OpenBid: Map-first Bidding Marketplace

DIGT 2107 — Project Iteration 1.1: Initial Vision & Planning

Team 1: Tyler, Mani, Yanness, Alaister

September 13, 2025

Course: DIGT2107 – Fall Term 2025 | **Instructor:** Dr. May Haidar

Contents

1 Introduction

Project Name: OpenBid
Team Number: 1
Team Members: Tyler, Mani, Yanness, Alaister

Document Overview This document follows the university's Iteration 1.1 guidelines for Initial Vision and Planning. It outlines the project vision, high-level features, and a near-term iteration plan aligned with the course schedule. A detailed breakdown of user stories and requirements will be provided in the next iteration submission.

2 Project Vision

Vision Statement

Our project delivers a **web-based local work marketplace** where anyone can post small jobs and nearby providers **bid** to win them. By centering discovery on a map and implementing **KYC for all users**, escrowed payments, and transparent reviews, we help posters get trustworthy help quickly and providers find nearby work efficiently. We will work iteratively and incorporate feedback at each stage.

Problem Statement

Informal local services in the Toronto GTA such as home repairs, moving help, yardwork, and small trades are fragmented across classifieds and social platforms, leading to high search costs, price opacity, and safety concerns. Consumers lack verified identity signals and protected payment flows, while providers face lead uncertainty, travel inefficiency across dispersed neighborhoods, and payment risk. GTA-specific factors-high population density with strong neighborhood heterogeneity, significant travel-time variability, and documented fraud/scam exposure-exacerbate these frictions. A map-first marketplace with mandatory identity verification (KYC), escrowed payments, and on-platform messaging/reviews can reduce search and trust costs, enable fair price discovery, and improve job completion outcomes in the GTA.

Target Users

- **Job Posters**: individuals or small businesses needing short tasks (e.g., repairs, yard work, errands).
- **Job Bidders**: KYC verified individuals who can earn income from small jobs.
- **Job Browser**: anyone can visit the website and browse jobs.

Project Goals

- Ship an intuitive, map-first marketplace for posting, bidding, and completing jobs.
- Increase trust with **mandatory KYC**, Duo-based 2FA, escrowed payments, and reviews.
- Ensure privacy and safety with approximate locations pre-acceptance and phone masking post-acceptance.

- Build on a pragmatic stack: **React + SCSS (Firebase Hosting), Node.js + Express, Firebase Firestore**.

3 High-Level Features

1. Core Marketplace

- Post jobs with photos, budget (fixed or open), category, and date.
- Bid on jobs with amount, note, and ETA; accept/decline; anti-sniping window (optional).
- Escrowed payments (hold → capture on completion; refund/dispute flow).
- Ratings and reviews on completion.

2. In-App Messaging & Negotiation

- Secure real-time chat between posters and providers linked to each job.
- Chat is available from the moment a bid is placed; posters and bidders can negotiate price, clarify scope, and attach images/documents.
- Messages are logged in Firestore under '/messages/messageId' per job, enabling dispute resolution and moderation.
- Notifications (push + SMS proxy fallback) alert users of new messages.
- For safety, personal phone/email contact info is blocked or masked until a job is awarded.

3. Safety, Identity & Trust

- **KYC for all users** (document + selfie) before posting/bidding.
- Duo Security 2FA for login and sensitive actions.
- Neighborhood Safety Score (0–100) informing friction: tips, daylight defaults, verified-only, or manual review.
- Background checks for providers (where permitted; consent required).
- Phone and exact address masking after acceptance; report/block; moderation queue.

4. Maps & Discovery

- Map-based job discovery: clustering, radius filtering, category/budget/date filters.
- Address autocomplete and validation; forward/reverse geocoding.
- Optional time-to-site sorting (distance matrix) in later iterations.

5. Customer Support & Contact

- In-app "Contact Us" form available under profile/settings.
- Support tickets stored in Firestore and routed to admin dashboard.
- Automatic email confirmation sent to user upon submission.
- Options for common categories: billing, identity verification, job disputes, technical issues.
- Escalation path: tickets auto-flagged for urgent review if related to payments or safety.

- Future enhancement: live chat or chatbot triage.

