FINAL PROJECT PROPOSAL

ISU Campus Explorer

Tuesday, October 28th, 2025

PREPARED FOR

SE/COM S 3190 — Construction of User Interfaces

Iowa State University, Department of Computer Science

Prepared by:

Mekhi San (sanm20@iastate.edu)

&

Ash Bhuiyan (mbhuiyan@iastate.edu)

Table of Contents

1 Introduction

We are extending our Midterm project "ISU Campus Explorer" into a full-stack single-page application (SPA) rebuilt in React and Node/Express with a real database. Our focus remains on helping new students and visitors navigate campus effectively with richer data, saved tours (routes/stops), and administration tools. Mekhi leads UI architecture and interaction design; Ash leads data modeling and API design.

2 Purpose of the proposal

This document outlines the scope, actors, technology stack, CRUD ownership, data model, application flow, error/accessibility strategy, and initial UX for the Final Project. It matches our midterm layout while satisfying Final requirements.

3 Goals and Objectives

Goals (outcomes):

- Deliver a responsive React SPA that quickly surfaces building info and campus navigation.
- Implement a Node/Express backend with a real database (MongoDB or MySQL).
- Enforce RBAC with Admin and User roles; robust validation and error handling.
- Polished, maintainable, accessible UI views and components.

Objectives (milestones):

- Week 1: Repo scaffold (frontend/, backend/, Documents/), React Router shell, two static pages, wireframes.
- Week 2: Express backend with sample routes; +2 pages (total 4); FE↔BE round-trip with seed data.
- Week 3: DB integration; +2 pages (total 6); full CRUD (Admin & User), validations, error handling, a11y; demo prep.

4 Exercising Project Options (Option 1)

Option 1 — extend the Midterm: We will rebuild the UI as React components and add a full backend and database. Work is clearly expanded beyond the Midterm (tours, admin CMS, RBAC, validations).

5 Project Description

ISU Campus Explorer combines mapping, navigation, and visual media to improve campus experience. The app includes a map/list hybrid view, rich building detail, and a tour planner to construct, save, and manage tours.

5.1 Application Flow and Pages (6+ SPA pages)

- Login / Signup (RBAC): JWT sessions; Admin accounts are seeded; signup only for non-admin.
- Home / Map Explorer: Map + list; search by name/department; tag filters; “Open now” toggle.
- Building Detail: Description, departments, hours, images/video; Add to Tour.
- Tour Planner (multi-step): Select buildings → Order → Save; drag-to-reorder; route preview.
- Confirmation: Success summary when saving a tour or updating profile.
- About / Team: Course, names, ISU emails (with photos).
- Admin CMS (protected): CRUD buildings, media, hours, tags, featured lists.

5.2 Actors & Roles (RBAC)

- Admin: Full CRUD on buildings, media, tags, hours, featured lists, tours.
- User: Browse/search; create/update/delete their tours; edit profile.
- Auth: Login for all roles; Admin seeded only; Signup for non-admin users. Routes gated in UI and server.

5.3 CRUD & Partner Responsibilities

Each feature is owned end-to-end (frontend + backend) by one partner:

Partner 1 — Mekhi

- Login & Signup (FE+BE): forms, validation, JWT auth endpoints, session handling.
- Admin CMS: Create/Delete (Admin POST/DELETE for buildings/media/tags/hours).
- Tours (User GET/PUT): fetch own tours; update steps/order with validation.
- Error handling: 401/403/422; inline messages and retry flows.

Partner 2 — Ash

- Home / Map Explorer (GET/PUT Admin): list/search view; admin update endpoints.
- Building Detail (User POST/DELETE): add/remove building in a tour; delete own tour.
- Confirmation & About: finalization of UX and course/team page.
- Error handling: 404/500; form validation; global toasts/modals.

5.4 Data & API (React ↔ Express ↔ DB)

- Building {id, name, code, coords: {lat,lng}, departments [], hours{mon–sun}, tags[], images [], videos []}
- Tour {id, userId, title, stops: [buildingId], createdAt, updatedAt }
- User {id, name, email, role}
- REST endpoints with validation and consistent JSON error shapes.

