# BGHS Alumni Website - Requirements Document

**Project Name:** BGHS Alumni Website  
**Organization:** Barasat Peary Charan Sarkar Government High School Alumni Association  
**Document Version:** 1.0  
**Date:** December 2024  
**Prepared By:** Business Analyst

## Table of Contents

1. [Project Overview](#X1c555f901308e0ca1fe83531c356e1a10b9c010)
2. [User Roles and Permissions](#Xc2f8806957c30c98ba4c49e86b6b85df237a3a7)
3. [Functional Requirements](#Xe7303fedc1caa0f98a54d3d573fddb57a70f32c)
4. [Non-Functional Requirements](#Xa059f3edd26bb652ad8373400c47e55bfc01b65)
5. [Technical Architecture](#X8f3f2959c8de0886e187879b786fd8dbaa82fd3)
6. [Integration Requirements](#X335103f96106d920965fc0c9ba383be00873014)
7. [Deployment Requirements](#X619d49d57f85a683d3ab3f19c5a81283b967c33)
8. [Appendices](#Xf8d5bfe00f4731c59609712ea967b645311d6e9)

## 1. Project Overview

### 1.1 Purpose

The BGHS Alumni Website serves as a comprehensive digital platform to connect alumni from Barasat Peary Charan Sarkar Government High School, facilitate networking, manage events, share memories, and support the school through donations.

### 1.2 Scope

This document outlines the complete requirements for developing a modern, scalable alumni website that serves multiple user types with varying levels of access and functionality.

### 1.3 Technology Stack

* **Frontend:** Next.js 14 with App Router
* **Backend:** Supabase (PostgreSQL + Auth)
* **Styling:** Tailwind CSS
* **File Storage:** Cloudflare R2
* **Email Service:** SendGrid
* **Payment Processing:** Stripe
* **Deployment:** Vercel

### 1.4 Key Features

* User registration and authentication
* Alumni directory with search capabilities
* Event management and registration
* Blog system with commenting
* Photo gallery with upload functionality
* Donation management system
* Admin panel for content and user management
* Multi-language support (English/Bengali)

## 2. User Roles and Permissions

### 2.1 User Role Definitions

#### 2.1.1 Public User

* **Description:** Unauthenticated visitors
* **Access Level:** Limited public content only
* **Primary Functions:** View landing page, public events, blog posts

#### 2.1.2 Alumni Member

* **Description:** Basic registered alumni
* **Access Level:** Standard member features
* **Primary Functions:** View directory, edit profile, register for events, comment on blog

#### 2.1.3 Alumni Premium

* **Description:** Premium members with enhanced features
* **Access Level:** All alumni features plus premium content
* **Primary Functions:** Download directory, access premium content

#### 2.1.4 Content Moderator

* **Description:** Users responsible for content moderation
* **Access Level:** Content management permissions
* **Primary Functions:** Moderate comments, edit public content

#### 2.1.5 Event Manager

* **Description:** Users responsible for event management
* **Access Level:** Event creation and management
* **Primary Functions:** Create events, manage registrations, send notifications

#### 2.1.6 Content Creator

* **Description:** Users who create content
* **Access Level:** Content creation permissions
* **Primary Functions:** Create blog posts, upload media, edit content

#### 2.1.7 Donation Manager

* **Description:** Users responsible for donation campaigns
* **Access Level:** Donation management permissions
* **Primary Functions:** Manage campaigns, view donations, generate reports

#### 2.1.8 Super Admin

* **Description:** System administrators
* **Access Level:** Full system access
* **Primary Functions:** All system functions, user management, analytics

### 2.2 Permission Matrix

| Permission | Public | Alumni | Premium | Moderator | Event Mgr | Creator | Donation Mgr | Super Admin |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| View Landing | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| View Directory | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Edit Profile | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| View Events | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Register Events | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| View Blog | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Comment Blog | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Access Premium | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Download Directory | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Moderate Comments | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| Edit Public Content | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| Create Events | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ |
| Manage Events | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ |
| Send Notifications | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ |
| Create Blog | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ |
| Edit Blog | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ |
| Upload Media | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ |
| View Donations | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ |
| Manage Campaigns | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ |
| Generate Reports | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ |
| Manage Users | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Manage Roles | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Access Admin | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| View Analytics | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |

