**Korak 1: Planiranje i dizajn (Najvažniji korak!)**

1. **Definiraj jezgru problema:**
   * Koje su najvažnije vrste problema koje građani mogu prijaviti? (Npr. rupa, smeće, neispravna rasvjeta, oštećena klupa.)
   * Koje informacije moraš prikupiti za svaku prijavu? (Lokacija, opis, fotografija, tip problema, status.)
   * Tko su korisnici? (Građani, djelatnici gradskih službi, administratori.)
2. **Dizajn baze podataka (Schema Design):**
   * Ovo je temelj. Skiciraj tablice: Korisnici, Uloge, PrijaveProblema, TipoviProblema, StatusiProblema, Bilješke.
   * Razmisli o relacijama između tablica. Koje ćeš baze koristiti? **PostgreSQL** je odličan izbor zbog **PostGIS** ekstenzije za geoprostorne podatke, što će ti trebati za lokacije na karti.
3. **API Endpoints (Backend Blueprint):**
   * Koje ćeš API pozive trebati na backendu?
     + POST /api/auth/register (registracija)
     + POST /api/auth/login (prijava)
     + POST /api/problems (kreiranje prijave problema)
     + GET /api/problems (dohvat svih problema)
     + GET /api/problems/{id} (dohvat detalja jednog problema)
     + PUT /api/problems/{id}/status (ažuriranje statusa)
     + POST /api/problems/{id}/notes (dodavanje bilješki)
   * Razmisli o tome koji će endpointi zahtijevati autentifikaciju/autorizaciju.
4. **UI/UX skice (Frontend Blueprint):**
   * Skiciraj glavne ekrane: Početna stranica, stranica za prijavu problema, popis prijava, detalji prijave, administratorski panel.
   * Kako će izgledati interakcija s mapom?

**Korak 2: Postavljanje projekta**

1. **Stvori prazan .NET Core Web API projekt.**
2. **Stvori prazan React projekt.**
3. **Postavi bazu podataka:**
   * Instaliraj **PostgreSQL** (ili odabranu bazu) lokalno.
   * Stvori bazu podataka za svoj projekt.
   * Koristi **Entity Framework Core Migrations** u .NET-u za automatsko kreiranje tablica na temelju tvojih modela (C# klasa).
4. **Instaliraj i konfiguriraj Ollamu:**
   * Preuzmi i instaliraj Ollamu.
   * Preuzmi neki manji, ali sposoban model za testiranje (npr. ollama pull llama3:8b-instruct-q4\_K\_M ili ollama pull mistral).

**Korak 3: Izgradnja temelja (Backend First)**

1. **Postavi Entity Framework Core:**
   * Definiraj svoje modele (C# klase) koji predstavljaju tablice u bazi podataka (User, Problem, ProblemType, Status, itd.).
   * Kreiraj DbContext.
   * Pokreni prvu migraciju i ažuriraj bazu podataka.
2. **Implementiraj autentifikaciju i autorizaciju (ASP.NET Core Identity):**
   * Postavi registraciju i prijavu korisnika.
   * Testiraj da možeš kreirati korisnike i dobiti JWT tokene.
3. **Razvij API Endpointe za Problem resurse:**
   * Kreiraj ProblemController.
   * Implementiraj POST /api/problems (samo osnovne podatke, bez slike i LLM-a za početak).
   * Testiraj s Postmanom/Insomniom.

**Korak 4: Razvoj Frontend sučelja**

1. **Postavi osnovnu React aplikaciju:**
   * Navigacija (React Router).
   * Autentifikacijske stranice (login, register).
2. **Stranica za prijavu problema:**
   * Kreiraj React formu za unos teksta i odabir tipa problema.
   * Sa kartom i upload slike.
   * Poveži s backend API-jem za spremanje problema.
3. **Prikaz problema:**
   * Kreiraj stranicu za prikaz popisa prijavljenih problema.
   * Dohvati podatke s backend API-ja.

**Korak 5: Integracija Mape i Slika**

1. **Backend (Upload slika):**
   * Ažuriraj ProblemController za primanje slika (npr. IFormFile).
   * Implementiraj logiku za spremanje slika (lokalno ili na cloud storage). Spremi putanju do slike u bazu podataka.
2. **Frontend (Prikaz/Upload slika i Mapa):**
   * Dodaj komponentu za upload slika u formu za prijavu problema.
   * Integriraj **Leaflet.js** (ili odabrani map API) u formu za prijavu problema. Omogući korisniku da označi lokaciju na mapi.
   * Prikaži kartu na stranici s popisom problema, sa markerima za svaki problem.

**Korak 6: Integracija LLM-a (Ollama)**

1. **Backend (LLM servis):**
   * Unutar .NET backenda, kreiraj servis (npr. LlmService) koji će komunicirati s Ollamom.
   * Koristi HttpClient za slanje POST zahtjeva na Ollamin API endpoint (obično http://localhost:11434/api/generate).
   * Definiraj promptove za kategorizaciju i ekstrakciju informacija.
   * Parsiraj JSON odgovor iz Ollame.
2. **Povezivanje LLM-a s prijavom problema:**
   * Kada se problem prijavi, pozovi LlmService s opisom problema.
   * Spremi LLM-generirane kategorije i informacije natrag u bazu podataka (u Problem tablicu ili novu tablicu ProblemDetails).
3. **Frontend:** Prikazuj LLM-generirane informacije administratorima radi pregleda i korekcije.

**Korak 7: Praćenje i Administrator panel**

1. **Backend (Ažuriranje statusa i bilješki):**
   * Dodaj API endpointe za ažuriranje statusa problema i dodavanje bilješki.
2. **Frontend (Administrator Dashboard):**
   * Kreiraj administratorski dashboard gdje se problemi mogu filtrirati, sortirati, ažurirati statusi i dodavati bilješke.
   * Prikaži detalje problema, uključujući LLM-generirane informacije.