6. Reporting & Analytics

- Basic dashboards: job throughput, bid velocity, completion rates.
- Admin tools: dispute intake, category management, policy/config tuning.

4 Iteration Plan

Long-Term Timeline (Sept 2024 – Mar 30, 2025)

- **Sept – Oct 2024: Project Foundations**
  During this stage, the team will set up the technical and design foundations of the project:
  - Create a shared GitHub repository with branching and pull request workflows.
  - Configure continuous integration and delivery (CI/CD) so code is automatically tested and deployed.
  - Implement user authentication with Firebase Auth and Duo two-factor authentication.
  - Define the Firestore database schema for users, jobs, and bids.
  - Build basic job posting functionality (create, view, edit, delete).
  - Configure SCSS for frontend styling.
  - Produce design wireframes to guide user interface development.
- **Nov – Dec 2024: Requirements & Map-first Features**
  In this stage, the team will move from planning into early feature development:
  - Finalize and document detailed user stories and acceptance criteria.
  - Integrate Google Maps JavaScript API to support map-based job discovery.
    * Implement geocoding to convert addresses into GPS coordinates.
    * Add Places autocomplete for smooth and accurate address entry.
  - Create a placeholder ("stub") version of the Neighborhood Safety Score for design/testing, to be replaced with real data later.
  - Set up the initial messaging/chat system, including database structure and a basic conversation UI, to allow poster and provider communication.
- **Jan 2025: Bidding Loop**
  Build the core workflow for how jobs get matched between posters and providers:
  - Implement bidding: providers can place offers (price, message, and ETA).
  - Allow posters to accept or decline bids, and formally award a job.
  - Add real-time notifications for bid activity and status changes.
  - Integrate escrowed payments with Stripe Connect.
  - Add a simple "Contact Us" form that logs issues to Firestore (basic ticket collection, direct email contact, etc.).
- **Feb 2025: Trust & Compliance**
  Strengthen platform reliability and regulatory compliance:
  - Make "Know Your Customer" (KYC) verification mandatory via Stripe Identity.
  - Integrate provider background checks (Checkr).

  - Build admin-facing moderation tools for disputes/reports.
  - Expand Customer Support system: add ticket status tracking, categories (billing, KYC, disputes, technical), and admin responses.

- **Mar 2025: Polish & Submission**
  Finalize and refine the platform ahead of project submission:
  - Add ratings and reviews for completed jobs to build trust and reputation.
  - Improve the Neighborhood Safety Score logic, using more realistic data and thresholds.
  - Conduct accessibility and mobile usability audits to ensure inclusivity.
  - Perform final end-to-end testing, fix bugs, and complete project documentation.

  Deliverables due: Sunday, 30 March 2025, 11:59 PM.

5 Tech Stack (MVP)

- **Frontend:** React + SCSS, Firebase Hosting.
- **Backend:** Node.js + Express (on Firebase Functions/Cloud Run).
- **Database:** Firebase Firestore (NoSQL, real-time).
- **Storage:** Firebase Cloud Storage for job photos/docs.
- **Auth:** Firebase Auth (OAuth, email+password) + Duo 2FA.
- **Payments:** Stripe Connect (escrow), Stripe Identity (KYC).
- **Maps/Geo:** Google Maps JS, Places, Geocoding, Distance Matrix.
- **Comms:** Twilio SMS + Proxy, Firebase Cloud Messaging.
- **Observability:** Sentry, OpenTelemetry, product analytics via PostHog.

6 High-Level User Stories

Job Posters

- As a **poster**, I want to complete KYC and enable Duo 2FA so I can use the platform securely.
- As a **poster**, I want to create a job with photos, budget options, and a map location so providers can find and bid on it.
- As a **poster**, I want to receive and compare multiple bids, so I can choose the best provider for the job.
- As a **poster**, I want to chat with bidders for clarification and negotiation before accepting one.
- As a **poster**, I want to fund escrow when I accept a provider, so the payment is secured until work is completed.
- As a **poster**, I want to confirm when the job is finished, so the provider is paid automatically from escrow.

- As a **poster**, I want to leave ratings and reviews after the job ends, so I can help build provider reputation.
- As a **poster**, I want to contact customer support if I have disputes or technical problems.

