Project Manthan: The Definitive MVP Implementation Blueprint

Section 1: The Strategic Foundation: De-Risking the Manthan MVP

The strategic pivot towards a "Managed Marketplace" MVP is a decisive move to de-risk Project Manthan by prioritizing speed-to-revenue and market validation.[1] This initial phase, termed the "Accelerated Path," focuses on established creators with existing scripts, leveraging a "human-in-the-loop" service model to solve the critical marketplace "cold start" problem.[1] This section provides a granular analysis of the member journey for this first strategic wedge, identifies critical operational blindspots, and proposes design solutions that transform these challenges into defensible strategic niches.

1.1. Analyzing the "Accelerated Path" Member Journey

The MVP workflow is not a self-serve platform but a high-touch, founder-led service designed to close deals.[1] The journey must be analyzed from the perspective of all three key participants: the Creator, the Founder, and the Buyer.

The Creator's Perspective

The journey for a creator within the "Founding Cohort" is designed to be one of high-value service with minimal friction.

1. **Invitation & Onboarding:** The journey begins with a personal invitation from the founder. Upon acceptance, the creator is directed to a private, secure sign-up page.

2. **Secure Sign-Up:** The creator registers for an account, providing basic information. During this process, they are presented with an explicit consent modal detailing how their intellectual property will be handled, establishing a foundation of trust from the first interaction.
3. **Project Ingestion:** Once onboarded, the creator accesses a simple dashboard where they can initiate a new project. They upload their script (in PDF or text format) and provide essential metadata such as the project title and a brief logline.
4. **AI-Assisted Packaging:** The platform's AI engine processes the script to generate a suite of professional pitch materials.[1, 1] The creator is notified once the initial package is ready for review.
5. **Review & Collaboration:** The creator can view the AI-generated assets (e.g., pitch deck, series outline). The primary interaction at this stage is collaborative, involving direct communication with the founder to refine the pitch and strategy.
6. **Strategic Alignment:** The creator works with the founder to finalize the pitch materials and develop a curated target list of potential buyers (studios, OTT platforms, producers).[1]
7. **Passive Monitoring:** The creator's active role pauses as the founder takes the lead on matchmaking. They are kept informed of progress and any expressions of interest.

The Founder's Perspective

The founder is the central node of the managed marketplace. Their workflow is the most complex and is the primary focus of the MVP's internal tooling.

1. **Cohort Curation:** The founder manually identifies and recruits the "Founding Cohort" of 20-30 established creators and 5-10 friendly industry executives.[1]
2. **Project Oversight:** From a private administrative dashboard, the founder monitors all incoming projects. They can view uploaded scripts, review the quality of AI-generated assets, and track the status of each project.
3. **Curation & Refinement:** The founder acts as the "human-in-the-loop," reviewing and refining the AI-generated pitch packages to ensure they meet industry standards and align with market trends.[1] This adds a crucial layer of expert validation.
4. **Market Intelligence Gathering:** Through ongoing conversations with buyers, the founder logs proprietary market intelligence—what platforms are looking for, genre trends, and specific content mandates—into a dedicated internal database.
5. **Strategic Matchmaking:** Armed with high-quality packages and market intelligence, the founder personally facilitates introductions between creators' projects and the most relevant buyers, leveraging the platform's internal CRM features to manage outreach and track communications.[1]
6. **Deal Flow Management:** The founder manages the entire deal pipeline, from initial introduction to tracking feedback and facilitating next steps when interest is shown.

7. **Transaction Facilitation:** Upon a successful match, the founder provides legally-vetted, India-specific contract templates and guides the parties through a secure escrow system for deal closure and payment management.[1]

The Buyer's Perspective

The buyer's journey is engineered for maximum convenience and value, eliminating the need for them to learn a new platform.

1. **Personalized Introduction:** A buyer receives a direct, personalized email from the founder.
2. **High-Value Package:** The email contains a link to a professionally packaged, data-rich pitch deck for a project that has been specifically curated for their interests and current content needs.
3. **Frictionless Engagement:** The buyer reviews the materials at their convenience. If interested, they simply reply to the founder to initiate a conversation or request a meeting with the creator. They are not required to sign up, log in, or navigate a marketplace interface. This solves the buyer-side of the "cold start" problem by delivering value directly to their inbox.[1]

1.2. Identifying Critical Blindspots and Strategic Niches

A surface-level implementation of the above journey would be inefficient and miss key opportunities to build a long-term competitive advantage. The architecture must address the following from day one.

The Founder's Command Center is the *Real* MVP

The "human-in-the-loop" model is the core of the Phase 1 strategy, making the founder's efficiency the primary bottleneck to success.[1] Attempting to manage this process with disparate tools like spreadsheets, email clients, and file folders is not only inefficient but also unscalable and detrimental to the long-term vision. For a cohort of 20 creators, each with one project targeting five buyers, the founder must manage up to 100 distinct outreach threads, each with its own status, feedback logs, and asset versions.[1] Without a centralized system,

this complexity becomes unmanageable, critical data for training the future AI matchmaking agent is lost, and the MVP success metric of closing 3-5 deals becomes significantly harder to achieve.[1]

Therefore, the first and most critical component to be built is a secure, private, founder-only administrative dashboard. This "Command Center" is not an auxiliary feature; it is the central nervous system of the managed marketplace. It must provide a unified view of all creators, projects, AI-generated assets, buyer interactions, and market intelligence. Building this internal tool first ensures the founder can operate at maximum efficiency, deliver a high-quality service to the founding cohort, and systematically capture the proprietary data that will fuel the platform's future automation.