## 3. Functional Requirements

### 3.1 Authentication & User Management

#### UC-001: User Registration

**Actor:** Public User  
**Precondition:** User is not authenticated  
**Priority:** High

**Main Flow:** 1. User navigates to registration page 2. User fills out registration form with: - First Name (required) - Last Name (required) - Middle Name (optional) - Email (required) - Password (required, min 8 chars, mixed case, numbers, special chars) - Year When Left BGHS (required) - Grade When Left (10th/12th/Other) 3. System validates input 4. System creates user account with is\_approved = false 5. System sends confirmation email 6. User account awaits admin approval

**Alternative Flow:** - 3a. Validation fails → Show error messages - 4a. Email already exists → Show error message

**Post-Condition:** User account created and pending approval

#### UC-002: User Login

**Actor:** Registered User  
**Precondition:** User has valid account  
**Priority:** High

**Main Flow:** 1. User navigates to login page 2. User enters email and password 3. System validates credentials 4. System checks if user is approved 5. System redirects to dashboard

**Alternative Flow:** - 3a. Invalid credentials → Show error message - 4a. User not approved → Show pending approval message

**Post-Condition:** User is authenticated and redirected to appropriate dashboard

#### UC-003: Forgot Password

**Actor:** Registered User  
**Precondition:** User has valid account  
**Priority:** Medium

**Main Flow:** 1. User clicks “Forgot Password” 2. User enters email address 3. System generates OTP 4. System sends OTP via email 5. User enters OTP 6. System validates OTP 7. User sets new password 8. System updates password

**Alternative Flow:** - 5a. OTP expires → User requests new OTP - 6a. Invalid OTP → Show error message

**Post-Condition:** User password is reset and user can log in

#### UC-004: Admin Password Reset

**Actor:** Super Admin  
**Precondition:** Admin is authenticated  
**Priority:** Medium

**Main Flow:** 1. Admin navigates to user management 2. Admin selects user 3. Admin clicks “Reset Password” 4. System generates random password 5. System updates user password 6. System sets email\_confirm = true 7. System sets must\_change\_password = true 8. System emails new password to user

**Post-Condition:** User receives new password and must change it on first login

#### UC-005: User Profile Management

**Actor:** Authenticated User  
**Precondition:** User is logged in  
**Priority:** High

**Main Flow:** 1. User navigates to profile page 2. User views current profile information 3. User edits profile fields: - Personal information - Professional details - Contact information - Social media links 4. System validates changes 5. System updates profile 6. System shows success message

**Post-Condition:** User profile is updated with new information

### 3.2 Alumni Directory

#### UC-006: View Alumni Directory

**Actor:** Authenticated User  
**Precondition:** User is logged in  
**Priority:** High

**Main Flow:** 1. User navigates to directory page 2. System displays list of approved alumni 3. User can search by name, batch year, profession, location 4. User can filter by various criteria 5. User clicks on alumni profile to view details

**Alternative Flow:** - 1a. Public user → Show teaser with login prompt

**Post-Condition:** User can view and search alumni information

#### UC-007: Search Alumni

**Actor:** Authenticated User  
**Precondition:** User is logged in  
**Priority:** Medium

**Main Flow:** 1. User enters search criteria 2. System searches across: - Full name - Batch year - Profession - Company - Location 3. System displays matching results 4. User can refine search criteria

**Post-Condition:** User sees filtered alumni results

### 3.3 Events Management

#### UC-008: View Events

**Actor:** Any User  
**Precondition:** None  
**Priority:** High

**Main Flow:** 1. User navigates to events page 2. System displays upcoming events 3. User can filter by category, date, location 4. User clicks on event for details 5. User can register for event (if authenticated)

