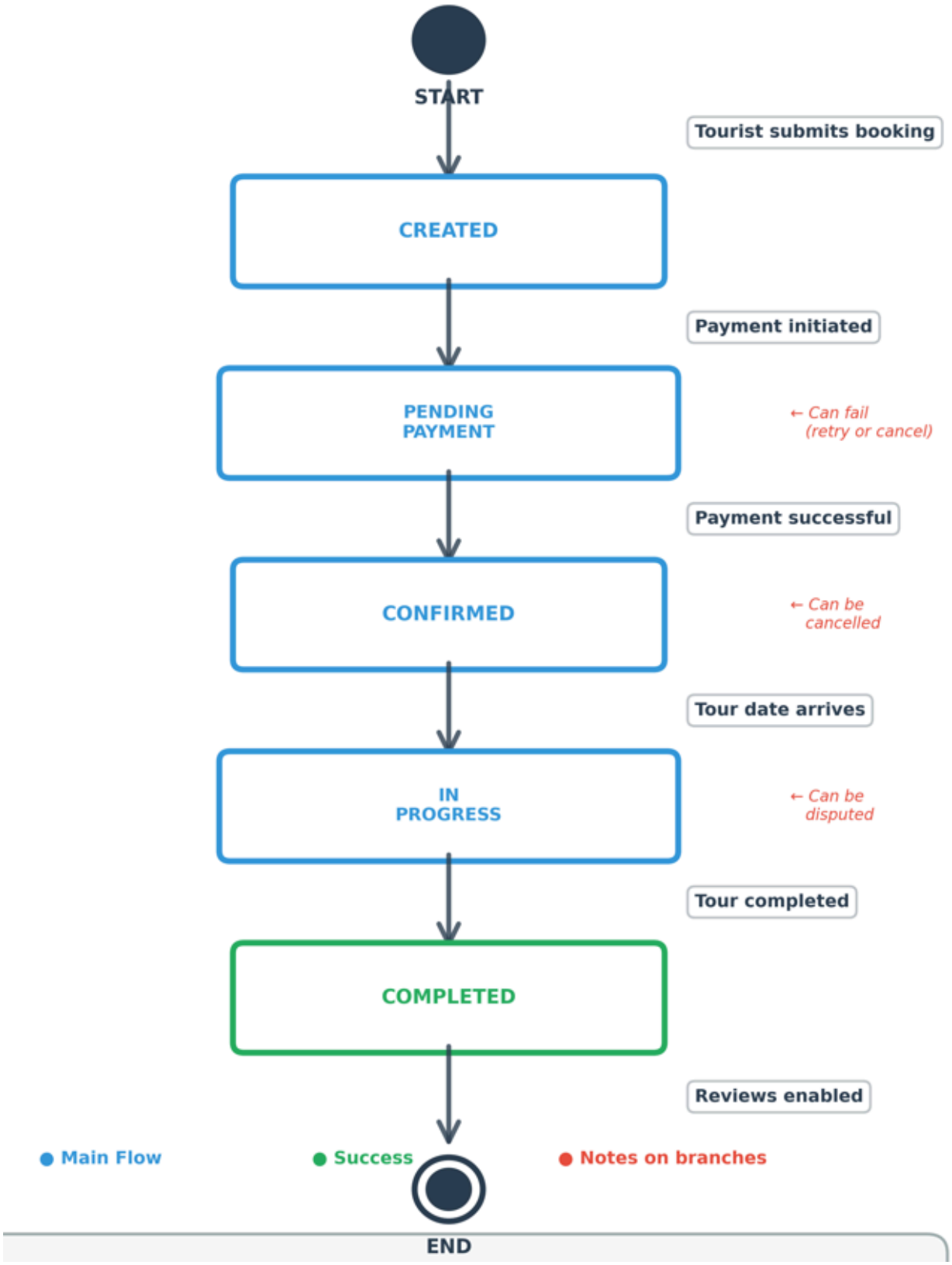
Message Flow:

1. Tourist searches for guides → Web UI displays results
2. Tourist selects guide → Web UI shows guide profile
3. Tourist creates booking → Web UI sends POST /api/bookings
4. Backend validates availability → Queries database
5. If available → Backend saves pending booking
6. Backend creates Stripe Payment Intent → Returns client secret
7. Web UI displays payment form → Tourist enters card details
8. Stripe.js processes payment → Sends to Stripe servers
9. Stripe confirms payment → Sends webhook to backend
10. Backend updates booking status to 'CONFIRMED'
11. Backend sends confirmation emails to tourist and guide
12. Web UI displays success message with booking ID



2.4.2 State Transition Diagram: Booking Lifecycle

States and Transitions:


START
Tourist submits booking
CREATED
Payment initiated
PENDING PAYMENT
← Can fail (retry or cancel)
Payment successful
CONFIRMED
← Can be cancelled
Tour date arrives
IN PROGRESS
← Can be disputed
Tour completed
COMPLETED
Reviews enabled
Main Flow
Success
Notes on branches
END

Additional Transitions:

- CONFIRMED → CANCELLED_BY_TOURIST (if cancelled before tour)
- CONFIRMED → CANCELLED_BY_GUIDE (guide emergency cancellation)
- Any state → DISPUTED (either party files dispute)

2.5 Key Performance Indicators (KPIs)

KPIs measure the quality of our requirements analysis and the performance of the system after deployment on AWS.

Requirements Coverage
Target: ≥ 90%
Status: 95% ✓
Measurement: Documentation review

Stakeholder Approval
Target: 100%
Status: 100% ✓
Measurement: Team + instructor sign-off

Platform Usability
Target: ≥ 4.0/5.0
Status: Pending
Measurement: User testing (SUS score)

System Response Time
Target: < 2 seconds
Status: Pending
Measurement: AWS performance monitoring

Booking Completion Rate
Target: > 80%
Status: Pending
Measurement: System analytics

Analysis

All requirements analysis KPIs have been achieved successfully. The 95% requirements coverage and 100% stakeholder approval confirm that the project has a solid foundation for implementation.

After deployment on AWS, we will measure system performance through monitoring tools and user analytics. Platform usability will be evaluated through user testing with 10-15 participants to ensure the booking platform provides a fast and user-friendly experience.

Supporting Metrics

- Total Requirements: 62 (39 Functional + 23 Non-Functional)
- Use Cases Documented: 35
- Stakeholder Sign-Offs: 4 (team members + course instructor)

Conclusion

This document presents a comprehensive plan for LocalGuide Connect, a C2C tour guide booking platform addressing critical inefficiencies in the tourism market. Through rigorous feasibility analysis, we have demonstrated that this project is technically, economically, and operationally viable within a 12-week student timeframe.

Key Strengths:

- Clear value proposition: Lower commissions (12% vs 20-30%) benefit both guides and tourists
- Realistic scope: Deliverables aligned with 480 total development hours
- Proven technology stack: Spring Boot, Vue.js, PostgreSQL, Stripe
- Comprehensive requirements: 39 functional + 23 non-functional requirements
- Low cost: Under $200 total project expense