5.5 Error States & Accessibility

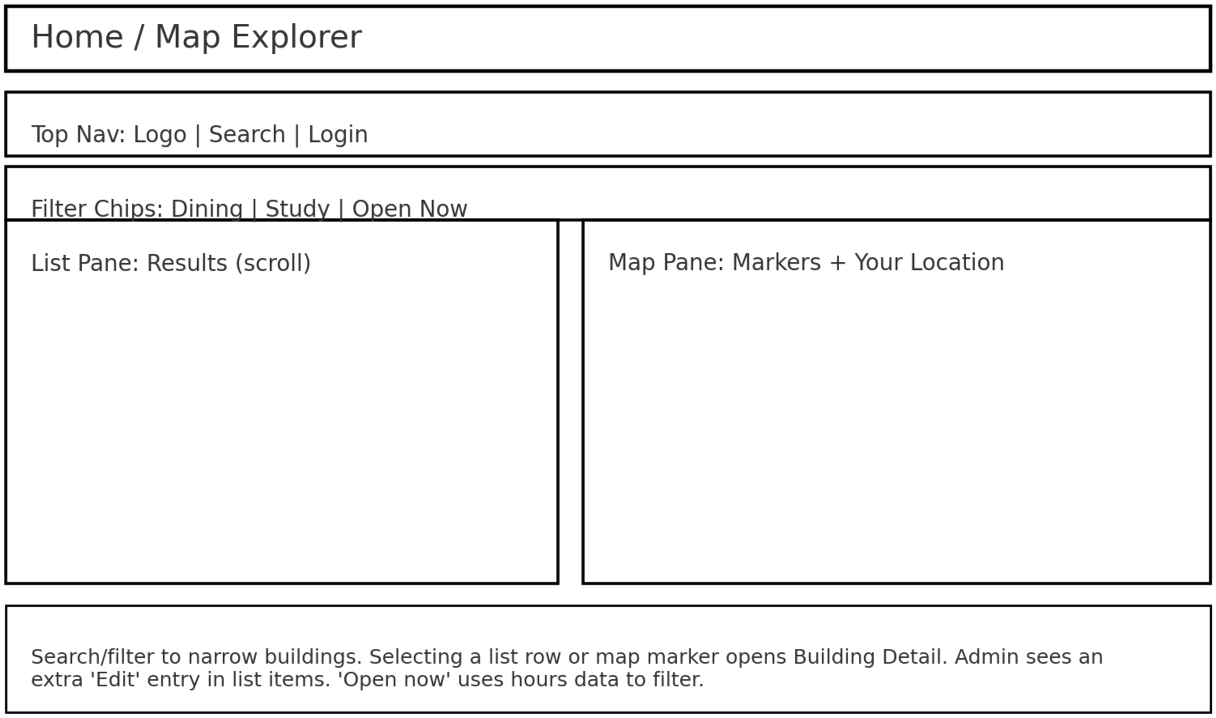
- Inline validation and descriptive errors; retry/undo where sensible.
- Responsive layouts; semantic HTML; focus styles and contrast checks.
- Alt text on media; empty states with guidance; loading skeletons/spinners.

6 Resources & Tools

- Frontend: React (functional components), React Router, Hooks.
- Backend: Node.js + Express; JWT auth; validation via Zod or express-validator.
- Database: MongoDB (Mongoose) or MySQL (Prisma/Sequelize).
- Map/UX: Leaflet or Google Maps; optional UI lib (Headless UI/Radix).
- Design: Figma/Excalidraw; Lighthouse/axe-core for accessibility checks.
- DevOps: GitLab mono-repo with frontend/, backend/, Documents/; issues/branches/MRs.
- Optional deploy: Vercel/Render/Netlify (not required).

