That's an excellent idea. A detailed, consolidated overview of all three modules provides the entire team with a complete roadmap, clarifies dependencies, and ensures everyone understands the project's phased delivery structure.

Here is the detailed overview of the entire project pipeline, broken down into the three modules.

🗺️ Project Roadmap: The Procrastination Analyst (3-Module Overview)

The project is structured into three sequential phases to ensure a stable foundation before integrating complex logic and the frontend user experience.

🧱 Module 1: Foundation & Security (Focus: Week 1)

This module establishes the secure entry point and the structural backbone of the entire application.

Area	Focus	Key Deliverables & Responsibilities
Lead Teammate	**Teammate 2 (Security)**	Owns server instantiation, DB connection, and user protection.
Scope	**Server Setup & Authentication Gateway.** The application must be connectable and all user access must be verified.	
Backend Components	1. **Server Initialization:** Express setup, MongoDB connection (mongoose). 2. **Schema Definition:** Define all three Mongoose Schemas (User, IntendedTask, LogEntry). 3. **JWT Authentication:** Implement /register and	

	/login routes with password hashing (bcryptjs) and token generation (jsonwebtoken). 4. **JWT Middleware:** The reusable function to protect all future API routes.	
Module Completion Check	Can a user successfully register, log in, receive a token, and use that token to access a simple protected route?	
Handoff (Module 2)	Teammate 1 receives the **finalized Schemas** and access to the **protected environment**.	

🧠 Module 2: Core Data & Logic (Focus: Weeks 2 & 3)

This module implements the core business logic and analysis that makes this project unique. It is a strictly backend-focused phase.

Area	**Focus**	**Key Deliverables & Responsibilities**
Lead Teammate	**Teammate 1 (Architect)**	Owns all data logic, aggregation, and the asynchronous pipeline.
Scope	**Data Capture, PVI Calculation, and Root Cause ID.** This phase turns raw data into diagnostic information.	
Backend Components	1. **Secure CRUD Endpoints:** Full Create,	

	Read, Update, Delete routes for **Tasks** and **Logs**, all protected by the JWT Middleware (Teammate 2 assists here). 2. **Async PVI Algorithm:** Mongoose **post-save hook** on LogEntry to asynchronously calculate and update the PVI score on the User profile. 3. **Root Cause Pattern ID:** Logic to aggregate log history and assign a behavioral label (e.g., *Overwhelm Avoidance*). 4. **Analysis Endpoints:** Create dedicated APIs for fetching chart data: **Time-of-Day Aggregation** and **Benchmarking Comparison** data. 5. **Benchmarking File Handling:** Implement the secure endpoint to serve the simple **Benchmarking Report** file.	
Module Completion Check	Using Postman, can we create a Log, and does the User profile show an updated PVI score and a correct Root Cause ID? Are the analysis APIs returning structured JSON data?	
Handoff (Module 3)	Teammate 3 (Frontend) receives the complete **API Contract** (all endpoints and the exact JSON data structures).	

🖥️ Module 3: Frontend & Polish (Focus: Weeks 4, 5, & 6)

This module is responsible for transforming the raw data into a dynamic, intuitive user experience, integrating all the features required for a high-quality presentation.

Area	Focus	Key Deliverables & Responsibilities
Lead Teammate	**Teammate 3 (Frontend)**	Owns all UI/UX, state management, and final data visualization.
Scope	**Full User Experience, Conditional Logic, and Visualization.** The front-end proves the efficacy of the backend analysis.	
Frontend Components	1. **Structural Setup:** React initialization, **React Router** for navigation. 2. **Forms & Auth:** Build all input forms (Login, Register, Tasks, Logs) and securely consume the JWT token. 3. **Dashboard Core:** Build the PVI Velocity Gauge with **Custom CSS**. 4. **Conditional Logic:** Implement logic to change the gauge color and conditionally suggest the **Task Deconstructor Mode** based on the Root Cause Pattern ID. 5. **Visualization:** Integrate components for the **Time-of-Day Chart** and **In-Dashboard Comparison Chart**. 6.	

	Intervention Mode: Implement the **Task Deconstructor Mode** using **React Hooks** for interactive sub-task management. 7. **Polish:** Final styling, responsiveness, and file download link for the Benchmarking Report.	
Module Completion Check	Can a new user register, log a distraction, see their PVI updated on the dashboard, and be automatically suggested the correct Countermeasure Mode?	
Final Delivery	Completed, tested, and deployable MERN stack application.	

This detailed overview confirms the project's entire scope and ensures your team has the roadmap needed for a successful 6-week academic delivery.

We are now ready to begin Module 1. Shall we start by writing the Mongoose code for the User Model and its security components?