**Megaminds IT Services - Assignment Task**

**Source code and Video report –** <https://drive.google.com/drive/folders/1EvvhEp-GEvT_U3S6ZJXFAeqphKQQFdsb?usp=sharing>

**1. Introduction**

This report details the secure design and development of **PixelForge Nexus**, a secure, role-based project and resource management system created for Creative SkillZ LLC. Built using the **MERN** stack (MongoDB, Express, React, Node.js), the system emphasizes secure software practices and was developed with assistance from a Large Language Model (LLM)—OpenAI’s ChatGPT. This report covers the design approach, development lifecycle, security testing, applied formal methods, and ethical considerations associated with building this secure web platform.

**2. System Design (35%)**

**2.1 Architecture Overview**

The system uses a **client-server architecture**:

* **Frontend:** React.js + TailwindCSS
* **Backend:** Node.js + Express.js REST API
* **Database:** MongoDB
* **Authentication:** JSON Web Tokens (JWT) for session management, bcrypt for password hashing

**2.2 Role-Based Access Control (RBAC)**

Implemented through middleware (authorizeRoles), the system supports three roles:

|  |  |
| --- | --- |
| **Role** | **Privileges** |
| **Admin** | Add/edit projects, manage users, upload documents |
| **Project Lead** | Assign developers to projects, upload documents for their own projects |
| **Developer** | View assigned projects and their associated documents |

Role-checking middleware verifies the token and associated role before granting access.

**2.3 Secure Design Principles**

|  |  |
| --- | --- |
| **Principle** | **Implementation** |
| **Least Privilege** | Users can only access features permitted by their roles |
| **Fail-Safe Defaults** | All sensitive endpoints are protected with authentication and role checks |
| **Secure by Design** | Security was a foundational goal, not an afterthought |
| **Separation of Concerns** | Frontend, backend, DB, and auth are separated for modularity |
| **Defense in Depth** | Input validation, JWT, sanitization, rate limiting, and secure file uploads layered |

**2.4 Threat Modelling (STRIDE)**

|  |  |  |
| --- | --- | --- |
| **Threat** | **Analysis** | **Mitigation** |
| **Spoofing** | Unauthorized login attempts | Password hashing with bcrypt, JWT-based sessions |
| **Tampering** | JWT manipulation | JWTs signed and verified; payloads validated |
| **Information Disclosure** | Unauthorized file access | Only project-assigned users can access documents |
| **Denial of Service (DoS)** | Brute-force login attacks | Rate limiting, OTP cooldown, and IP ban logic |
| **Elevation of Privilege** | Exploiting privilege flaws | RBAC middleware and role verification from token payload |

**3. System Development (20%)**

**3.1 Development Tools & Stack**

|  |  |
| --- | --- |
| Component | Technology Used |
| Frontend | React.js, Tailwind CSS, Axios, React Router |
| Backend | Node.js, Express.js, MongoDB, JWT (jsonwebtoken), Bcrypt.js |
| Testing | ThunderClient for API testing |
| Security Libraries | Helmet, Express Rate Limit, Express Mongo Sanitize, Crypto, Dotenv, CORS |
| Email/OTP | Nodemailer, Crypto |
| Sanitization | DOMPurify, jsdom (server-side XSS protection for rich inputs) |
| File Uploads | Multer (configured with file type and size validations) |

**3.2 Core Functionalities**

* **Authentication**
  + Admin login/register
  + JWT-based authentication
  + Passwords securely hashed using bcrypt
* **Project Management**
  + Admins can create, edit, and mark projects completed
  + Developers and leads can view assigned projects
* **Team Assignment**
  + Leads assign developers to projects they manage
* **Document Uploads**
  + Admins and leads upload project docs (PDF, WORD, TXT, XLSV)
  + Developers access only documents tied to their assigned projects
* **User Dashboard**
  + Dynamically updates based on role
* **Account Settings**
  + Password reset functionality using email

**4. Security Testing and Analysis (35%)**

**4.1 Static and Dynamic Analysis**

* **Tool Used:** ESLint for static code analysis
* **Results:**
  + All critical linting and security flags resolved
  + No hardcoded secrets
  + All .env data isolated from version control

**4.2 Key Security Enhancements**

* **Environment Secrets Management**
  + .env used for sensitive keys
  + Example .env.example shared; real keys excluded from repo
* **Rate Limiting on Login**
  + Max 5 attempts per 10 minutes using express-rate-limit
* **OTP Protection**
  + OTP expires in 5 minutes
  + Resend OTP restricted to 60-second intervals
  + Abuse detection logic recommended via DB-stored attempts
* **Input Sanitization**
  + express-mongo-sanitize strips malicious MongoDB operators
  + helmet adds common security headers
* **Secure File Uploads with Multer**
  + Allowed types: PDF, WORD, TXT, XLSV
  + File size validation
  + Files stored **outside public directory** to prevent direct access
* **Role-Based Middleware**
  + Tokens verified
  + Roles decoded from JWT payload before endpoint access

**4.3 Security Testing Summary**

|  |  |  |
| --- | --- | --- |
| **Test** | **Description** | **Result** |
| **Brute Force Attack** | Tried rapid login | Blocked after 5 attempts |
| **XSS Attempt** | Injected <script> in inputs | Sanitized |
| **File Upload Bypass** | Uploaded .exe | Blocked due to MIME check |
| **JWT Forgery** | Tampered token | Rejected due to signature mismatch |
| **Privilege Escalation** | Developer accessed admin route | Denied by middleware |

**5. Formal Methods (10%)**

**5.1 Behavioral Modeling (FSM)**

**User Authentication FSM:**

* States:
  + LoggedOut → LoggingIn → Authenticated
* Transitions:
  + Invalid credentials → stay in LoggedOut
  + Token expired → return to LoggedOut

**Project Lifecycle FSM:**

* States:
  + New → In Progress → Completed
* Transitions:
  + Admin creates → New
  + Work begins → In Progress
  + Mark complete → Completed

**5.2 Verification Techniques**

* Preconditions/postconditions used in routes
* Manual validation of FSM against system logic
* Formal reasoning ensured consistency between states and roles

**6. Development Life Cycle**

|  |  |
| --- | --- |
| **Phase** | **Description** |
| **Requirement Analysis** | Studied brief to identify secure project/document/user handling needs |
| **Design** | Drafted RBAC model, UI wireframes, data flow diagrams |
| **Development** | Used modular routes, LLM support for secure patterns |
| **Testing** | Conducted security testing using ThunderClient and manual verification |
| **Deployment (Local)** | .env used for secure variables; folder structure maintained |
| **Documentation** | Code comments, API usage notes, .env.example for setup |