7 User Experience Wireframes

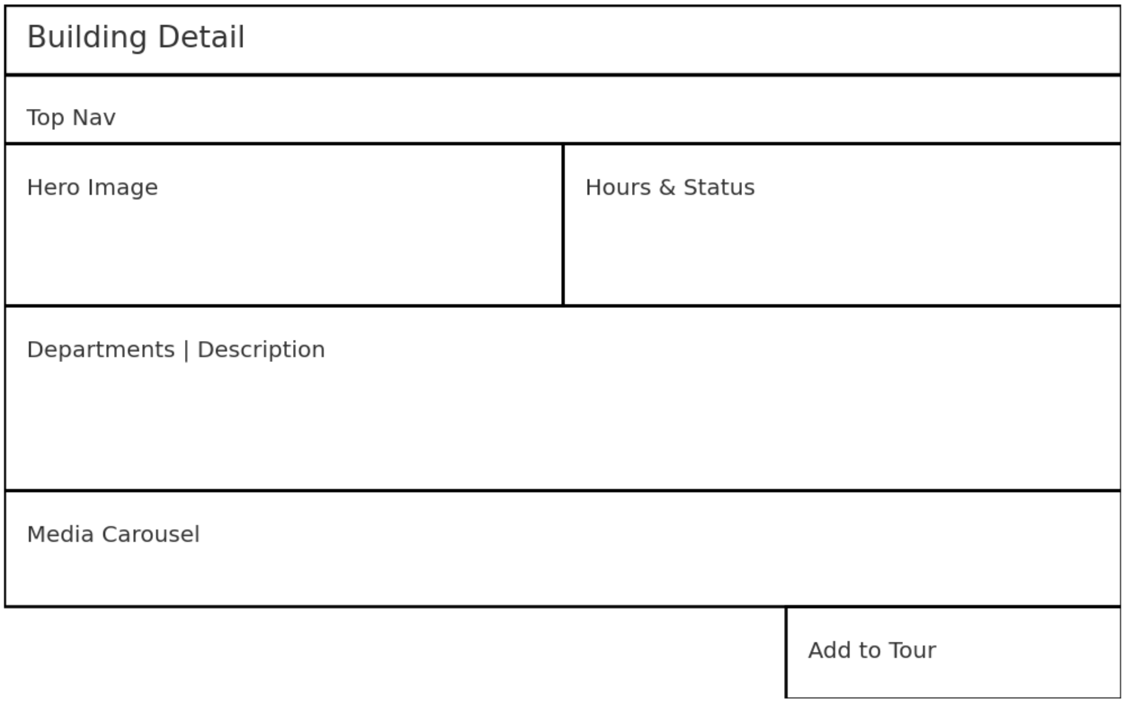
Figure 1. Home / Map Explorer


Home / Map Explorer
Top Nav: Logo | Search | Login
Filter Chips: Dining | Study | Open Now
List Pane: Results (scroll)
Map Pane: Markers + Your Location

Search/filter to narrow buildings. Selecting a list row or map marker opens Building Detail. Admin sees an extra 'Edit' entry in list items. 'Open now' uses hours data to filter.

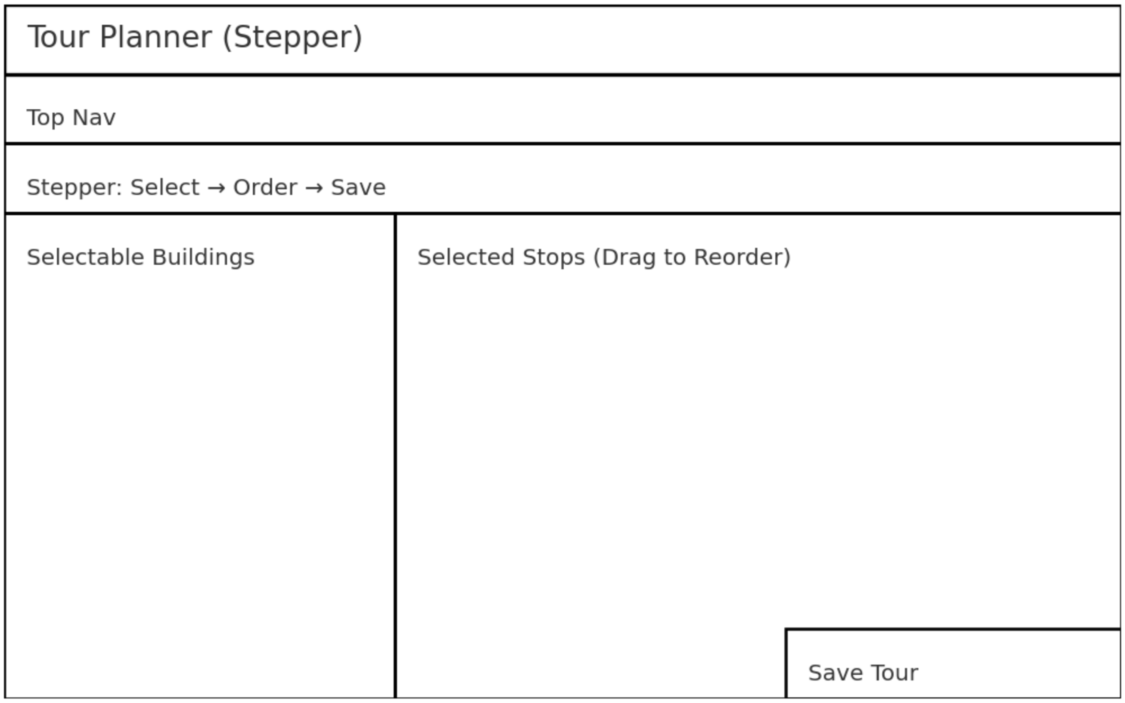
Map/list hybrid with search and filter chips. Selecting a list row or map marker opens Building Detail. Supports 'Open now' filter using hours data.

