# School Vaccination Portal Documentation – Gaurav Rahate(2024tm93277@wilp.bits-pilani.ac.in)

## Table of Contents

---

## System Overview

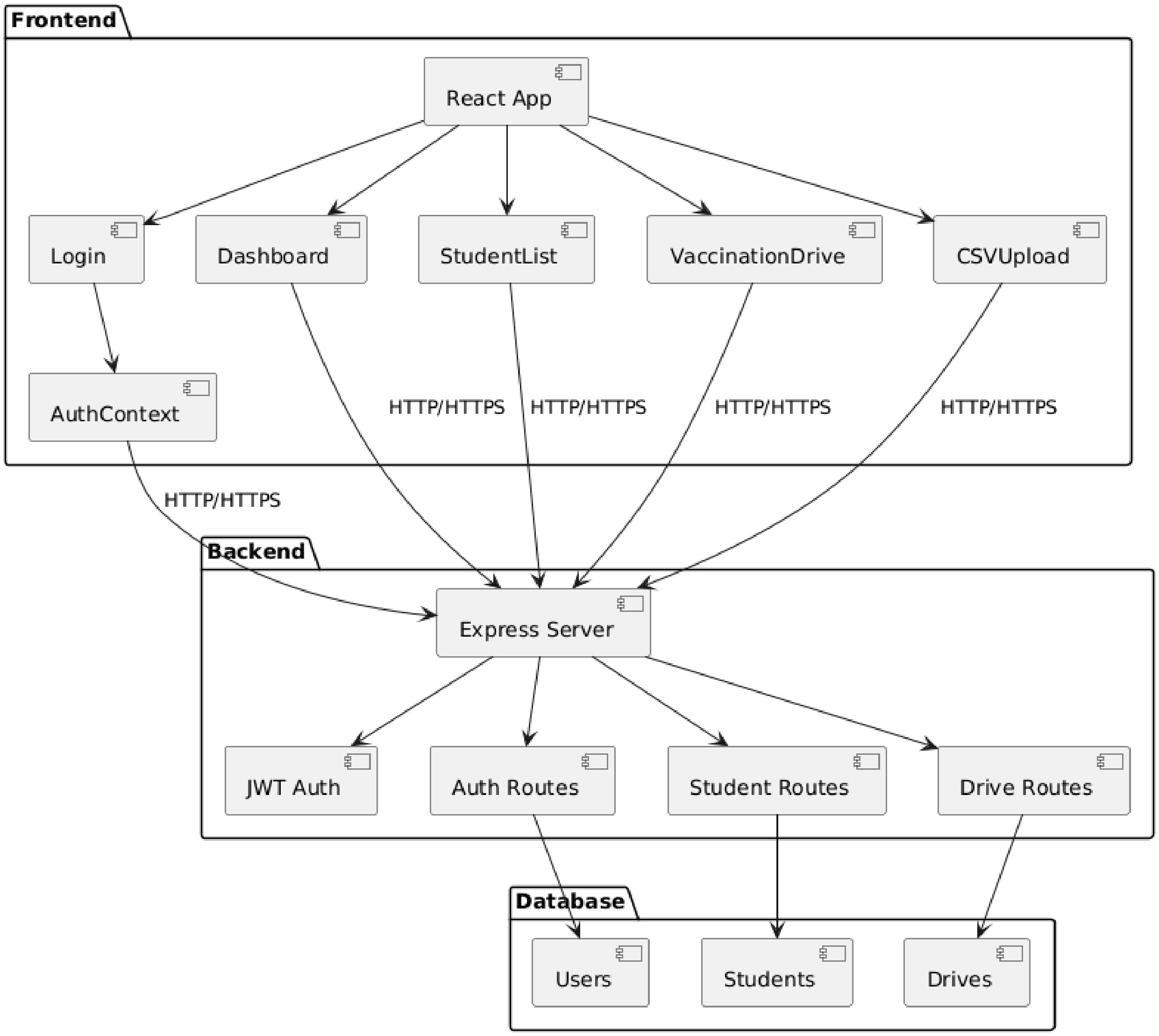
### Project Architecture

The School Vaccination Portal is built using: - **Frontend**: React.js with Material-UI - **Backend**: Node.js with Express - **Database**: MongoDB - **Authentication**: JWT

### Technology Stack

- **Frontend**:
  - React.js
  - Material-UI
  - Axios for API calls
  - React Router for navigation
  - Context API for state management
- **Backend**:
  - Node.js
  - Express.js
  - MongoDB with Mongoose
  - JWT for authentication
  - Multer for file uploads
  - CSV parser for bulk imports