Trust is a Feature, Not a Given

Creators are being asked to upload their most valuable asset—their intellectual property—to a new, unproven platform. Establishing trust is paramount and must be woven into the product's design. While legal compliance is a necessity, it can be strategically leveraged as a powerful trust-building mechanism. India's Digital Personal Data Protection (DPDP) Act, 2023, mandates that consent must be free, specific, informed, and unambiguous.[2] A standard, boilerplate "I agree to the terms and conditions" checkbox is legally insufficient and does nothing to assuage creator anxiety.[4]

To address this, the user onboarding flow will feature a prominent "Creator's Bill of Rights" consent modal. This is a UX element designed to transform a legal requirement into a statement of values. Before a user can create an account, they must actively acknowledge a series of clear, plain-language statements, such as:

- "I understand my uploaded script will be used *only* for the purpose of generating my project's pitch materials."
- "I understand my script and personal data will *never* be shared with third parties without my explicit, case-by-case permission."
- "I understand my intellectual property will *not* be used to train any public or third-party AI models."

This approach directly confronts the primary fears a creator would have, builds a moat of trust that larger, more opaque platforms cannot easily replicate, and demonstrates a creator-first ethos from the very first click.

Hyper-Verticalization Requires a Proprietary "Mandate" Database

The core defensible moat for Manthan is "Hyper-Verticalization"—an intimate, data-driven understanding of the Indian Media & Entertainment industry's specific needs.[1] This strategy is strongly validated by market trends showing an explosion in demand for regional and platform-specific content.[5] The planned "Format Adaptation Engine" is designed to capitalize on this by tailoring a script for different targets like OTT series or YouTube pilots.[1]

However, the efficacy of this engine depends entirely on the quality of its guiding data. A generic AI model does not know that Netflix India's 2025 slate is focusing on high-stakes dramas and South Indian originals [8], or that Amazon Prime Video is looking for stories with broad emotional resonance.[10] This information is proprietary. It is gathered through the founder's direct interactions with buyers, who might state, "We are actively seeking a female-led thriller for the Tamil market." This specific, timely market intelligence is the true "data flywheel" that must be captured.[1]

Consequently, the Founder's Command Center must include a dedicated feature for logging and tagging these "Platform Mandates." This internal, proprietary database will be the founder's secret weapon. When the AI Packaging and Adaptation agents are run, the prompts will be dynamically enriched with relevant, up-to-date mandates from this database. This will make the AI's output exponentially more valuable, relevant, and defensible than any competitor relying on publicly available data, truly embedding Manthan into the hyper-local transactional workflow of the Indian M&E industry.

Section 2: The Architectural Blueprint: Vercel, Supabase, and Claude

The technology stack and architecture are selected to maximize development speed, minimize operational overhead, and align with the lean, bootstrapped budget of a non-technical solo founder.[1, 1] The combination of Vercel for hosting and serverless functions, and Supabase for backend services, provides a powerful, scalable, and cost-effective foundation for the MVP.[1]

2.1. System Architecture Diagram

The system is designed as a modern web application with a decoupled frontend and backend, orchestrated by serverless functions.

1. **User Interaction:** The Creator and Founder access the application via a web browser. The user interface is a Next.js application.
2. **Frontend Hosting (Vercel):** The Next.js frontend is deployed on Vercel's global edge network. This provides fast load times, automatic scaling, and seamless continuous deployment directly from a GitHub repository.[1]
3. **Backend Logic (Vercel Serverless Functions):** All backend API endpoints are implemented as Python Serverless Functions, also hosted by Vercel. When the frontend needs to perform an action like running the AI packaging agent, it makes an API call to a Vercel function endpoint.[11]
4. **Backend-as-a-Service (Supabase):** The Vercel functions and the Next.js frontend interact with a suite of Supabase services:
    - **Supabase Auth:** Handles all user sign-up, login, and session management. It provides the security layer for the application.[13]
    - **Supabase Database (PostgreSQL):** A managed PostgreSQL database stores all application data, including user profiles, project details, AI-generated assets, and the founder's proprietary market intelligence. Row-Level Security (RLS) policies are applied directly in the database to enforce data access rules.[1]
    - **Supabase Storage:** Manages the storage of large files, specifically the creators' uploaded scripts and the AI-generated pitch decks. This is separate from the database to handle binary files efficiently.[15]
5. **AI Engine (Anthropic API):** The Python serverless functions make secure, server-to-server API calls to Anthropic's Claude 3 Opus model to perform the script analysis and content generation tasks.

This architecture allows the founder to manage a single GitHub repository, and Vercel handles the entire deployment and infrastructure complexity for both the frontend and backend logic, while Supabase provides a robust, managed backend with minimal configuration.

2.2. Technology Stack Breakdown

- **Frontend: Next.js (React)** deployed on **Vercel**.
    - **Rationale:** Vercel is created by the makers of Next.js, offering a zero-configuration deployment experience. Its direct GitHub integration means any code pushed to the repository is automatically deployed, a perfect workflow for a non-technical founder working with an AI assistant. The generous free "Hobby" tier is sufficient for the entire MVP phase.[1]
- **Backend Logic: Python (Flask)** Serverless Functions on **Vercel**.

    - **Rationale:** Python is the industry standard for data science and AI, with extensive libraries for text processing and interacting with LLM APIs. Vercel's serverless function model is highly cost-effective, as you only pay for compute time when an API endpoint is actively being used, aligning perfectly with the lean Year 1 budget.[1]
- **Database & BaaS: Supabase**.
    - **Rationale:** Supabase provides a complete backend-in-a-box. It offers a standard PostgreSQL database without the need for manual setup or management. Crucially, it also includes pre-built solutions for user authentication and file storage, which are essential features for Manthan. This dramatically reduces the amount of custom backend code that needs to be written, saving significant development time and cost. Its free tier is also robust enough for the MVP launch.[1]
- **AI Engine: Anthropic's Claude 3 Opus API**.
    - **Rationale:** For the core task of analyzing full-length scripts, a large context window is non-negotiable. Claude 3 Opus offers a 200K token context window, allowing it to process an entire screenplay in a single pass, ensuring high contextual understanding. It excels at creative writing, summarization, and structured data extraction, making it the ideal choice for the multi-faceted "Packaging Agent".[17]

2.3. Supabase Data Model & Security

The database schema is designed not only to support the immediate needs of the MVP but also to proactively capture the necessary data for training future AI models, such as the "Success Prediction Score" and the automated "Matchmaking Agent".[1, 1] Research into movie success prediction consistently highlights features like genre, budget, and key personnel as significant predictors.[19] By including nullable fields for this data from the outset, the platform begins building its most valuable long-term asset—a proprietary, structured dataset of Indian creative projects and their market outcomes.