**7. Ethical & Legal Considerations**

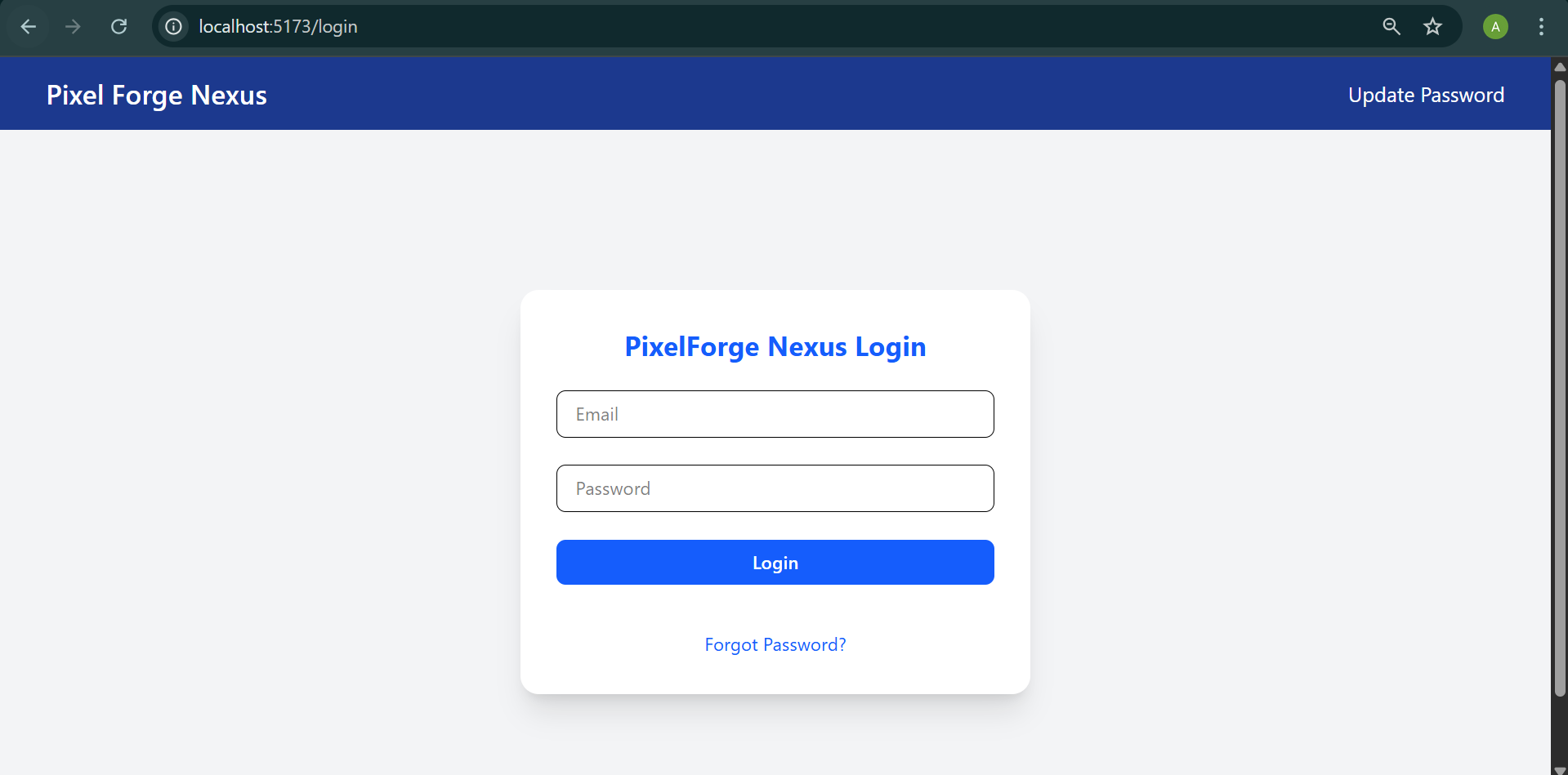
* **Data Privacy**: All sensitive data encrypted (passwords), no PII exposed
* **Role Separation**: Prevented data leaks and unauthorized access
* **Open Source Compliance**: All NPM packages checked for licensing and vulnerabilities
* **No Hardcoded Credentials**: Keys and secrets stored securely
* **GDPR Consideration**: Design aligns with privacy principles such as data minimization

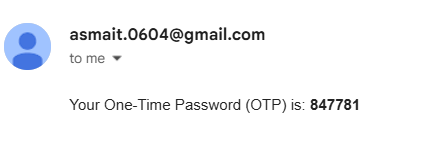
**8. Conclusion**

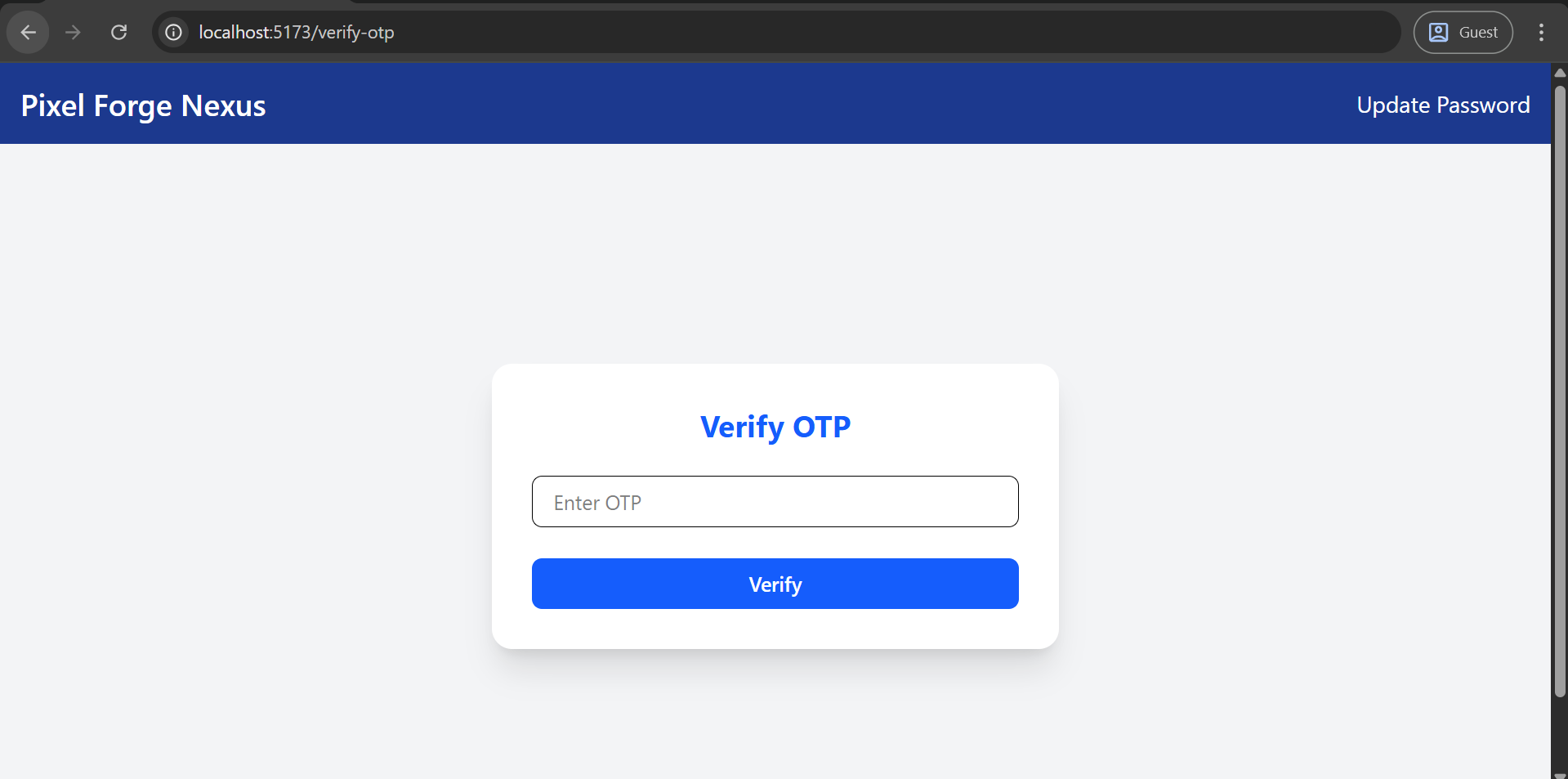
PixelForge Nexus is a fully functional, secure project and asset management platform developed with a strong emphasis on secure design principles. Built with MERN stack and enhanced by best practices recommended through LLM-assisted development, it incorporates robust role-based access, encrypted authentication, secure file handling, and comprehensive threat mitigation strategies.