**Post-Condition:** User can view and potentially register for events

#### UC-009: Create Event

**Actor:** Event Manager/Super Admin  
**Precondition:** User has event creation permissions  
**Priority:** Medium

**Main Flow:** 1. User navigates to event creation page 2. User fills event details: - Title, description, date, time - Location, category, max attendees - Event image 3. System validates input 4. System creates event 5. System notifies relevant users

**Post-Condition:** New event is created and visible to users

#### UC-010: Register for Event

**Actor:** Authenticated User  
**Precondition:** User is logged in  
**Priority:** High

**Main Flow:** 1. User views event details 2. User clicks “Register” 3. System checks availability 4. System creates registration 5. System updates attendee count 6. System sends confirmation

**Alternative Flow:** - 3a. Event full → Add to waitlist - 3b. User already registered → Show current status

**Post-Condition:** User is registered for the event

### 3.4 Blog System

#### UC-011: View Blog Posts

**Actor:** Any User  
**Precondition:** None  
**Priority:** Medium

**Main Flow:** 1. User navigates to blog page 2. System displays published posts 3. User can filter by category, tags 4. User clicks post to read full content 5. User can view comments and likes

**Post-Condition:** User can read blog content

#### UC-012: Create Blog Post

**Actor:** Content Creator/Super Admin  
**Precondition:** User has content creation permissions  
**Priority:** Medium

**Main Flow:** 1. User navigates to blog creation page 2. User writes post content: - Title, content, excerpt - Category, tags, featured image 3. User saves as draft or publishes 4. System validates content 5. System creates blog post 6. System notifies moderators (if published)

**Post-Condition:** New blog post is created

#### UC-013: Comment on Blog Post

**Actor:** Authenticated User  
**Precondition:** User is logged in  
**Priority:** Low

**Main Flow:** 1. User reads blog post 2. User scrolls to comments section 3. User writes comment 4. System validates comment 5. System adds comment to post 6. System notifies post author

**Post-Condition:** Comment is added to the blog post

### 3.5 Photo Gallery

#### UC-014: View Photo Gallery

**Actor:** Any User  
**Precondition:** None  
**Priority:** High

**Main Flow:** 1. User navigates to gallery page 2. System displays approved photos in masonry layout 3. User can filter by category 4. User clicks photo to view full size 5. User can navigate between photos 6. User can zoom, rotate, and navigate with keyboard

**Post-Condition:** User can view and interact with photos

#### UC-015: Upload Photos

**Actor:** Content Creator/Super Admin  
**Precondition:** User has upload permissions  
**Priority:** Medium

**Main Flow:** 1. User navigates to upload page 2. User selects photo file 3. User adds metadata: - Title, description, category 4. System validates file type and size 5. System uploads to Cloudflare R2 6. System generates thumbnail 7. System stores metadata in database 8. System shows success message

**Post-Condition:** Photo is uploaded and available in gallery

#### UC-016: Manage Photo Categories

**Actor:** Super Admin  
**Precondition:** Admin is authenticated  
**Priority:** Low

**Main Flow:** 1. Admin navigates to category management 2. Admin creates new category 3. Admin edits existing categories 4. Admin sets display order 5. System updates category list

**Post-Condition:** Photo categories are updated

### 3.6 Donations System

#### UC-017: View Donation Causes

**Actor:** Any User  
**Precondition:** None  
**Priority:** Medium

**Main Flow:** 1. User navigates to donations page 2. System displays active donation causes 3. User can view cause details and progress 4. User can make donation (if authenticated)

**Post-Condition:** User can view donation opportunities

#### UC-018: Make Donation

**Actor:** Authenticated User  
**Precondition:** User is logged in  
**Priority:** Medium

**Main Flow:** 1. User selects donation cause 2. User enters donation amount 3. User adds optional message 4. User chooses anonymous option 5. System processes payment 6. System updates cause progress 7. System sends confirmation