Table Name	Column Name	Data Type	Constraints / Description
profiles	id	uuid	Primary Key, Foreign Key to auth.users.id.
	full_name	text	User's full name.

	role	text	User role ('creator' or 'founder'). Defaults to 'creator'.
	created_at	timestamptz	Default now().
projects	id	uuid	Primary Key, Default gen_random_uuid().
	owner_id	uuid	Foreign Key to profiles.id.
	title	text	Not Null.
	status	text	'draft', 'submitted', 'in_review', 'active'. Default 'draft'.
	logline	text	
	synopsis	text	
	genre	text	Array of genres (e.g., {'Thriller', 'Drama'}). For future ML.
	character_breakdowns	jsonb	Stores structured character data from AI analysis.
	budget_range	text	e.g., 'Below 1 Cr', '1-5 Cr'. For future ML.
	target_platforms	text	e.g., {'Netflix',

			'YouTube'}.
	created_at	timestamptz	Default now().
script_uploads	id	uuid	Primary Key, Default gen_random_uuid().
	project_id	uuid	Foreign Key to projects.id.
	file_path	text	Path to the file in Supabase Storage. Not Null.
	file_name	text	Original name of the uploaded file.
	file_size	bigint	File size in bytes.
	uploaded_at	timestamptz	Default now().
generated_assets	id	uuid	Primary Key, Default gen_random_uuid().
	project_id	uuid	Foreign Key to projects.id.
	asset_type	text	'pitch_deck', 'series_outline', 'character_bible'.
	asset_url	text	Path to the generated file in Supabase Storage.

	version	integer	Version number of the asset. Default 1.
	created_at	timestamptz	Default now().
platform_mandates	id	uuid	Primary Key, Default gen_random_uuid().
	platform_name	text	e.g., 'Netflix', 'SonyLIV'.
	mandate_description	text	The core market intelligence. Not Null.
	tags	text	Searchable tags (e.g., {'thriller', 'female-led', 'tamil'}).
	source	text	How the intelligence was obtained (e.g., 'Conversation with Exec A').
	created_by	uuid	Foreign Key to founder's profiles.id.
	created_at	timestamptz	Default now().
deal_pipeline	id	uuid	Primary Key, Default gen_random_uuid().

	project_id	uuid	Foreign Key to projects.id.
	target_buyer_name	text	Name of the buyer/studio being pitched.
	status	text	'introduced', 'passed', 'in_discussion', 'deal_closed'.
	feedback_notes	text	Logs feedback from the buyer. Critical data for future ML.
	updated_at	timestamptz	Default now().

Row-Level Security (RLS) Policies

RLS is a critical security feature of Supabase that will be enabled on all tables.

- **profiles Table:** Users can only view and update their own profile.
- **projects Table:** A user can view/update/delete a project only if their id matches the owner_id of the project. The founder (with role = 'founder') will have full access to all projects.
- **platform_mandates & deal_pipeline:** Access will be restricted exclusively to the founder role. Creators will have no visibility into these tables.

2.4. API Endpoint Definitions (Vercel Serverless Functions)

These endpoints define the contract between the Next.js frontend and the Python backend logic. They will be created in an /api directory in the project root, which Vercel automatically

maps to serverless functions.[12]

Method	Endpoint	Description	Role Access
POST	/api/projects/request-upload-url	**[Creator]** Takes fileName and fileType in the request body. Generates and returns a secure, time-limited signed URL for direct client-side upload to Supabase Storage. This avoids hitting Vercel function payload limits.[15]	Creator
POST	/api/projects/finalize-upload	**[Creator]** Takes projectId and the filePath from the successful Supabase Storage upload. Updates the script_uploads table and triggers the AI packaging agent as a background task.	Creator
POST	/api/projects/run-packaging-agent	**[Internal]** An asynchronous function that performs the core AI processing. It fetches the script, executes the prompt chain with the Claude API,	Server-side only

		generates the pitch assets, and saves them to storage. This is triggered by the finalize-upload endpoint.	
GET	/api/founder/dashboard	**[Founder]** A secure endpoint that fetches all data required for the Founder's Command Center: all projects, all platform mandates, and all deal pipeline entries.	Founder
POST	/api/founder/mandates	**[Founder]** Allows the founder to create a new platform mandate entry.	Founder
PUT	/api/founder/mandates/{id}	**[Founder]** Allows the founder to update an existing platform mandate.	Founder
POST	/api/founder/pipeline	**[Founder]** Allows the founder to create or update an entry in the deal_pipeline table for a specific project.	Founder

Section 3: The Intelligence Engine: AI Strategy for the

Packaging Agent

The "Creator Intelligence Engine" is the core value proposition of the Manthan platform.[1] For the MVP, this engine is embodied by the "Packaging Agent," which transforms a raw script into a professional, market-ready pitch package. The strategy prioritizes immediate value delivery through sophisticated prompt engineering with a state-of-the-art commercial LLM, while systematically collecting the data required for future, proprietary models.

3.1. LLM Selection and Rationale

For the MVP, the AI engine will be powered exclusively by **Anthropic's Claude 3 Opus API**. This choice is strategic and deliberate for several reasons:

1. **Massive Context Window:** Screenplays are long documents. Claude 3 Opus's 200K token context window is a decisive advantage, capable of ingesting an entire 120-page script in a single prompt. This allows the model to maintain a holistic understanding of plot, character arcs, and thematic consistency, which is crucial for generating high-quality, coherent analysis and summaries.
2. **Advanced Reasoning and Creativity:** The tasks required—extracting nuanced themes, analyzing character motivations, and creatively adapting a story for different formats—demand more than simple text summarization. Claude 3 has demonstrated state-of-the-art performance in complex reasoning and creative generation tasks, making it well-suited for producing the high-quality pitch materials that will impress industry executives.[17]
3. **Speed-to-Market:** Leveraging a powerful, general-purpose model via its API allows the MVP to deliver sophisticated AI features without the immense cost, time, and data requirements of training or fine-tuning a custom model. The focus for Phase 1 is on prompt engineering, which provides the best balance of performance and development velocity.