Through proactive threat modeling, strict access control, formal method applications, and security-focused testing, the platform demonstrates both functional and security excellence. This solution is extendable and ready for real-world production use with minimal enhancements.

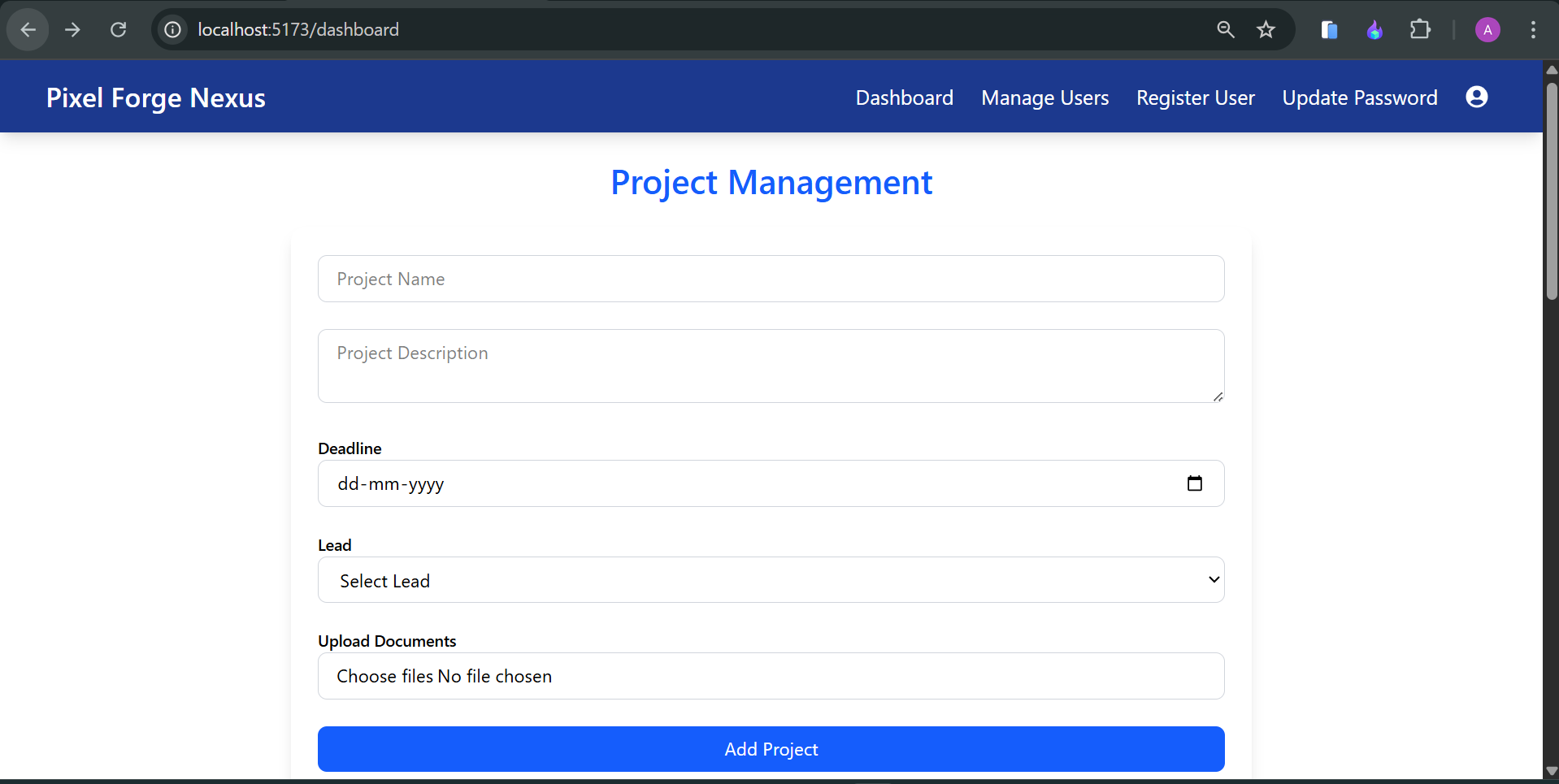
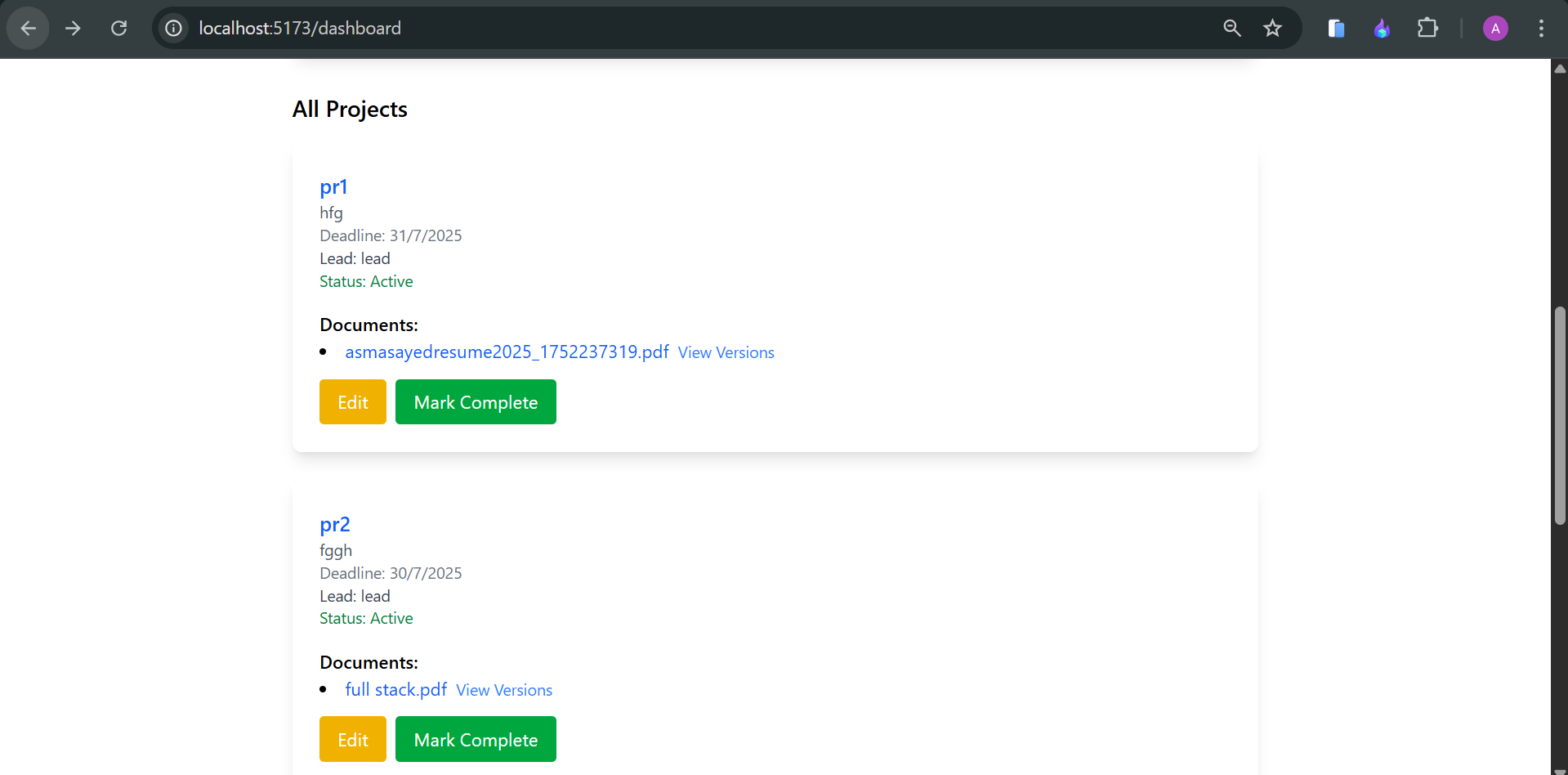
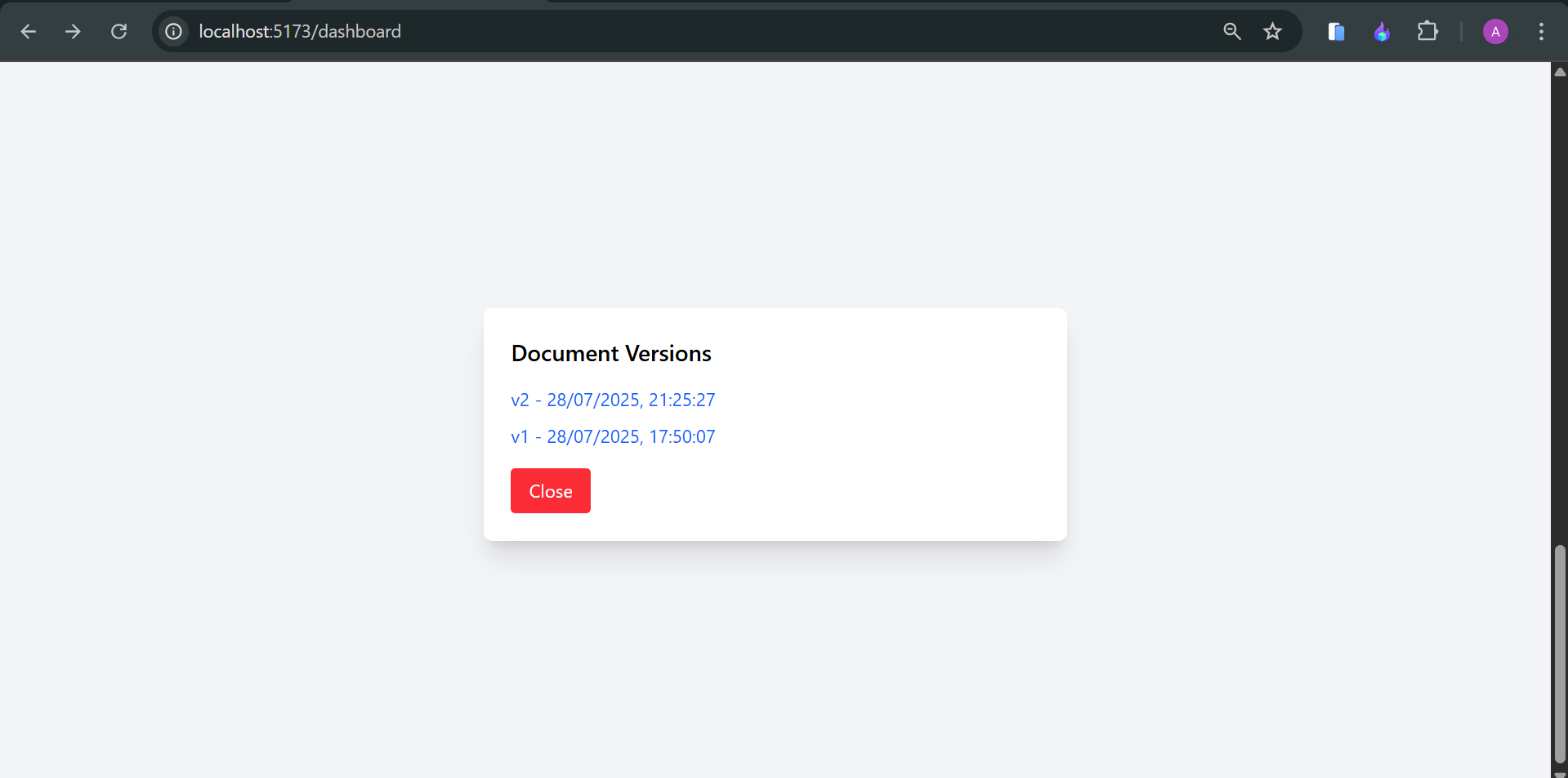
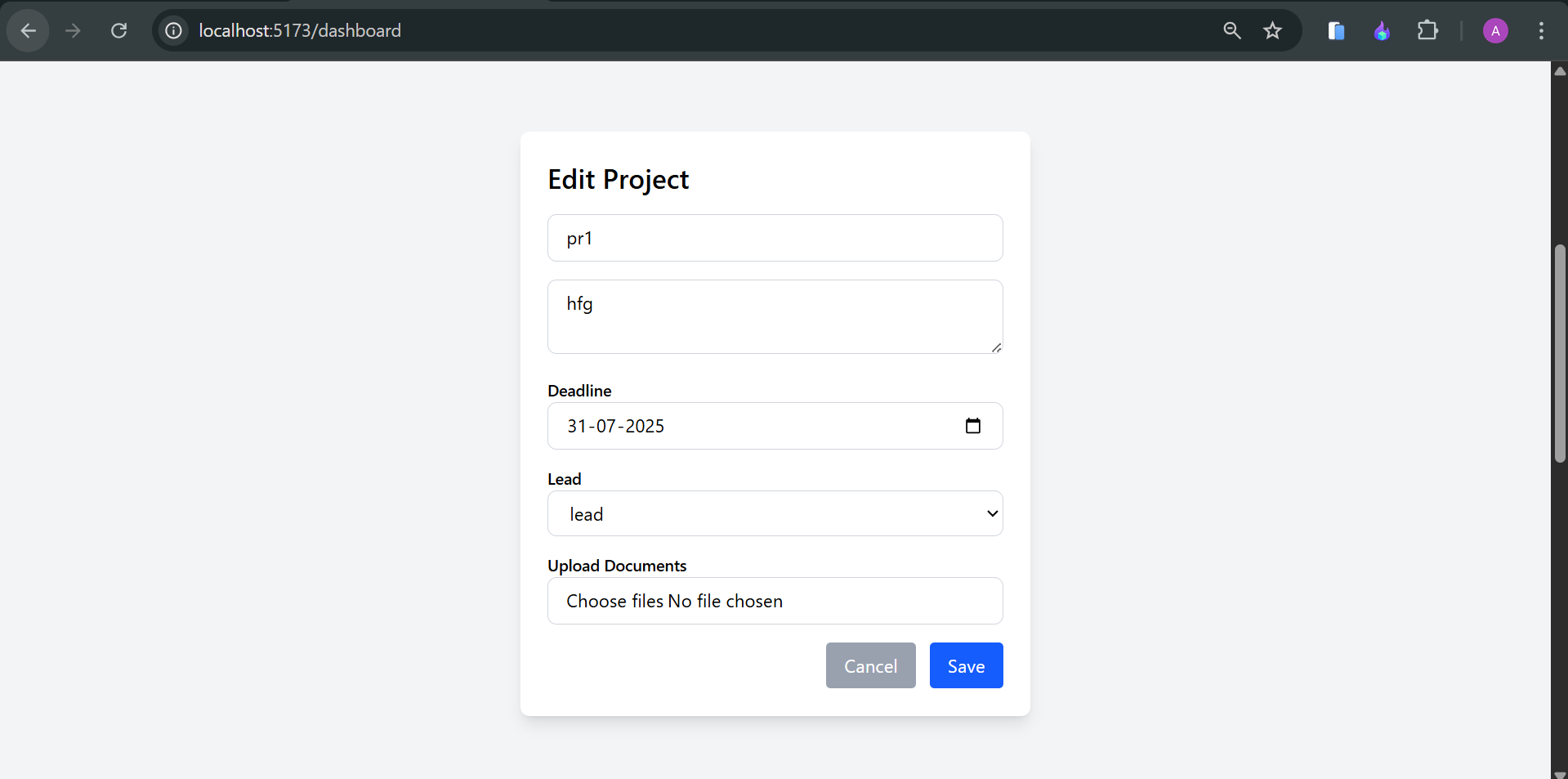
**Project Screenshots:**

