Prompt Machine - Claude Code Setup Guide

Overview

This guide provides the complete file structure and setup instructions for implementing the Prompt Machine project. The project has been broken down into manageable components to avoid crashes.

Project Structure

```
prompt-machine/
├── api/
│   ├── src/
│   │   ├── index.js              ✓ Created
│   │   ├── database/
│   │   │   ├── connection.js      ✓ Created
│   │   │   ├── migrate.js         → To create
│   │   │   └── seed.js            → To create
│   │   ├── middleware/
│   │   │   ├── auth.js            ✓ Created
│   │   │   ├── errorHandler.js    → To create
│   │   │   └── validation.js      → To create
│   │   ├── routes/
│   │   │   ├── auth.js            ✓ Created
│   │   │   ├── projects.js        → To create
│   │   │   ├── prompts.js         → To create
│   │   │   ├── deployments.js     → To create
│   │   │   ├── analytics.js       → To create
│   │   │   └── tools.js           → To create
│   │   ├── services/
│   │   │   ├── claude.js          → To create
│   │   │   ├── deployment.js      → To create
│   │   │   └── subdomain.js       → To create
│   │   └── utils/
│   │       ├── logger.js          → To create
│   │       └── validators.js      → To create
│   ├── tests/                     → To create
│   └── package.json               ✓ Created
├── app/
│   ├── src/                       → React app to create
│   ├── public/
│   └── package.json               → To create
├── tools/
```

```
│   ├── server.js              → To create
│   └── generator.js            → To create
├── config/
│   ├── .env.template           → Created by setup script
│   └── .env                  → To be created from template
├── scripts/
│   ├── setup-ssl.sh           → Created by setup script
│   └── deploy.sh              → Created by setup script
└── ecosystem.config.js         → Created by setup script
```

Setup Steps for Claude Code

1. Server Preparation

```bash
# Run the server setup script first
bash server-setup-script.sh
```

2. Database Setup

```bash
# Connect to PostgreSQL and create database/user
psql -h sql.prompt-machine.com -U postgres -W
# Password: Uhr4ryPWey94oE1q7K

# In psql:
CREATE USER promptmachine_userbeta WITH PASSWORD '94oE1q7K';
CREATE DATABASE promptmachine_dbbeta OWNER promptmachine_userbeta;
\q

# Run the schema
psql -h sql.prompt-machine.com -U promptmachine_userbeta -d promptmachine_dbbeta < database-schema.sql
```

3. Environment Configuration

```bash
```

```
cd ~/prompt-machine/config
cp .env.template .env
# Edit .env and add:
# - JWT_SECRET (generate random string)
# - SESSION_SECRET (generate random string)
# - CLAUDE_API_KEY (your API key)
```

4. API Implementation

The following files need to be created:

Core Utilities

- `api/src/utils/logger.js` - Winston logger configuration
- `api/src/middleware/errorHandler.js` - Global error handling
- `api/src/utils/validators.js` - Common validation functions

Route Implementations

- `api/src/routes/projects.js` - Project CRUD operations
- `api/src/routes/prompts.js` - Prompt versioning and management
- `api/src/routes/deployments.js` - Tool deployment endpoints
- `api/src/routes/analytics.js` - Analytics data endpoints
- `api/src/routes/tools.js` - Tool-specific endpoints

Service Layer

- `api/src/services/claude.js` - Claude API integration
- `api/src/services/deployment.js` - Deployment automation
- `api/src/services/subdomain.js` - Subdomain management

Database Scripts

- `api/src/database/migrate.js` - Database migration tool
- `api/src/database/seed.js` - Initial data seeding

5. Admin Dashboard (React App)

Create a React application in the `app/` directory with:

- Project management interface

- Interactive prompt builder with Claude
- Deployment controls
- Analytics dashboard

6. Tool Server

Implement the tool server that:

- Serves deployed tools on subdomains
- Handles user interactions
- Integrates with Claude API
- Tracks analytics

7. Key Features to Implement

Interactive Prompt Builder

1. Admin initiates conversation with Claude
2. Claude asks about expert role and purpose
3. Interactive refinement of prompt
4. Field configuration for user inputs
5. Testing interface

Deployment System

1. Generate tool frontend based on prompt
2. Create subdomain entry
3. Configure Nginx
4. Generate SSL certificate
5. Deploy to PM2

Analytics & Monetization

1. Track tool usage
2. Google AdWords integration
3. Revenue tracking
4. Usage reports

Implementation Priority

1. **Phase 1**: Complete API backend
    - Finish all routes
    - Implement Claude service
    - Basic authentication working
2. **Phase 2**: Admin Dashboard
    - Project management
    - Basic prompt builder
    - Manual deployment
3. **Phase 3**: Tool Generation
    - Dynamic frontend generation
    - Subdomain automation
    - Basic tool serving
4. **Phase 4**: Advanced Features
    - Analytics
    - Monetization
    - Mobile optimization

Next Immediate Steps

1. Create the missing utility files (logger, error handler)
2. Implement the project routes
3. Set up Claude API integration
4. Create basic admin dashboard
5. Test end-to-end flow

Important Notes

- The database credentials are included in the original request
- Default admin credentials need to be properly hashed
- SSL setup should be run after domains are properly configured
- PM2 ecosystem file is already created by setup script
- Remember to handle the subdomain wildcard certificate properly

Security Reminders

- Change all default passwords immediately
- Generate strong secrets for JWT and sessions
- Implement proper rate limiting
- Sanitize all user inputs
- Use HTTPS for all communications

This modular approach should prevent the crashes you've been experiencing. Each component can be implemented and tested independently.