Figure 2. Building Detail



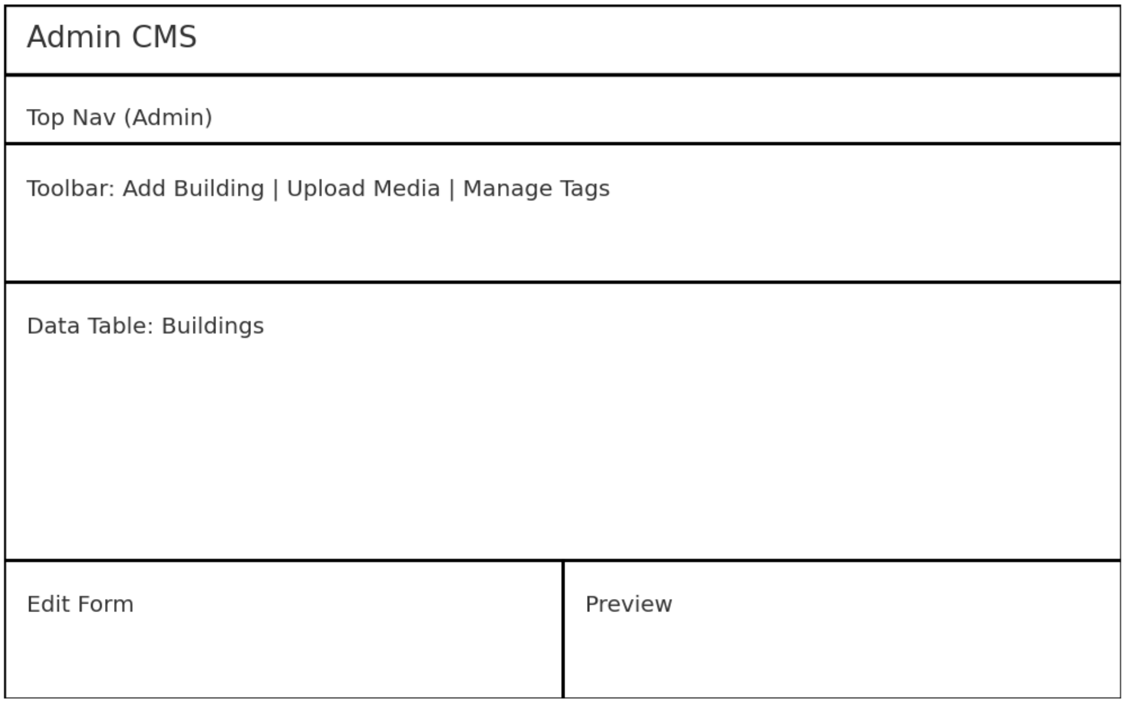
Overview with hours, departments, tags, and media. Primary CTA adds building to the current user's draft tour.

Figure 3. Tour Planner (Stepper)


Tour Planner (Stepper)
Top Nav
Stepper: Select → Order → Save
Selectable Buildings
Selected Stops (Drag to Reorder)
Save Tour

