Technical Specifications

AI-Powered LaTeX CV Generator

1. Technology Stack

1.1 Frontend

- **Framework**: React.js
- **State Management**: Redux or Context API
- **UI Library**: Material-UI or Tailwind CSS
- **Form Management**: Formik with Yup validation
- **HTTP Client**: Axios
- **PDF Viewer**: React-PDF
- **Build Tool**: Vite

1.2 Backend

- **Framework**: Node.js with Express
- **API Architecture**: RESTful
- **Authentication**: None for MVP (stateless)
- **Validation**: Joi or express-validator

1.3 AI Integration

- **Primary AI Service**: OpenAI GPT-3.5-turbo API
- **Fallback Service**: Claude 2.1 or Mistral 7B
- **Prompt Management**: Server-side prompt templates

1.4 LaTeX Processing

- **LaTeX Engine**: pdfLaTeX
- **Processing Library**: Node-LaTeX or LaTeX.js
- **Template System**: Handlebars for template variables

1.5 Deployment

- **Frontend Hosting**: Vercel or Netlify
- **Backend Hosting**: Render, Railway, or Fly.io
- **CI/CD**: GitHub Actions

2. API Specifications

2.1 Internal APIs

2.1.1 CV Generation Endpoint

- **Endpoint**: `/api/generate`
- **Method**: POST

- **Request Body**: JSON object containing all CV data
- **Response**: JSON with LaTeX code and operation status
- **Status Codes**:
  - 200: Success
  - 400: Invalid input
  - 500: Server error
  - 503: AI service unavailable

2.1.2 PDF Generation Endpoint

- **Endpoint**: `/api/pdf`
- **Method**: POST
- **Request Body**: JSON with LaTeX code
- **Response**: PDF file or error message
- **Status Codes**:
  - 200: Success with PDF
  - 400: Invalid LaTeX
  - 500: Compilation error

2.1.3 Template Listing Endpoint

- **Endpoint**: `/api/templates`
- **Method**: GET
- **Response**: JSON array of available templates
- **Status Codes**:
  - 200: Success
  - 500: Server error

2.2 External API Integration

2.2.1 OpenAI API

- **API Version**: Latest stable
- **Endpoint**: `https://api.openai.com/v1/chat/completions`
- **Authentication**: API key (environment variable)
- **Model**: gpt-3.5-turbo
- **Max Tokens**: 2048
- **Temperature**: 0.3
- **Error Handling**: Retry logic, fallback to alternative service

3. Data Models

3.1 CV Data Model

```
{
  personalInfo: {
    name: String,
    title: String,
    email: String,
    phone: String,
    location: String,
    linkedIn: String,
    github: String,
    website: String,
    summary: String
  },
  education: [{
    institution: String,
    degree: String,
    field: String,
    startDate: String,
    endDate: String,
    gpa: String,
    honors: String,
    courses: String
  }],
  experience: [{
    company: String,
    position: String,
    startDate: String,
    endDate: String,
    location: String,
    description: String,
    bullets: [String]
  }],
  skills: {
    technical: [String],
    languages: [{ name: String, level: String }],
    certifications: [String],
    soft: [String]
  },
  projects: [{
    name: String,
    description: String,
    technologies: [String],
    url: String,
    startDate: String,
    endDate: String
  }],
  additional: {
    publications: [String],
    conferences: [String],
    volunteer: [String],
    awards: [String],
    references: [String]
  }
}
```

3.2 Template Model

```
{
  id: String,
  name: String,
  preview: String,
  description: String,
  latexTemplate: String
}
```

4. Database

- No persistent database for MVP
- Session storage using:
    - Redis for server-side caching (optional)
    - localStorage for client-side data persistence during session

5. AI Implementation

5.1 Prompt Engineering

```
You are a LaTeX expert creating professional CVs. Format the following information into clea

Personal Information:
{personalInfo}

Education:
{education}

Work Experience:
{experience}

Skills:
{skills}

Projects:
{projects}

Additional Information:
{additional}

Provide only the complete LaTeX code with no explanations or markdown.
```

5.2 Response Processing

- JSON parsing of AI response
- Validation of LaTeX structure
- Error detection and correction
- Template variable replacement

6. LaTeX Implementation

6.1 Base Template Structure

```latex
\documentclass[11pt,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage{geometry}
\usepackage{hyperref}
\usepackage{enumitem}
\usepackage{fontawesome5}
% Additional packages

\begin{document}
% CV Content from AI
\end{document}
```

6.2 PDF Generation Pipeline

1. Receive LaTeX code from AI
2. Insert code into template
3. Create temporary file
4. Execute LaTeX compilation
5. Stream PDF to client
6. Clean up temporary files

7. Security Considerations

7.1 Input Sanitization

- HTML entity encoding
- LaTeX special character escaping
- Command injection prevention

7.2 API Security

- Rate limiting
- API key rotation schedule
- Request validation

7.3 Infrastructure Security

- Environment variable encryption

- HTTPS enforcement
- CSP implementation

8. Performance Optimizations

8.1 Frontend

- Code splitting
- Lazy loading
- Memoization of expensive operations
- Service worker for offline capabilities

8.2 Backend

- Request caching
- Response compression
- Connection pooling
- Stateless design for horizontal scaling

8.3 LaTeX Processing

- Template precompilation
- Worker threads for compilation
- Output caching

9. Monitoring and Logging

9.1 Application Monitoring

- Error tracking with Sentry
- Performance monitoring
- API usage tracking

9.2 Logging

- Request logs
- Error logs
- AI interaction logs
- LaTeX compilation logs

10. Testing Strategy

10.1 Unit Tests

- React component tests with Jest and React Testing Library
- API endpoint tests with Jest and Supertest
- LaTeX generation tests

10.2 Integration Tests

- End-to-end form submission tests
- API integration tests
- PDF generation tests

10.3 Load Testing

- Concurrent user simulation
- API load testing
- PDF generation stress testing

11. Deployment Pipeline

1. Code commit
2. CI trigger
3. Run tests
4. Build frontend assets
5. Deploy to staging
6. Run integration tests
7. Deploy to production

This technical specification provides the implementation details for the AI-Powered LaTeX CV Generator. It should be used in conjunction with the Functional Specifications document.