3.2. The "Packaging Agent" - A Prompt Chaining Approach

To ensure reliability, modularity, and high-quality output, the Packaging Agent will not use a single, monolithic prompt. Instead, it will execute a "prompt chain"—a sequence of focused, interdependent API calls to Claude. This approach deconstructs the complex packaging task

into manageable steps, improves the accuracy of each output, and allows for easier debugging and future refinement.

Step	Prompt Name	Input(s)	AI Task & Output
1	**Structural Analysis & Pre-processing**	Uploaded script file (PDF/text).	**(Non-LLM Task)** The Python backend first parses the script text to identify standard screenplay elements (scene headings, character names, dialogue) based on industry formatting conventions.[22] This structured text is then passed to the LLM.
2	**Core Elements Extraction**	Full script text.	**Task:** "Read the entire screenplay provided. Analyze its structure, plot, and characters. Return a JSON object with the following keys: logline (a compelling 1-2 sentence summary), one_page_synopsis (a detailed summary of the plot), main_themes (an array of key themes), and main_characters

			(an array of objects, each with a name and a brief_description)."
3	**Character Bible Generation**	Full script text, main_characters list from Step 2.	**Task:** (Run in a loop for each main character) "Given the full screenplay and the character '[Character Name]', generate a detailed 'Character Bible' entry. Describe their primary motivation, internal and external conflicts, key relationships, and their complete character arc from the beginning to the end of the story."
4	**Format Adaptation**	one_page_synopsis, main_themes, target_platform (e.g., "Netflix India"), relevant entries from the platform_mandates database.	**Task:** "You are a development executive. Based on the provided synopsis and themes, adapt this story for. The platform's current mandate is: ''. Generate a platform_specific_adaptation in the form of a 10-episode series outline, including a pilot summary, key

			plot points for each episode, a mid-season twist, and a season finale cliffhanger. Emphasize the [thriller/dramatic/comedic] elements that align with the platform's mandate."
5	**Pitch Deck Content Generation**	All outputs from Steps 2, 3, and 4. Creator's bio from profiles table.	**Task:** "Synthesize all the provided information to generate the complete text content for a professional pitch deck. The output should be a single text document with clearly marked sections: Title Page, Logline, Synopsis, Why This Story Now?, Character Introductions (using the Character Bible entries), Season 1 Outline (using the adaptation), and Creator's Bio."
6	**Document Assembly**	Text output from Step 5.	**(Non-LLM Task)** The Python backend uses a library like python-docx or

			reportlab to assemble the generated text into a formatted DOCX or PDF document, which is then saved to Supabase Storage. Visuals are a manual step for the founder in the MVP.

3.3. Data Strategy for the Future "Success Prediction Score"

While the MVP will not feature an automated "Success Prediction Score," its architecture is explicitly designed to collect the high-quality, labeled data required to build one in the future.[1] The strategy will evolve from a simple heuristic model to a more sophisticated machine learning system.

Phase 1.5: Heuristic Model Development

Once the first 20-30 projects have been processed and pitched, the founder will have the initial dataset needed to build a simple, rule-based heuristic model. This serves as a precursor to a full ML model and can provide immediate value.

- **Data Collection:** The founder will manually score each project (on a 1-10 scale) across several key dimensions informed by industry knowledge and success prediction research[24]:
  - Story_Strength: Subjective assessment of the script's quality, originality, and narrative power.
  - Market_Fit: How well the project aligns with known platform_mandates.
  - Creator_Track_Record: Based on the creator's previous work and industry standing.
  - Budget_Feasibility: An assessment of whether the project can be produced within a realistic budget range.
- **Heuristic Logic:** A simple weighted average of these scores can produce an initial "Manthan Score." For example: Score = (0.4 * Story_Strength) + (0.3 * Market_Fit) + (0.2 * Creator_Track_Record) + (0.1 * Budget_Feasibility). The weights can be adjusted as more

data on which projects get deals becomes available.

Phase 2: Machine Learning Model Training

The true long-term value lies in the proprietary dataset being built from day one. The database schema is designed to capture the features and labels needed to train a robust prediction model.

- **Feature Set (The Predictors):**
  - **Project Features:** Structured data from the projects table, including genre, budget_range, and AI-extracted features like main_themes and character archetypes.
  - **Creator Features:** Data associated with the creator's profile, such as a manually entered "experience level" or number of past projects.
  - **Market Signals:** Initially, this will be based on the number and quality of matching platform_mandates. In later phases, this could be augmented with external data like social media buzz or trailer views.[19]
- **Target Variable (The Label):**
  - The outcome data from the deal_pipeline table will be used to create the label. This can be a binary classification (deal_closed = 1, passed = 0) or a regression task to predict the deal_value.
- **Model Selection:** Once a sufficient dataset (e.g., 100+ projects with definitive outcomes) is collected, standard machine learning models like Logistic Regression, Random Forest, or Gradient Boosting can be trained to predict the likelihood of a project's success based on its features.[27] This data-driven "Success Prediction Score" will become a core part of the platform's intelligence layer.

Section 4: Building the Bridge: A Cloud-Only Implementation Guide

This section provides a chronological, step-by-step guide for constructing the Manthan MVP using an entirely online, browser-based workflow. This approach eliminates the need for a local development machine, leveraging GitHub, Vercel, Supabase, and a cloud-based IDE (like GitHub Codespaces) for all development and deployment activities.

Phase 0: Project Scaffolding & Foundation (The First Hour)

This phase uses a highly automated "one-click" process to establish the foundational infrastructure for the entire project.