**Post-Condition:** Donation is processed and recorded

#### UC-019: Manage Donation Campaigns

**Actor:** Donation Manager/Super Admin  
**Precondition:** User has donation management permissions  
**Priority:** Low

**Main Flow:** 1. User navigates to campaign management 2. User creates new campaign: - Title, description, target amount - Category, image, active status 3. User monitors campaign progress 4. User generates reports 5. User can deactivate campaigns

**Post-Condition:** Donation campaigns are managed

### 3.7 Admin Functions

#### UC-020: User Management

**Actor:** Super Admin  
**Precondition:** Admin is authenticated  
**Priority:** High

**Main Flow:** 1. Admin navigates to user management 2. System displays all users with pagination 3. Admin can: - View user details - Approve/reject users - Change user roles - Reset passwords - Edit user profiles 4. Admin can search and filter users 5. System updates user status

**Post-Condition:** User accounts are managed

#### UC-021: Content Moderation

**Actor:** Content Moderator/Super Admin  
**Precondition:** User has moderation permissions  
**Priority:** Medium

**Main Flow:** 1. User navigates to moderation panel 2. System displays content requiring review: - Blog comments - Photo uploads - User profiles 3. User reviews content 4. User approves or rejects content 5. System updates content status 6. System notifies content creator

**Post-Condition:** Content is moderated

#### UC-022: System Analytics

**Actor:** Super Admin  
**Precondition:** Admin is authenticated  
**Priority:** Low

**Main Flow:** 1. Admin navigates to analytics dashboard 2. System displays key metrics: - User registrations - Event attendance - Blog views and engagement - Donation totals - Photo uploads 3. Admin can filter by date range 4. Admin can export reports

**Post-Condition:** Analytics data is available

### 3.8 Dashboard Features

#### UC-023: User Dashboard

**Actor:** Authenticated User  
**Precondition:** User is logged in  
**Priority:** High

**Main Flow:** 1. User logs in successfully 2. System redirects to dashboard 3. System displays personalized content: - Recent events - Blog posts - Photo gallery highlights - Profile completion status 4. User can access quick actions based on permissions

**Post-Condition:** User sees personalized dashboard

#### UC-024: Admin Dashboard

**Actor:** Super Admin  
**Precondition:** Admin is authenticated  
**Priority:** High

**Main Flow:** 1. Admin logs in successfully 2. System redirects to admin dashboard 3. System displays admin-specific content: - User management - Content moderation - System analytics - Quick actions 4. Admin can access all management functions

**Post-Condition:** Admin sees management dashboard

## 4. Non-Functional Requirements

### 4.1 Performance Requirements

* **Page Load Time:** < 3 seconds for all pages
* **Image Optimization:** Automatic compression and lazy loading
* **Database Response:** < 500ms for standard queries
* **Concurrent Users:** Support for 1000+ simultaneous users
* **File Upload:** Support for images up to 10MB

### 4.2 Security Requirements

* **HTTPS:** All communications encrypted
* **Authentication:** Secure session management
* **Authorization:** Role-based access control
* **Data Protection:** Row Level Security (RLS) in database
* **Input Validation:** All user inputs sanitized
* **File Security:** Secure file upload and storage
* **Password Policy:** Minimum 8 characters with complexity requirements

### 4.3 Scalability Requirements

* **Horizontal Scaling:** Support for multiple server instances
* **Database Scaling:** Optimized queries and indexing
* **CDN Integration:** Global content delivery
* **Caching:** Redis for session and data caching
* **Load Balancing:** Automatic traffic distribution

### 4.4 Usability Requirements

* **Responsive Design:** Mobile-first approach
* **Accessibility:** WCAG 2.1 AA compliance
* **Browser Support:** Chrome, Firefox, Safari, Edge (latest 2 versions)
* **Navigation:** Intuitive menu structure
* **Search:** Fast and accurate search functionality
* **Multi-language:** English and Bengali support

### 4.5 Reliability Requirements

* **Uptime:** 99.9% availability
* **Backup:** Daily automated backups
* **Recovery:** Point-in-time recovery capability
* **Monitoring:** Real-time system monitoring
* **Error Handling:** Graceful error handling and logging