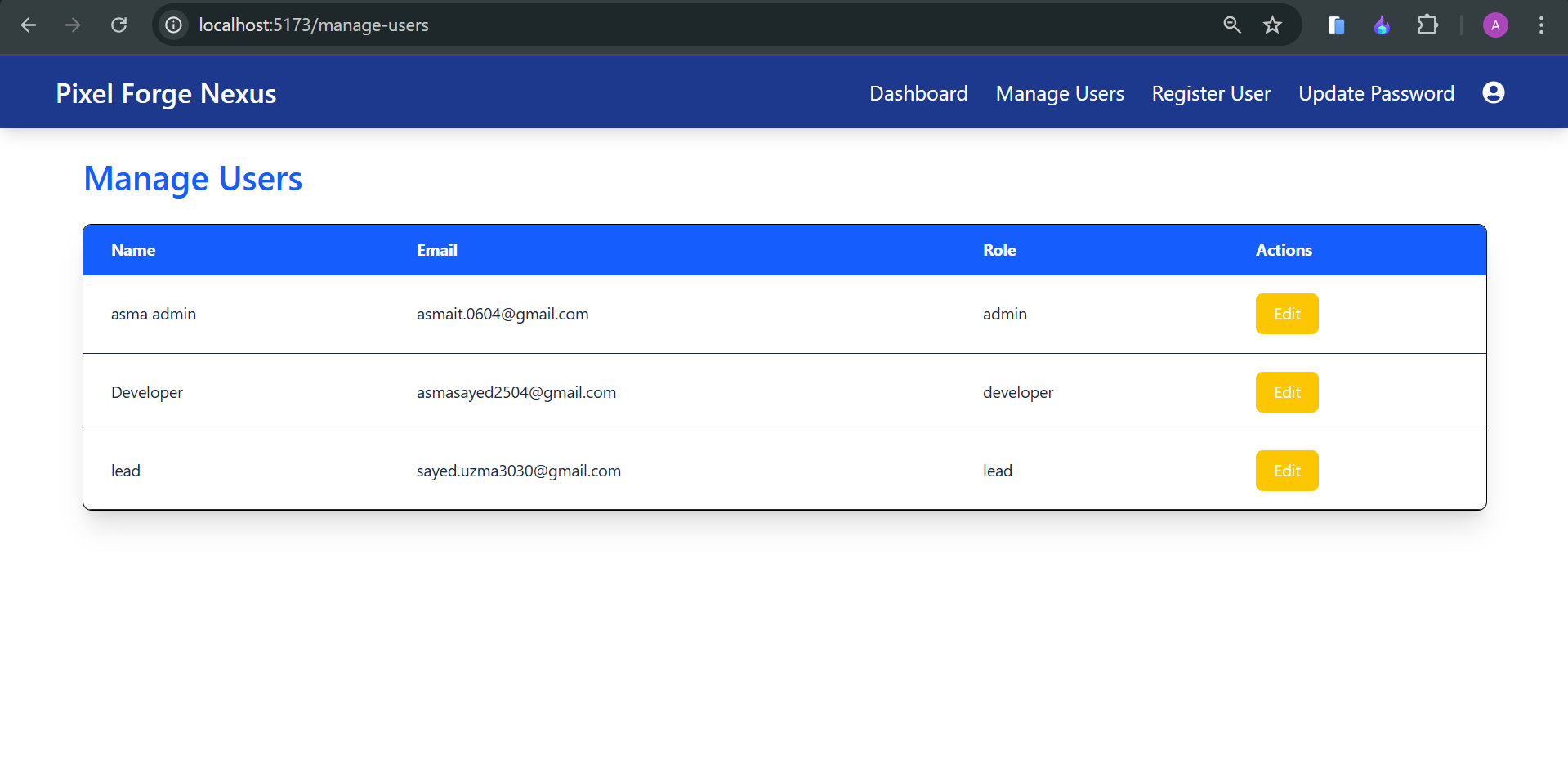
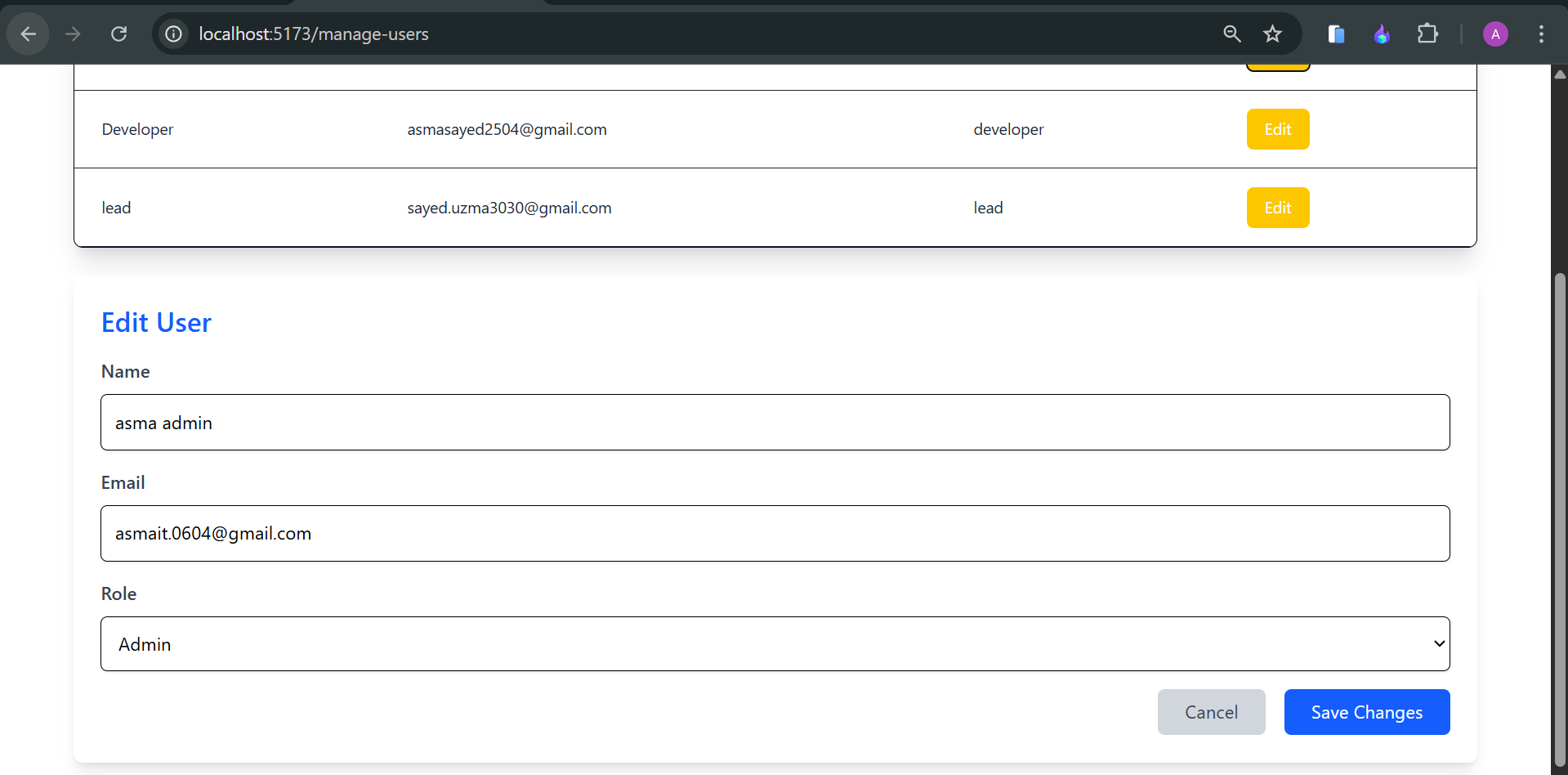