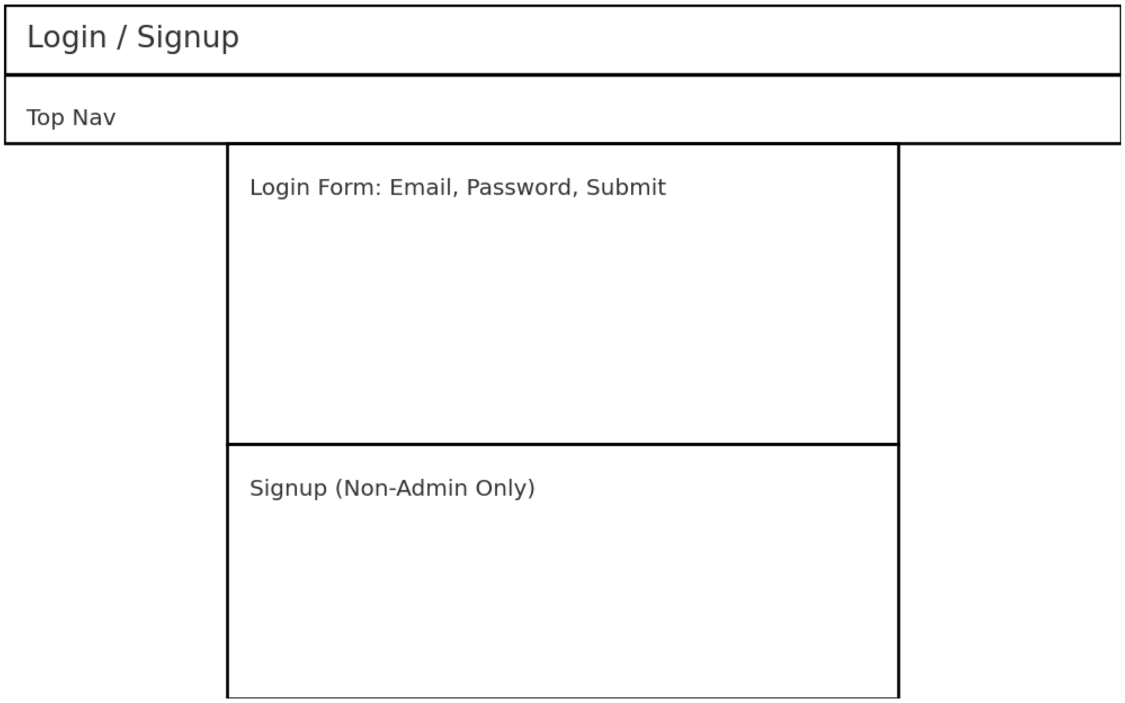
The Stepper guides selection and ordering. Drag-and-drop changes stop order. Save persists Tour {userId, title, stops []} to DB.

Figure 4. Admin CMS



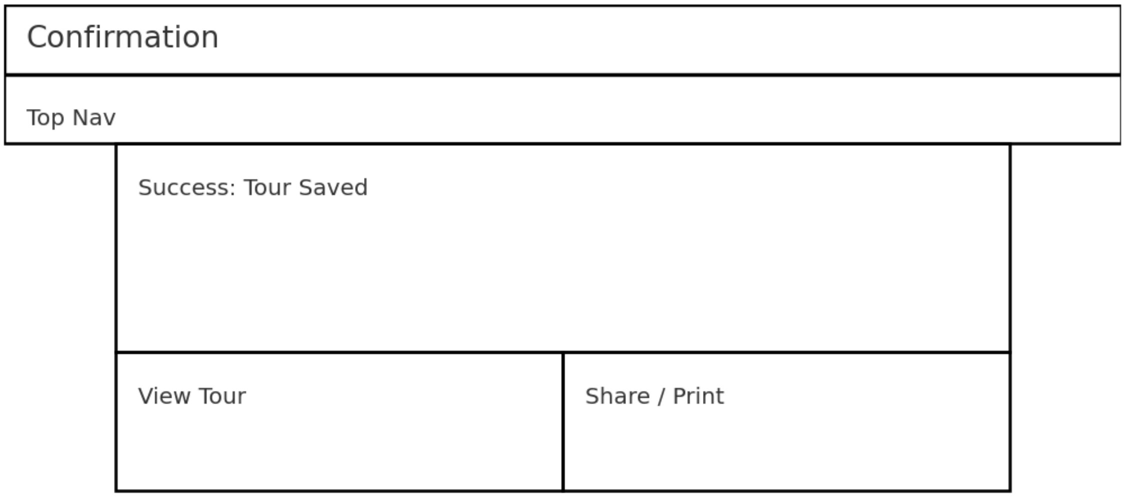
Admin-only table and forms to CRUD buildings, tags, hours, and media with validation and preview.

Figure 5. Login / Signup



Auth views. Admin is seeded; users sign up. JWT/session stored securely; client routes gated by role.

Figure 6. Confirmation / Success



Success screen with actions to view the saved tour, share, or print a summary.

8 Future Work

- AI Building Finder: Natural Language queries (e.g., “quiet study spaces near Parks Library”).
- AI Tour Optimizer: efficient order suggestions and time estimates.
- Indoor navigation and accessibility routing (elevators, ramps) if time permits.

9 Final Comments

We will rebuild the midterm in React + Node with a real database and expanded functionality. This document mirrors the midterm layout while meeting Final requirements. Primary contacts: Mekhi San (sanm20@iastate.edu) and Ash Bhuiyan (mbhuiyan@iastate.edu).