- **Step 0.1: Create Accounts:** In your web browser, sign up for free accounts on GitHub, Vercel, and Supabase. Use your GitHub account to sign up for the other two services to ensure seamless integration.
- **Step 0.2: Deploy the Vercel + Supabase Template:**
  1. Navigate to the official Vercel + Supabase Next.js starter template page.[29]
  2. Click the "Deploy" button. This will initiate a setup wizard on Vercel.
  3. **Create Git Repository:** Vercel will prompt you to create a new GitHub repository. Give it a name like manthan-mvp and click "Create". This automatically copies the template code into your new repository.
  4. **Integrate Supabase:** The wizard will then guide you to connect your Supabase account. You will be prompted to create a new Supabase project. Follow the on-screen instructions.
  5. **Automatic Configuration:** The Vercel-Supabase integration will automatically create the necessary database tables and, most importantly, will automatically add all required environment variables (like NEXT_PUBLIC_SUPABASE_URL and SUPABASE_SERVICE_ROLE_KEY) to your Vercel project.[29] This eliminates the most common and error-prone setup step.
  6. **Deploy:** Once the integration is complete, Vercel will automatically build and deploy your new project. You will be given a public URL (e.g., manthan-mvp.vercel.app).
- **Step 0.3: Launch Cloud Development Environment:**
  1. In your browser, navigate to the manthan-mvp repository that was just created on GitHub.
  2. Click the <> Code button, select the "Codespaces" tab, and click "Create codespace on main".
  3. GitHub will prepare a complete, browser-based VS Code development environment with your project's code already loaded. This is your new "local machine" in the cloud.[31]
- **Step 0.4: Start the Development Server:**
  1. Once the Codespace loads, a terminal will be available at the bottom of the screen.
  2. In the terminal, run the command npm install to ensure all dependencies are installed.
  3. Then, run npm run dev.
  4. GitHub Codespaces will detect the running application and show a pop-up to open the URL in a new browser tab. This gives you a private development URL to test your changes before pushing them live.

The Core Development Workflow

For all subsequent phases, you will follow this simple, repeatable, cloud-only workflow:

1. **Develop:** Open your project in GitHub Codespaces.
2. **Code:** Use the browser-based VS Code editor and the integrated terminal to write and test code, using the prompts in Section 5 to guide your AI assistant.
3. **Test:** Use the private development URL provided by Codespaces to test your changes in real-time.
4. **Commit & Push:** Once you are satisfied with your changes, use the source control panel in Codespaces (or the terminal) to commit and push your code to the GitHub repository.
5. **Deploy:** Vercel will automatically detect the push and start a new deployment.[32]
6. **Verify:** Check the public Vercel URL to confirm your changes are live.

Phase 1: User Authentication & Profiles (Creator & Founder Roles)

This phase implements the core user management and security layer within your cloud environment.

- **Step 1.1: Install Supabase Auth Helpers:** In the terminal of your GitHub Codespace, run npm install @supabase/ssr @supabase/supabase-js to install the server-side rendering auth library.[13]
- **Step 1.2: Create Supabase Clients:** In the Codespace editor, create utility files for instantiating the Supabase client for both client-side and server-side contexts, following the official Supabase Next.js documentation.[13]
- **Step 1.3: Implement Middleware:** Create a middleware.ts file in the root of the project within the Codespace. This middleware will run on every request to manage and refresh the user's session cookie, which is essential for server-side authentication.[13]
- **Step 1.4: Build Authentication Pages:**
  1. In the Codespace editor, create app/login/page.tsx and app/signup/page.tsx.
  2. Build the UI forms for login and sign-up using Tailwind CSS.
  3. Implement the sign-up logic. On the sign-up page, include the "Creator's Bill of Rights" modal, which must be accepted before the form can be submitted.
  4. Use Next.js Server Actions to handle form submissions, calling the Supabase signInWithPassword and signUp methods securely on the server.
- **Step 1.5: Create Profile Trigger and Set Roles:**
  1. In your web browser, navigate to your Supabase project dashboard and open the

SQL Editor. Create a database function and a trigger that automatically adds a new row to the profiles table when a user signs up.
2. After signing up with the founder's own account, manually navigate to the profiles table in the Supabase dashboard and change the role value for that user from 'creator' to 'founder'.

Phase 2: The Creator's Project Ingestion Engine

This phase builds the core functionality for the creator user, all from your Codespace.

- **Step 2.1: Build Creator Dashboard:**
  1. Create a route group app/(creator)/dashboard/page.tsx. Protect this route using the middleware to ensure only authenticated users can access it.
  2. On this page, fetch and display a list of projects owned by the currently logged-in user.
  3. Add a "Create New Project" button.
- **Step 2.2: Create New Project Form:**
  1. Create a page app/(creator)/projects/new/page.tsx.
  2. Build a form that allows the creator to input the initial project details (title, logline, etc.). On submission, this form will use a Server Action to insert a new row into the projects table in Supabase.
- **Step 2.3: Implement Secure File Upload:**
  1. On the project page, add a file input component for the script upload.
  2. The file upload process must follow the secure signed URL pattern to handle large files and avoid platform limitations.[15]
  3. **On the client:** When a user selects a file, the "Upload" button's onClick handler will *first* call a Server Action or client-side function that fetches from the /api/projects/request-upload-url endpoint, passing the file's name and type.
  4. **On the backend:** The /api/projects/request-upload-url function (a Python serverless function) will use the Supabase admin client to generate a signed upload URL with a short expiry time (e.g., 5 minutes).
  5. **Back on the client:** The client receives the signed URL. It then uses the standard Supabase client-side library (supabase.storage.from(...).uploadToSignedUrl(...)) to upload the file directly from the user's browser to Supabase Storage.
  6. **Final confirmation:** Upon a successful direct upload, the client makes a final call to the /api/projects/finalize-upload endpoint, passing the project ID and the file path. This endpoint updates the database and triggers the AI agent.

Phase 3: The AI Packaging Agent Backend

This phase builds the core intelligence of the MVP using Python serverless functions, developed in your Codespace and deployed automatically by Vercel.