Job Bidders (Providers)

- As a **bidder**, I want to complete KYC verification before bidding so my identity is trusted on the platform.
- As a **bidder**, I want to browse nearby jobs on a map with filters (category, date, budget) so I can find relevant opportunities.
- As a **bidder**, I want to place bids with notes and estimated time of arrival, so posters understand my proposal.
- As a **bidder**, I want to chat securely with posters inside the app to negotiate details or clarify requirements.
- As a **bidder**, I want to receive real-time notifications when a poster responds to my bid or message.
- As a **bidder**, I want to start work once I am awarded a job and see that payment is secured in escrow.
- As a **bidder**, I want to get paid automatically from escrow when the poster confirms completion.
- As a **bidder**, I want to build a reputation with ratings and reviews from past jobs, so I can win future bids more easily.
- As a **bidder**, I want to contact support if I experience fraud, safety issues, or payment disputes.

Job Browsers (Not KYC-Verified Users)

- As a **browser**, I want to explore posted jobs on a map in read-only mode, so I can understand the platform's services before committing.
- As a **browser**, I want to view limited job details (e.g., title, category, approximate location, budget range) without registering, so I see the platform's value.
- As a **browser**, I want to be prompted to create an account if I try to post a job or place a bid, so I know registration is required.
- As a **newly registered user**, I want clear instructions that I must complete KYC verification and enable Duo 2FA before being allowed to post jobs or bid, so I understand the trust policy.
- As a **browser**, I want access to support or FAQ pages, so I can ask questions or learn more before verifying my account.

7 Planning: Task Allocation (Agile Rotation)

Principle: Everyone works across frontend, backend, and Firebase. Pairs rotate weekly in a round-robin so each member pairs with every other member during the term.

Process

- Sprint cadence: 1-week sprints with backlog refinement and sprint review each Friday.
- Daily stand-up: 10 minutes; blockers captured as GitHub issues.
- Rotation: Two pairs per week; cycle repeats every 3 weeks.
- Scrum Master: rotates weekly (Tyler → Mani → Yanness → Alaister, then repeat).
- Quality gates: PR requires one reviewer outside the pair, passing tests, lint compliance.

Appendix: API Inventory

Area	APIs / Notes
Maps	Google Maps JS; Places Autocomplete/Details; Geocoding; Distance Matrix
Auth	Firebase Auth; Duo 2FA; Stripe Identity (KYC)
Payments	Stripe Connect (escrow), Webhooks, Radar
Notifications	Firebase Cloud Messaging; Twilio SMS + Proxy
Storage	Firebase Cloud Storage
Analytics/Ops	PostHog; Sentry; OpenTelemetry

Appendix: Firestore Schema (Indicative)

Listing 1: Firestore Collections & Example Documents

```
/users/{userId}
{
  name: string,
  email: string,
  phone: string,
  avatarUrl: string,
  isProvider: boolean,
  kycStatus: "pending" | "verified" | "failed",
  duoEnabled: boolean,
  createdAt: timestamp,
  updatedAt: timestamp
}


/jobs/{jobId}
{
  posterId: string (ref: users/{userId}),
  title: string,
  description: string,
  category: string,
  budgetType: "fixed" | "open",
  budgetAmount: number,
  location: { lat: number, lng: number, address: string },
  desiredDate: timestamp,
  status: "open" | "awarded" | "in_progress" | "completed" | "cancelled",
  createdAt: timestamp,
  updatedAt: timestamp
}


/bids/{bidId}
{
  jobId: string (ref: jobs/{jobId}),
  providerId: string (ref: users/{userId}),
  amount: number,
  note: string,
  etaHours: number,
  status: "active" | "declined" | "accepted" | "cancelled",
  createdAt: timestamp
}


/messages/{messageId}
{
  jobId: string (ref: jobs/{jobId}),
  senderId: string (ref: users/{userId}),
  body: string,
  attachmentUrl: string,
  createdAt: timestamp
}
```

```
/reviews/{reviewId}
{
  jobId: string (ref: jobs/{jobId}),
  raterId: string,
  rateeId: string,
  rating: number (1-5),
  comment: string,
  createdAt: timestamp
}
```