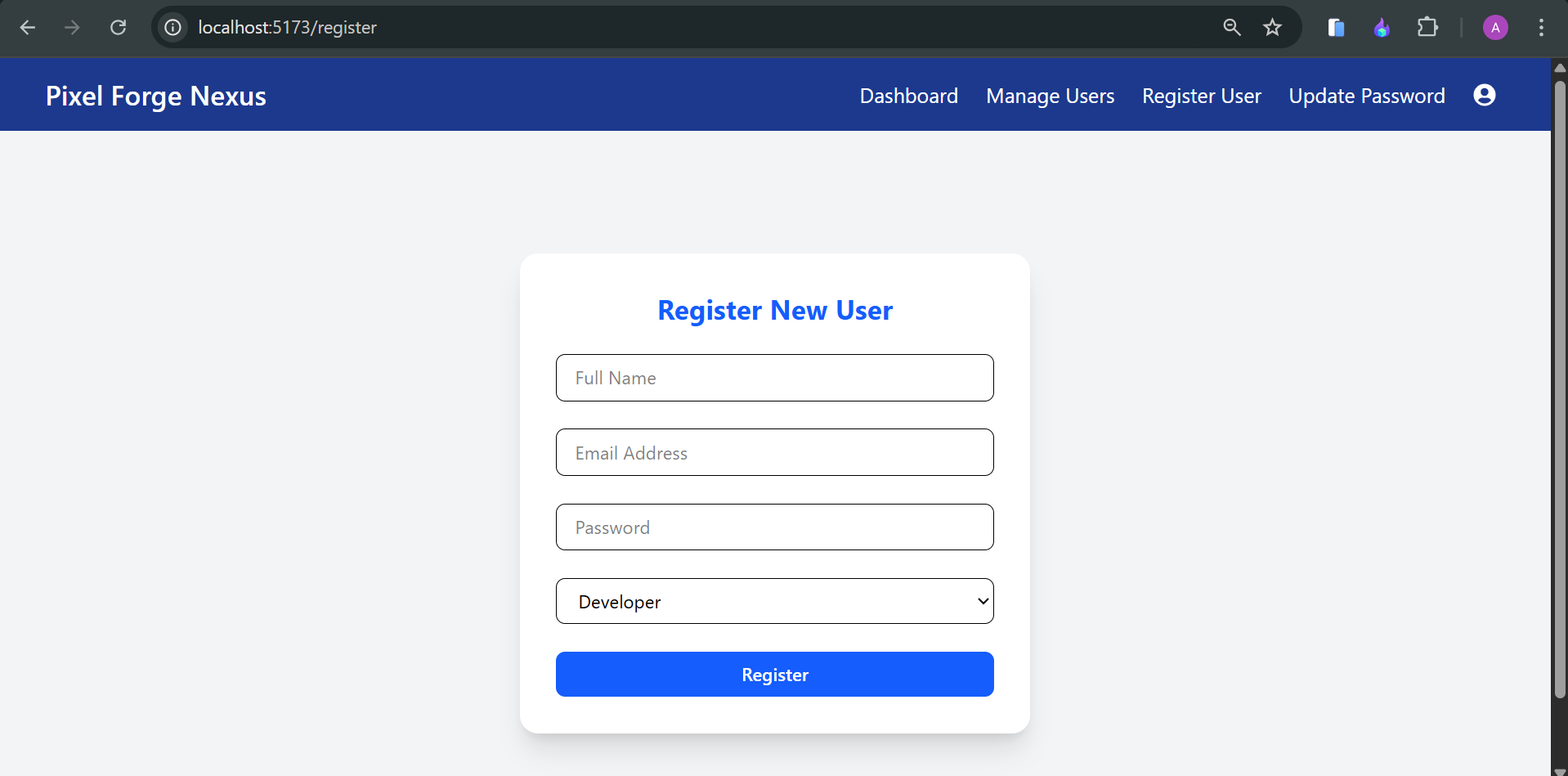
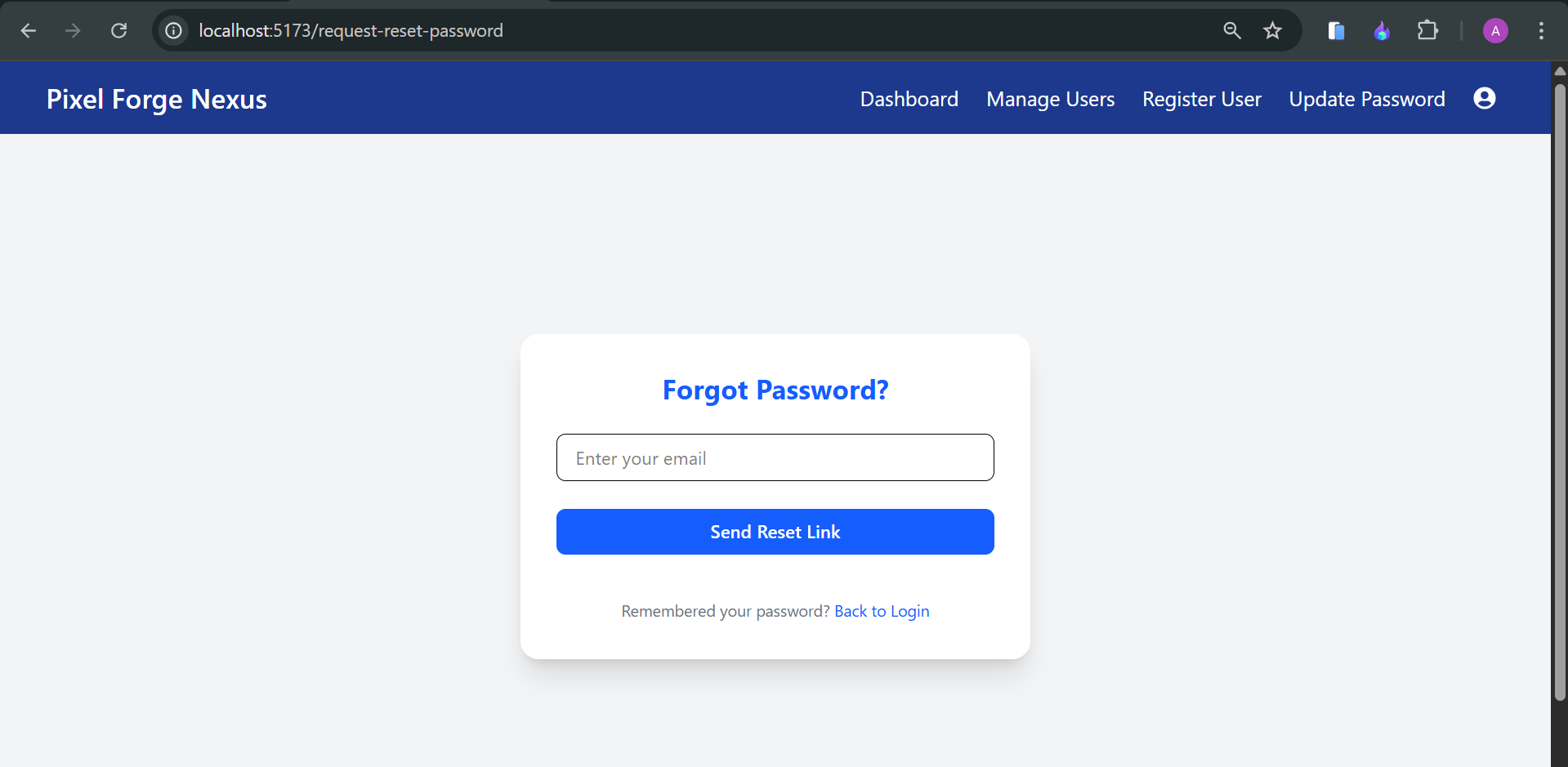
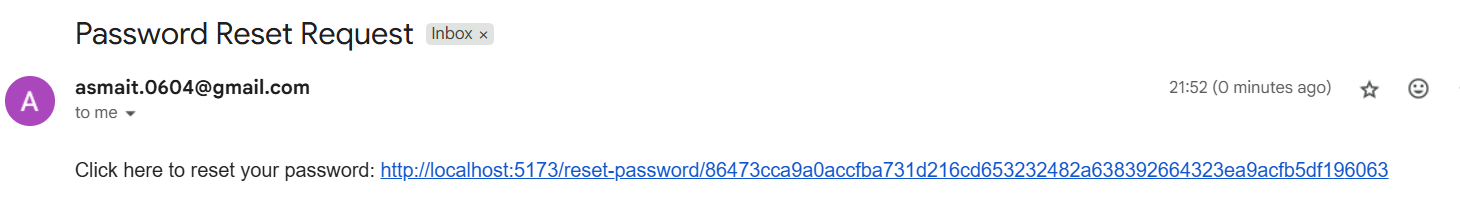