- **Step 3.1: Create Python API Endpoint:** In the Codespace editor, create a file at api/projects/run-packaging-agent/index.py. This will be the entry point for the Flask application that handles the AI processing.[11] Set up a requirements.txt file in the same directory to include dependencies like flask, supabase, and anthropic.
- **Step 3.2: Implement the Main Function:** The function will be designed to be triggered asynchronously (e.g., by the finalize-upload endpoint). It will receive a projectId as input.
- **Step 3.3: Fetch Data:** The function will use the Supabase Python client to fetch the project details and download the associated script from Supabase Storage.
- **Step 3.4: Execute Prompt Chain:** Implement the prompt chain logic from Section 3.2. This involves making a sequence of calls to the Claude 3 Opus API via the anthropic Python library. Each step's output is used as input for the next. Implement robust error handling and retries for these API calls.
- **Step 3.5: Generate Document:** After receiving the final text content for the pitch deck from Claude, use a library like python-docx to programmatically create a formatted .docx file.
- **Step 3.6: Store and Link Asset:** Upload the newly generated .docx file to a separate folder in Supabase Storage. Finally, insert a new record into the generated_assets table, linking the asset back to the original project.

Phase 4: The Founder's Command Center

This phase builds the private, internal tool that is the operational core of the managed marketplace.

- **Step 4.1: Create Founder-Only Routes:**
  1. Create a new route group in Next.js, app/(founder)/....
  2. Update the middleware.ts file to protect all routes under /founder. The middleware will check the user's session and then query the profiles table to ensure the user has the role of 'founder'. If not, it will redirect them.
- **Step 4.2: Build Main Dashboard UI:**
  1. Create app/(founder)/dashboard/page.tsx.
  2. This page will make a server-side request to the /api/founder/dashboard endpoint to fetch all necessary data.

3. Display the data in a clear, organized manner: a table of all projects, a list of platform mandates, and an overview of the deal pipeline.

- **Step 4.3: Build Project Review Page:**
  1. Create a dynamic route app/(founder)/projects/[id]/page.tsx.
  2. This page will display all information for a single project, including the creator's details, the uploaded script (with a download link), and all AI-generated assets (with download links). This is where the founder performs their curation and review.
- **Step 4.4: Implement Mandates Management:**
  1. Create a page app/(founder)/mandates/page.tsx.
  2. Build a simple CRUD (Create, Read, Update, Delete) interface for the platform_mandates table. A form will allow the founder to add new mandates, and a table will display existing ones with options to edit or delete.
- **Step 4.5: Implement Deal Pipeline Management:**
  1. On the project review page (/projects/[id]), add a section for managing the deal pipeline.
  2. This interface will allow the founder to add new outreach entries to the deal_pipeline table (e.g., "Pitched to Buyer X"), update their status, and log any feedback received.

Section 5: The Claude Co-Pilot: A Library of Production-Ready Prompts

This section contains a library of specific, copy-pasteable prompts designed to instruct an AI coding assistant (Claude) to build the components described in Section 4 within your cloud-based development environment.

5.1: Supabase Schema & RLS Prompts

Prompt 1: Create Core Database Tables
Role: You are an expert PostgreSQL database administrator specializing in Supabase.
Task: Your task is to write the SQL statements to create the core tables for the Manthan application: profiles, projects, script_uploads, and generated_assets.
Context: The profiles table links to Supabase's built-in auth.users table. The projects table is the central entity for a creator's work. script_uploads and generated_assets store file metadata, with the actual files residing in Supabase Storage. I will be running this SQL in the Supabase Dashboard's SQL Editor.
Requirements:

- Use uuid for all primary keys.
- Establish foreign key relationships correctly.
- Use timestamptz for timestamps and set default values to now().
- Use jsonb for flexible data structures like character breakdowns.
- Use text for array fields like genres.
- Provide the complete, runnable SQL script.

SQL

```sql
-- Create the profiles table to store public user data
CREATE TABLE public.profiles (
  id UUID PRIMARY KEY REFERENCES auth.users(id) ON DELETE CASCADE,
  full_name TEXT,
  role TEXT NOT NULL DEFAULT 'creator',
  created_at TIMESTAMPTZ NOT NULL DEFAULT now()
);
-- Set up Row Level Security for profiles
ALTER TABLE public.profiles ENABLE ROW LEVEL SECURITY;
CREATE POLICY "Users can view their own profile" ON public.profiles FOR SELECT USING
(auth.uid() = id);
CREATE POLICY "Users can update their own profile" ON public.profiles FOR UPDATE USING
(auth.uid() = id);

-- Create the projects table
CREATE TABLE public.projects (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  owner_id UUID NOT NULL REFERENCES public.profiles(id) ON DELETE CASCADE,
  title TEXT NOT NULL,
  status TEXT NOT NULL DEFAULT 'draft',
  logline TEXT,
  synopsis TEXT,
  genre TEXT,
  character_breakdowns JSONB,
  budget_range TEXT,
  target_platforms TEXT,
  created_at TIMESTAMPTZ NOT NULL DEFAULT now()
);
-- Set up RLS for projects (to be completed later)
ALTER TABLE public.projects ENABLE ROW LEVEL SECURITY;
```

```sql
-- Create the script_uploads table
CREATE TABLE public.script_uploads (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  project_id UUID NOT NULL REFERENCES public.projects(id) ON DELETE CASCADE,
  file_path TEXT NOT NULL,
  file_name TEXT,
  file_size BIGINT,
  uploaded_at TIMESTAMPTZ NOT NULL DEFAULT now()
);
-- Set up RLS for script_uploads (to be completed later)
ALTER TABLE public.script_uploads ENABLE ROW LEVEL SECURITY;

-- Create the generated_assets table
CREATE TABLE public.generated_assets (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  project_id UUID NOT NULL REFERENCES public.projects(id) ON DELETE CASCADE,
  asset_type TEXT NOT NULL,
  asset_url TEXT,
  version INTEGER NOT NULL DEFAULT 1,
  created_at TIMESTAMPTZ NOT NULL DEFAULT now()
);
-- Set up RLS for generated_assets (to be completed later)
ALTER TABLE public.generated_assets ENABLE ROW LEVEL SECURITY;
```