## 5. Technical Architecture

### 5.1 Frontend Architecture

* **Framework:** Next.js 14 with App Router
* **Language:** TypeScript
* **Styling:** Tailwind CSS
* **State Management:** React Hooks and Context
* **Authentication:** Supabase Auth
* **Image Optimization:** Next.js Image component
* **Form Handling:** React Hook Form

### 5.2 Backend Architecture

* **Database:** Supabase PostgreSQL
* **API:** Next.js API Routes
* **Authentication:** Supabase Auth
* **File Storage:** Cloudflare R2
* **Email Service:** SendGrid
* **Payment Processing:** Stripe

### 5.3 Database Schema

#### 5.3.1 Core Tables

* **profiles** - User information and profiles
* **user\_roles** - Role definitions and permissions
* **user\_role\_assignments** - User role assignments

#### 5.3.2 Content Tables

* **events** - Event information
* **event\_registrations** - Event attendance
* **blog\_posts** - Blog content
* **blog\_comments** - Blog interactions
* **blog\_likes** - Blog engagement

#### 5.3.3 Media Tables

* **gallery\_photos** - Photo gallery
* **photo\_categories** - Photo organization

#### 5.3.4 Donation Tables

* **donation\_causes** - Donation campaigns
* **donations** - Donation records

#### 5.3.5 System Tables

* **password\_reset\_otps** - Password recovery
* **newsletters** - Newsletter subscriptions

### 5.4 Security Architecture

* **Row Level Security (RLS):** Database-level access control
* **JWT Tokens:** Secure authentication
* **CORS Configuration:** Cross-origin request handling
* **Rate Limiting:** API request throttling
* **Input Sanitization:** XSS and injection prevention

## 6. Integration Requirements

### 6.1 Email Service Integration

* **Provider:** SendGrid
* **Templates:** Transactional email templates
* **Features:** OTP delivery, notifications, confirmations
* **Configuration:** Environment-based setup

### 6.2 Payment Processing Integration

* **Provider:** Stripe
* **Features:** Secure payment processing
* **Webhooks:** Payment status updates
* **Security:** PCI compliance

### 6.3 File Storage Integration

* **Provider:** Cloudflare R2
* **Features:** Image storage and CDN
* **Optimization:** Automatic thumbnail generation
* **Security:** Signed URLs for private content

### 6.4 Analytics Integration

* **Provider:** Google Analytics (optional)
* **Features:** User behavior tracking
* **Privacy:** GDPR compliant
* **Custom Events:** Business-specific metrics

## 7. Deployment Requirements

### 7.1 Environment Configuration

* **Production:** Vercel deployment
* **Staging:** Vercel preview deployments
* **Development:** Local development environment
* **Environment Variables:** Secure configuration management

### 7.2 Monitoring and Logging

* **Error Tracking:** Real-time error monitoring
* **Performance Monitoring:** Application performance metrics
* **Log Management:** Centralized logging system
* **Alerting:** Automated alert notifications

### 7.3 Backup and Recovery

* **Database Backups:** Daily automated backups
* **File Backups:** Regular file system backups
* **Recovery Testing:** Monthly recovery tests
* **Disaster Recovery:** Business continuity planning

## 8. Appendices

### 8.1 Glossary

* **Alumni:** Former students of BGHS
* **RLS:** Row Level Security
* **OTP:** One-Time Password
* **CDN:** Content Delivery Network
* **API:** Application Programming Interface

### 8.2 References

* Next.js Documentation
* Supabase Documentation
* Tailwind CSS Documentation
* SendGrid API Documentation
* Stripe API Documentation

### 8.3 Change Log

* **Version 1.0:** Initial requirements document
* **Date:** December 2024

**Document Status:** Draft  
**Next Review Date:** January 2025  
**Approval Required:** Project Stakeholders

*This document contains confidential and proprietary information. Distribution is restricted to authorized personnel only.*