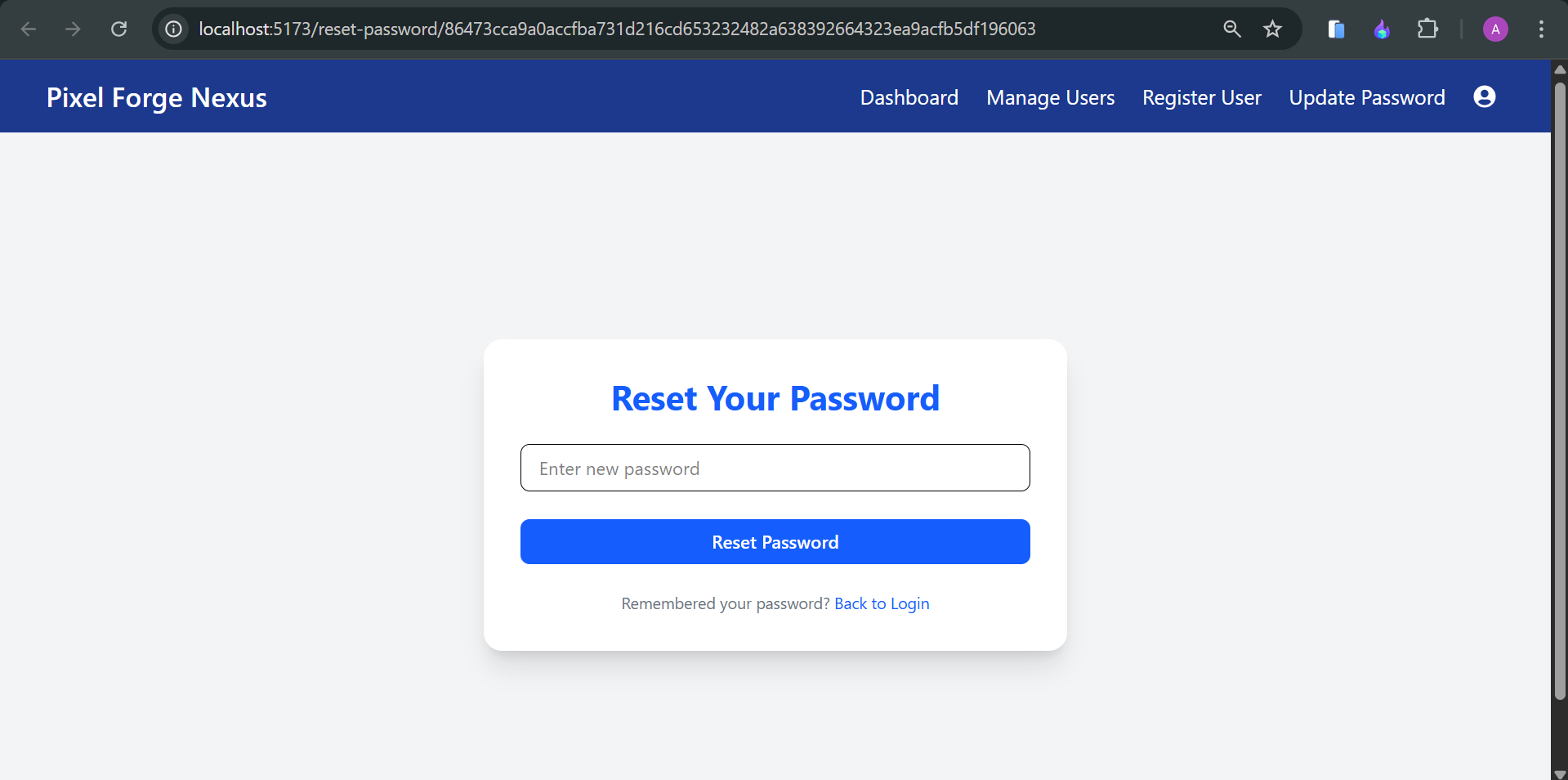
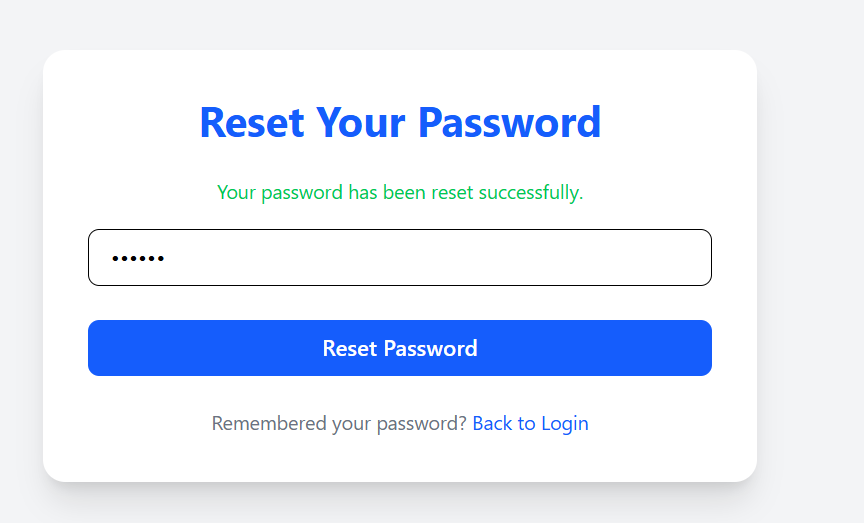