5.2: Next.js Frontend Component Prompts

Prompt 2: Create the Creator Dashboard Page
Role: You are an expert Next.js developer proficient in using Supabase, TypeScript, and Tailwind CSS.
Task: Your task is to create the main dashboard page for a logged-in creator.
Context: I am working in a GitHub Codespace. My project uses Next.js with the App Router. A server-side Supabase client has already been configured. The page should be a Server Component that fetches data on the server and renders it.
Requirements:

1. Create a new file at app/(creator)/dashboard/page.tsx.
2. The component must be an async function.
3. It should import and use the server-side Supabase client to fetch the user's session. If no user is logged in, it should redirect to /login.
4. It should then fetch all projects from the projects table where the owner_id matches the current user's ID.

5. The UI should display the user's full name (e.g., "Welcome, [User Name]!").
6. It should display a list of the user's projects. If there are no projects, display a message like "You haven't created any projects yet."
7. Include a "Create New Project" button that links to /projects/new.
8. Use basic Tailwind CSS for styling. Provide the full, complete code for the file.

5.3: Vercel Serverless Function Prompts (Python)

Prompt 3: Create the AI Packaging Agent Endpoint
Role: You are an expert Python developer specializing in creating serverless functions on Vercel using the Flask framework.
Task: Create the backend API endpoint that runs the AI Packaging Agent.
Context: The endpoint will be located at /api/projects/run-packaging-agent. It will be triggered by another function and receive a projectId in a JSON payload. It needs to interact with Supabase to get data and the Anthropic API to process it. For this prompt, focus on the overall structure and the first two steps of the AI prompt chain. The environment variables are already set in Vercel.
Requirements:

1. Create a file at api/projects/run-packaging-agent/index.py.
2. Set up a basic Flask application.
3. The main route should accept POST requests.
4. It should extract the projectId from the request body.
5. Initialize the Supabase client using environment variables (SUPABASE_URL, SUPABASE_SERVICE_KEY).
6. Initialize the Anthropic client using the ANTHROPIC_API_KEY environment variable.
7. Write the logic to:
   a. Fetch the project's script_upload path from the script_uploads table.
   b. Download the script file from Supabase Storage.
   c. Read the script content into a variable.
   d. Construct and execute the "Core Elements Extraction" prompt (Step 2 from the prompt chain table) by sending the script content to the Claude 3 Opus model.
8. For now, simply print the JSON response from Claude to the console.
9. Return a 200 OK response.
10. Create a requirements.txt file in the same directory with Flask, supabase, and anthropic.

5.4: Authentication and Middleware Prompts

Prompt 4: Create the Security Middleware
Role: You are an expert Next.js developer specializing in authentication and middleware using the @supabase/ssr library.
Task: Create the middleware.ts file to manage user sessions and protect routes.
Context: The application has public routes (e.g., /login), creator-only routes (e.g., /dashboard), and founder-only routes (e.g., /founder). The middleware must handle session refreshing and enforce access control. I am developing in a GitHub Codespace.
Requirements:

1. Create a new file at middleware.ts in the project root.
2. Import createMiddlewareClient from @supabase/ssr.
3. The middleware function should refresh the user's session on each request.
4. Implement logic to protect routes:
   a. If a user is not authenticated and tries to access a protected route (anything not /login or /signup), redirect them to /login.
   b. (Advanced) Add a placeholder comment for the logic that will check the user's role from the profiles table to protect the /founder routes. Explain that this will require an additional database call.
5. Configure the matcher in the config object to exclude static assets and _next files from the middleware's execution.
6. Provide the full, complete code for the file.

Conclusion and Actionable Recommendations

This report provides an exhaustive, microscopic blueprint for the development and deployment of the Project Manthan MVP's first strategic wedge. By adhering to the "Managed Accelerated Path" strategy, the focus remains squarely on the most critical, revenue-generating activities: packaging high-quality creative projects and facilitating deals.[1] The proposed architecture, leveraging Vercel and Supabase, is optimized for a non-technical founder, ensuring rapid development, minimal operational overhead, and a lean financial burn rate.[1, 1]

The key actionable recommendations are as follows:

1. **Prioritize the Founder's Command Center:** The immediate development priority should be the private administrative dashboard. This tool is the engine of the "human-in-the-loop" model and the primary mechanism for capturing the proprietary data that will form Manthan's long-term competitive moat.
2. **Execute the Build Sequentially:** Follow the phased, cloud-only implementation guide in Section 4 precisely. Each phase builds upon the last, from foundational authentication to the creator-facing tools and finally the founder's command center. This logical

progression de-risks the development process.

3. **Leverage the AI Co-Pilot Prompts:** Use the detailed prompts in Section 5 as direct instructions for the AI coding assistant. This will ensure the implementation aligns perfectly with the architectural design and strategic requirements, minimizing ambiguity and rework.
4. **Embrace Data Collection from Day One:** Meticulously use the founder's tools to log all project data, market intelligence (platform_mandates), and deal outcomes (deal_pipeline). This disciplined data entry is not administrative overhead; it is the active process of building the company's most valuable asset—the training data for future AI-driven matchmaking and success prediction.

By executing this plan, Project Manthan can launch a highly defensible, monetizable MVP that solves a critical pain point for Indian creators, builds a foundation of trust, and strategically positions itself to become the indispensable operating system for the Indian M&E industry.

