Week-by-Week Work Plan

Week 1: Core Plugin Setup & Admin Recording

Goal: Make sure the plugin is installable, user login works, and tours can be recorded with screenshots.

Day 1–2: Plugin Bootstrap

- Set up basic folder structure with manifest.json, popup.html, popup.js, content.js, firebase.js
- Configure Firebase (Auth, Firestore, Storage)
- Add Firebase login in popup (Google or email/password)
- Show basic dashboard with "Start Tour" and "Stop Tour" buttons

Day 3–5: Admin Tour Recording (Manual)

- Inject a floating bot (bottom-left sphere icon)
- Start listening to clicks using content.js
- Use html2canvas to capture screenshots on click
- Save:
    - Screenshot → Firebase Storage
    - Metadata: selector, page title, timestamp → Firestore
- Stop button finalizes and uploads data as a new "Tour"

Deliverables by End of Week:

- Working plugin with login
- Admin can record a tour with screenshots and selectors
- Screenshots stored in Firebase Storage
- Metadata stored in Firestore under tours/{tourId}/steps

Week 2: User Playback + Image Matching API

Goal: Let the user search and follow a recorded tour; start AI-based matching support.

Day 6–7: Playback System (DOM Highlighting)

- On user login, show search bar in center
- Show only blurred background until task is chosen
- Load the corresponding tour steps from Firestore
- Show step count, “Next”, “Previous” buttons
- Highlight each element using stored selector
- Wait for:
    - Click on correct element **OR**
    - Click on “Next” button

Day 8–10: AI Matching Integration (Teammate work)

- Create API to receive screenshot → return probable selector
- Use OpenCV or feature matching
- Try matching current html2canvas shot to closest stored image
- Return confidence + step ID

Day 11–12: Integrate Matching into User Flow

- If user is lost (wrong step), use AI to locate nearest matching screen
- Auto-jump to correct step

Deliverables by End of Week:

- User can follow a recorded tour step-by-step
- AI API is functional and integrated
- Selector highlighting is working with fallback from API if needed

Week 3: Final Polishing, Testing & Demo Prep

Goal: Add UX polish, fix bugs, test multiple flows, and get ready to present.

Day 13–14: UI/UX & Polish

- Design better bot icon and animations
- Smooth highlight transitions (use `scrollIntoView`, fade-in)
- Blurring background during playback
- Error handling (missing selectors, login fail, etc.)

Day 15–16: Testing & Multi-Tour Support

- Test:
  - Multiple admins
  - Multiple users
  - Same task but different page structures
- Add feature to replay/edit tour
- Create fallback: show screenshot reference if selector not found

Day 17–18: Final Wrap-Up & Documentation

- Write brief README.md
- Prepare presentation/demo script
- Prepare 2–3 sample tours (e.g., Gmail, sample website)
- Create short video demo (if needed)

Final Deliverables:

- Fully working plugin with:
  - Admin tour recording
  - User playback with highlight
  - AI-based fallback support
- Basic documentation (Firebase config, setup)
- Ready-to-show walkthrough

Work Division

Task	Me (Plugin, Rendering)	Teammate (AI, Backend)
Plugin architecture	✅	
Firebase setup	✅	
Tour recording & storage	✅	
Screenshot capture	✅	
Playback & highlighting	✅	
AI matching API		✅ (Flask/OpenCV/ML Model)
API integration	✅ (call from plugin)	✅ (expose endpoint)
Testing + Debugging	✅	✅

Tools We'll Use

Area	Tool
Plugin development	Chrome Extension + JS
Screenshots	html2canvas
Data storage	Firebase Firestore
Image hosting	Firebase Storage
Auth	Firebase Auth
AI API	Python + Flask (teammate)
Deployment	GitHub (zip for plugin)