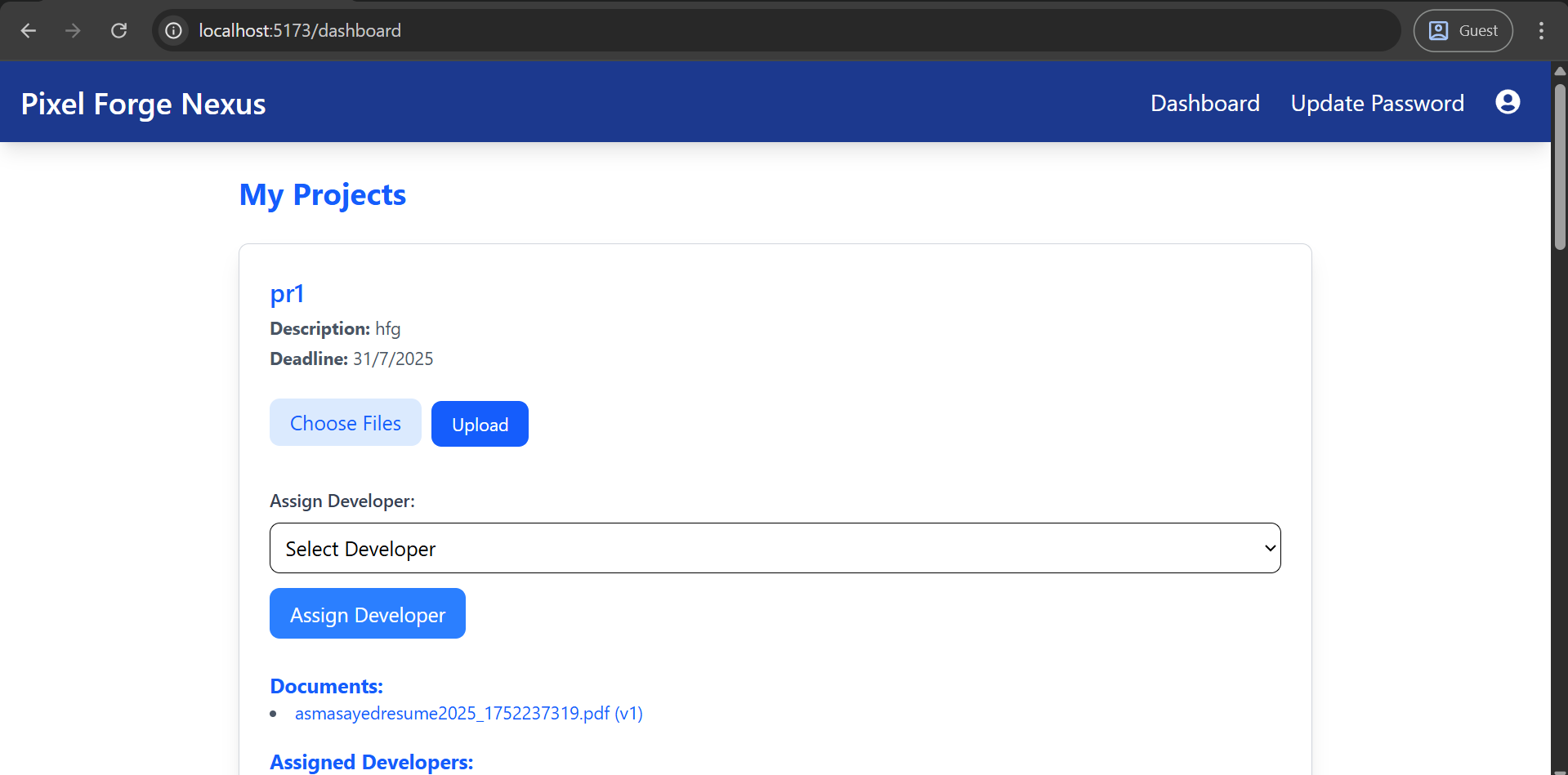


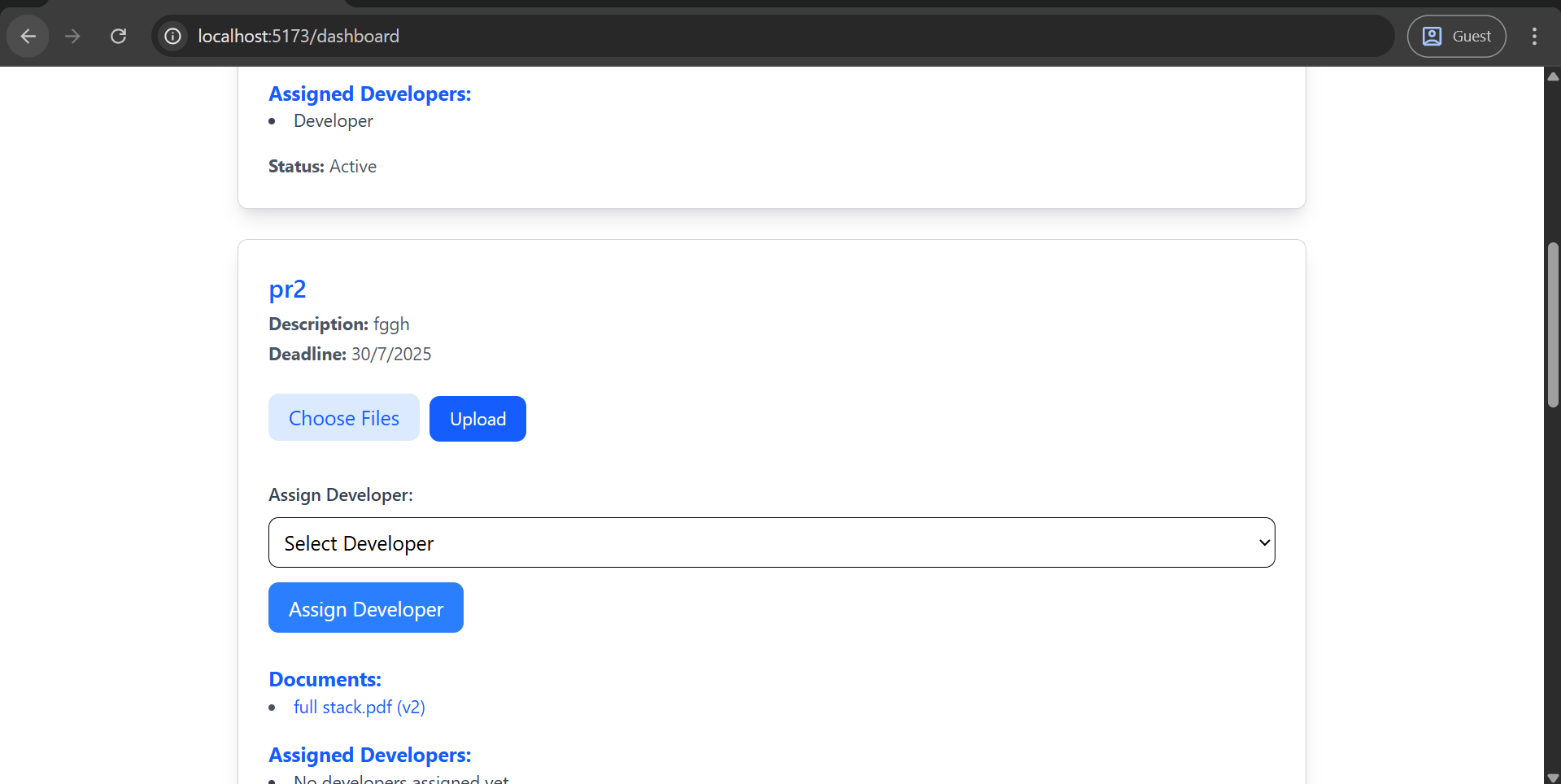
Admin dashboard -

Lead dashboard –





Developer dashboard -