Works cited

1. Manthan Pivot - Overview.docx
2. India | Jurisdictions - DataGuidance, accessed September 1, 2025, https://www.dataguidance.com/jurisdictions/india
3. THE DIGITAL PERSONAL DATA PROTECTION ACT, 2023 (NO. 22 OF 2023) An Act to provide for the processing of digital personal data in, accessed September 1, 2025, https://www.meity.gov.in/static/uploads/2024/06/2bf1f0e9f04e6fb4f8fef35e82c42aa5.pdf
4. Asking for phone number of shoppers against data law, accessed September 1, 2025, https://timesofindia.indiatimes.com/business/india-business/asking-for-phone-no-of-shoppers-against-data-law/articleshow/123534278.cms
5. India Over The Top Content Market Size, Share Report Forecast 2035, accessed September 1, 2025, https://www.marketresearchfuture.com/reports/india-over-the-top-content-market-61504
6. Journal of Civil & Legal Sciences - Diversity of Indian Regional Content on OTT Platforms: A Critical Review, accessed September 1, 2025, https://www.omicsonline.org/open-access/diversity-of-indian-regional-content-on-ott-platforms-a-critical-review-124924.html
7. From Margins to Mainstream: India's OTT boom turns regional to capture Bharat's diverse viewers - Storyboard18, accessed September 1, 2025, https://www.storyboard18.com/how-it-works/from-margins-to-mainstream-indias-ott-boom-turns-regional-to-capture-bharats-diverse-viewers-65800.htm
8. "In 2025, we're redefining storytelling," says Monika Shergill as Netflix India unveils its content slate for 2025 - MediaBrief, accessed September 1, 2025, https://mediabrief.com/netflix-india-unveils-its-content-slate-for-2025/
9. Netflix India's 2025 lineup bets big on star kids, South hits, and sports

entertainment, accessed September 1, 2025, https://www.fortuneindia.com/enterprise/netflix-indias-2025-lineup-bets-big-on-star-kids-south-hits-and-sports-entertainment/120389
10. Prime Video's new brand campaign puts feelings at the heart of streaming - BrandEquity, accessed September 1, 2025, https://brandequity.economictimes.indiatimes.com/news/advertising/prime-videos-new-brand-campaign-puts-feelings-at-the-heart-of-streaming/122566862
11. How can I use Python and JavaScript in the same application? - Vercel, accessed September 1, 2025, https://vercel.com/guides/how-to-use-python-and-javascript-in-the-same-application
12. Using the Python Runtime with Vercel Functions, accessed September 1, 2025, https://vercel.com/docs/functions/runtimes/python
13. Build a User Management App with Next.js | Supabase Docs, accessed September 1, 2025, https://supabase.com/docs/guides/getting-started/tutorials/with-nextjs
14. Use Supabase Auth with Next.js, accessed September 1, 2025, https://supabase.com/docs/guides/auth/quickstarts/nextjs
15. Signed URL file uploads with NextJs and Supabase | by Ollie - Medium, accessed September 1, 2025, https://medium.com/@olliedoesdev/signed-url-file-uploads-with-nextjs-and-supabase-74ba91b65fe0
16. Supabase storage guide for Next.JS - SupaLaunch, accessed September 1, 2025, https://supalaunch.com/blog/supabase-storage-guide-for-nextjs
17. Best LLMs for Creative Writing - AI Writing Tools Guide - Aloa, accessed September 1, 2025, https://aloa.co/ai/comparisons/llm-comparison/best-llms-for-creative-writing/
18. What's the Best LLM for Creative Writing? - Generative, accessed September 1, 2025, https://www.genagency.ca/generative-blog/whats-the-best-llm-for-creative-writing
19. Movie Success Prediction using Historical and Current Data Mining - International Journal of Computer Applications, accessed September 1, 2025, https://www.ijcaonline.org/archives/volume178/number47/chakraborty-2019-ijca-919415.pdf
20. Predicting Movie Success..! - Kaggle, accessed September 1, 2025, https://www.kaggle.com/code/harshadeepvattikunta/predicting-movie-success
21. Finding Nemo: Predicting Movie Performances by Machine Learning Methods - MDPI, accessed September 1, 2025, https://www.mdpi.com/1911-8074/13/5/93
22. How to Write a Movie Script: Screenplay Format and Examples - StudioBinder, accessed September 1, 2025, https://www.studiobinder.com/blog/how-to-write-a-screenplay/
23. Screenplay - Wikipedia, accessed September 1, 2025, https://en.wikipedia.org/wiki/Screenplay
24. Heuristics & approximate solutions | AP CSP (article) - Khan Academy, accessed

September 1, 2025, https://www.khanacademy.org/computing/ap-computer-science-principles/algorithms-101/solving-hard-problems/a/using-heuristics
25. A Predictor for Movie Success - CS229: Machine Learning, accessed September 1, 2025, https://cs229.stanford.edu/proj2013/EricsonGrodman-APredictorForMovieSuccess.pdf
26. Predicting movie success with machine learning techniques: ways to improve accuracy, accessed September 1, 2025, https://pure.coventry.ac.uk/ws/files/11062379/author_revision_copy_Predicting_Movie_Success_with_Machine_Learning_Te....pdf
27. Movie Success Prediction System using Python with the help of data science - Medium, accessed September 1, 2025, https://medium.com/@kattilaxman4/movie-success-prediction-system-using-python-with-the-help-of-data-science-49d3a1593254
28. Early Prediction of Movie Success Using Machine Learning and Evolutionary Computation | Request PDF - ResearchGate, accessed September 1, 2025, https://www.researchgate.net/publication/365255175_Early_Prediction_of_Movie_Success_Using_Machine_Learning_and_Evolutionary_Computation
29. Supabase for Vercel, accessed September 1, 2025, https://vercel.com/marketplace/supabase
30. vercel/nextjs-subscription-payments: Clone, deploy, and fully customize a SaaS subscription application with Next.js. - GitHub, accessed September 1, 2025, https://github.com/vercel/nextjs-subscription-payments
31. How To Deploy a Next.js App To Vercel With GitHub Actions - freeCodeCamp, accessed September 1, 2025, https://www.freecodecamp.org/news/deploy-to-vercel-with-github-actions/
32. Deploying GitHub Projects with Vercel, accessed September 1, 2025, https://vercel.com/docs/git/vercel-for-github