School Management System - Project Flow Documentation

1. Initial Setup and Database

Database Configuration

- MySQL database named `school_management` must be running
- Required environment variables:

  ```
  DB_HOST=localhost
  DB_USER=root
  DB_PASSWORD=your_password
  DB_NAME=school_management
  PORT=3001
  JWT_SECRET=your_secret_key
  ```

Project Structure

```
SchoolWebsite/
  src/
     public/
        login.html        # Main login page
        registration.html # Student registration
        images/          # Static images
        css/             # Stylesheets
        js/              # Client-side scripts
     student/            # Student dashboard and features
     admin/              # Admin dashboard and features
     database/           # Database initialization and models
     routes/             # API routes
  server.js               # Main server file
```

2. Server Startup Flow

Start the Server

```bash
# First, ensure no existing server is running
pkill -f "node server.js"

# Start the server
node server.js
```

Server Initialization Sequence

- Load environment variables
- Connect to MySQL database
- Initialize database tables
- Start Express server on port 3001
- Set up middleware and routes

3. User Access Flow

A. First-Time Access

1. User visits `http://localhost:3001`
2. Server redirects to `/login.html`
3. User sees login page with three role options:

- Student
- Parent
- Admin

B. Login Process

1. User selects role and enters credentials
2. System validates against appropriate table:
   - Students/Parents: `users` table
   - Admins: `admin_users` table
3. On successful login:
   - JWT token is generated
   - User data is stored in localStorage
   - Redirected to appropriate dashboard

C. Dashboard Access

1. **Student Dashboard** (`/student/dashboard.html`)
   - View academic records
   - Check attendance
   - View assignments
   - Access library
   - View upcoming events
2. **Parent Dashboard** (`/parent/dashboard.html`)
   - View child's academic progress
   - Check attendance
   - View report cards
   - Access school calendar
3. **Admin Dashboard** (`/admin/dashboard.html`)
   - Manage student records
   - Handle attendance
   - Manage academic records
   - System configuration

4. Protected Routes and Authentication

Public Routes (No authentication required)

- `/login.html`
- `/registration.html`
- `/api/auth/login`
- Static files (`/images/`, `/css/`, `/js/`)

Protected Routes (Authentication required)

- All dashboard pages
- All API endpoints except login
- Student and parent features

5. Error Handling

Authentication Errors

- Invalid credentials → Redirect to login
- Expired token → Redirect to login
- Invalid role → Access denied

Database Errors

- Connection issues → Server error
- Query failures → Appropriate error messages

Client-Side Errors

- Form validation errors
- API request failures
- Network issues

6. Security Measures

Authentication

- JWT token-based authentication
- Role-based access control
- Password hashing with bcrypt

Data Protection

- Input validation
- SQL injection prevention
- XSS protection
- CSRF protection

7. Testing the System

Start the Server

```
node server.js
```

Access Points

- Main URL: `http://localhost:3001`
- Login: `http://localhost:3001/login.html`
- Student Dashboard: `http://localhost:3001/student/dashboard.html`
- Admin Dashboard: `http://localhost:3001/admin/dashboard.html`

Test Users

- Create test users in database with appropriate roles
- Test login with different roles
- Verify dashboard access
- Test protected routes

8. Troubleshooting

Common Issues

- Port 3001 already in use → Kill existing process
- Database connection failed → Check credentials
- Authentication errors → Verify JWT secret
- Static files not loading → Check file paths

Debug Steps

- Check server logs
- Verify database connection
- Test API endpoints
- Check browser console for errors