**Vortexify Development Roadmap**

**🧩 Phase 1: Heart Module – Automation Engine & Deployment Logic**

🧠 *"The heart pumps the blood (code) into the system, initiates life (automation), and builds the body (VM images)."*

**📅 Estimated Time: 4-5 weeks**

**✅ Core Responsibilities:**

* Git repo cloning
* Docker image creation
* VMware virtual deployment
* Shell script automation
* Python orchestration engine

**📌 Key Milestones:**

| **Week** | **Tasks** |
| --- | --- |
| **Week 1** | 🔹 Setup folder structure 🔹 Develop repo cloning logic in Python 🔹 Test Cloning Logic work on only main Branch. |
| **Week 2** | 🔹 Integrate Dockerfile creation + image build 🔹 Tag images per user ID |
| **Week 3** | 🔹 Automate image deployment on VMware using vmrun or govc 🔹 Add error logs and rollback |
| **Week 4** | 🔹 Final testing & dry-run automation 🔹 Document automation logic & create flowcharts |

**💻 Phase 2: Skin Module – UI/UX & User Interaction Layer**

🎨 *"The skin is what the user touches. It reflects personality and responds to interaction."*

**📅 Estimated Time: 3–4 weeks**

**✅ Core Responsibilities:**

* Laravel + Blade UI for user interaction
* Registration, login, dashboard, deployment form
* Display deployment history
* Optional: Integrate React-based Forgot Password module

**📌 Key Milestones:**

| **Week** | **Tasks** |
| --- | --- |
| **Week 1** | 🔹 Setup Laravel project structure 🔹 Design UI mockups (Login, Register, Dashboard) |
| **Week 2** | 🔹 Implement Auth system (JWT/Sanctum) 🔹 Create form to collect repo URL, branch, config |
| **Week 3** | 🔹 Fetch and display deployment status using backend APIs 🔹 Connect with Heart APIs |
| **Week 4** | 🔹 Add logs/history viewer 🔹 (Optional) Add React-based password recovery page |
| **Week 5** | 🔹 Conduct UX testing 🔹 Finalize all front-end logic and forms |

**🧠 Phase 3: Brain Module – Orchestration & Backend Services**

🧩 *"The brain remembers, connects, and controls execution."*

**📅 Estimated Time: 4–5 weeks**

**✅ Core Responsibilities:**

* Spring Boot service for job management
* User-to-deployment mapping
* Handle queueing, job execution status
* Docker APIs, log endpoints
* Connect to MySQL DB
* Optional: C++-based performance logging (future phase)

**📌 Key Milestones:**

| **Week** | **Tasks** |
| --- | --- |
| **Week 1** | 🔹 Setup Spring Boot base 🔹 Create DTOs for user, deployment jobs |
| **Week 2** | 🔹 Connect to MySQL 🔹 REST API: Job trigger, status, logs |
| **Week 3** | 🔹 Create job manager service 🔹 Queue/worker-based design |
| **Week 4** | 🔹 Integrate with Laravel frontend 🔹 Return results + logs |
| **Week 5** | 🔹 (Optional) Design C++ log module stub 🔹 Final test of end-to-end communication |

**📦 Final Integration Phase**

| **Milestone** | **Description** |
| --- | --- |
| 🔗 **System Integration** | Make sure Laravel triggers backend, backend triggers Python scripts |
| 🔁 **Error Handling** | Add fault-tolerant code and rollback support |
| 🧪 **System Testing** | Perform E2E testing with real-world sample repos |
| 📘 **Documentation** | Update system doc, tech stack doc, and GitHub README |
| 🚀 **Deployment Setup** | Prepare final local/VM setup & share instructions for run |

**🔧 Optional Enhancements:**

* Logging using C++ (Heart or Brain module)

✅ Summary View

PHASE 1 (HEART) => Automation & Docker Engine

PHASE 2 (SKIN ) => Laravel UI + React Forgot Password

PHASE 3 (BRAIN) => Spring Boot + MySQL Service Layer

FINAL INTEGRATION => Connect all modules, test end-to-end