## Full Application Architecture

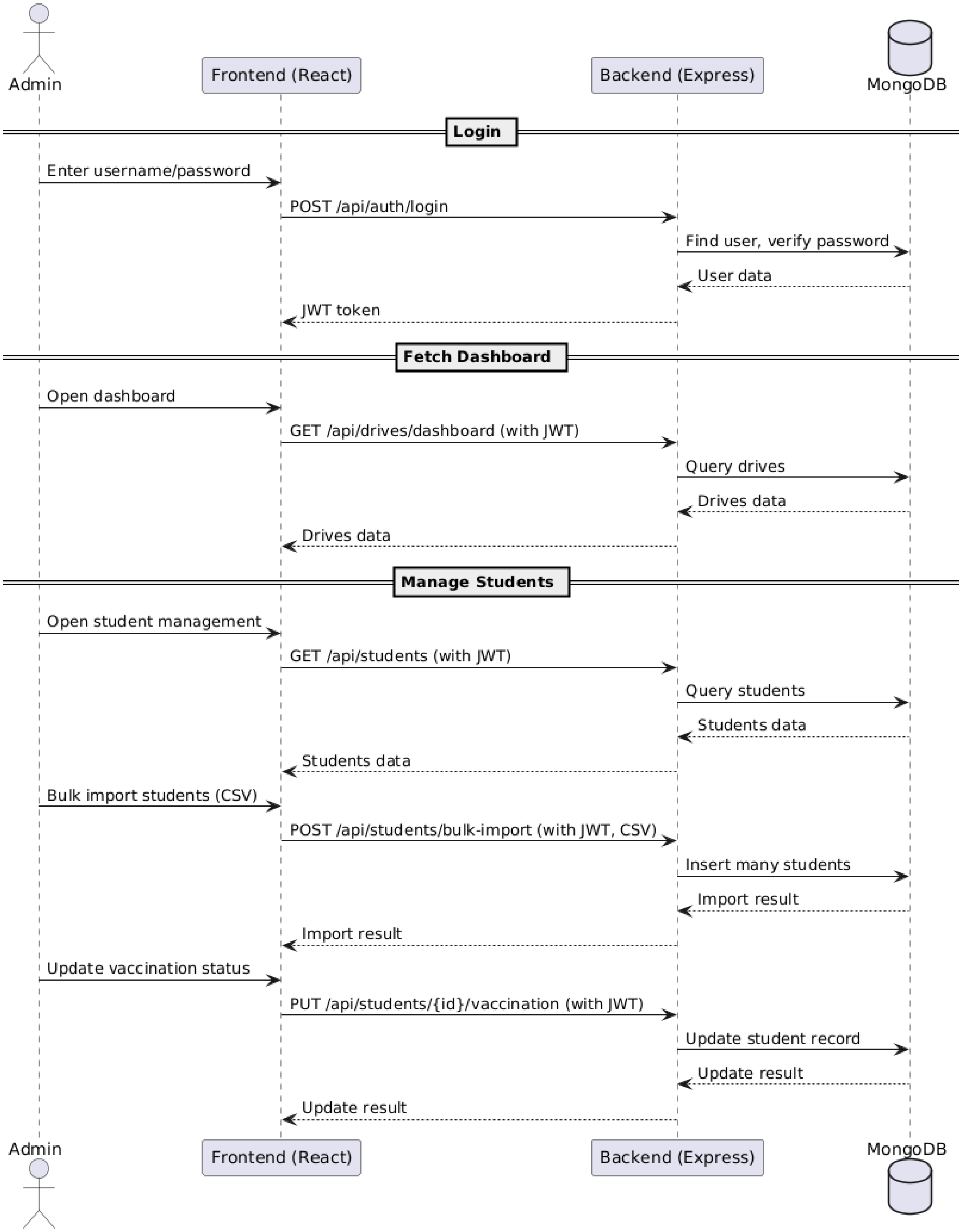


## Security Features

- JWT-based authentication
- Password hashing
- Input validation
- CORS protection
- File upload validation

# Application Architecture

## Frontend-Backend Interaction



### Frontend Structure

```
frontend/
├── src/
│   ├── components/
│   │   ├── Dashboard.js
│   │   ├── StudentManagement.js
│   │   ├── VaccinationDriveManagement.js
│   │   └── Reports.js
│   ├── context/
│   │   └── AuthContext.js
│   ├── pages/
│   │   ├── Login.js
│   │   └── Dashboard.js
│   └── App.js
```

### Backend Structure

```
backend/
├── routes/
│   ├── auth.js
│   ├── students.js
│   ├── drives.js
│   └── reports.js
├── models/
│   ├── User.js
│   ├── Student.js
│   └── VaccinationDrive.js
└── server.js
```

---

## API Documentation

### Authentication Endpoints

#### *Login*

```
POST /api/auth/login
Content-Type: application/json

{
  "username": "admin",
  "password": "admin123"
}

// Response
{
  "token": "jwt_token_here"
}
```

---

## Student Endpoints

### *Get All Students*

```
GET /api/students
Authorization: Bearer <token>

// Query Parameters
?name=<name>
?class=<class>
?studentId=<studentId>
?vaccinationStatus=<status>

// Response
{
  "students": [
    {
      "_id": "123",
      "studentId": "STU001",
      "name": "John Doe",
      "class": "Grade 1",
      "dateOfBirth": "2010-05-15",
      "vaccinations": [
        {
          "vaccineName": "MMR",
          "driveId": "456",
          "date": "2024-03-15",
          "status": "completed"
        }
      ]
    }
  ]
}
```

### *Bulk Import Students*

```
POST /api/students/bulk-import
Authorization: Bearer <token>
Content-Type: multipart/form-data

file: <csv_file>

// CSV Format
studentId,name,class,dateOfBirth
STU001,John Doe,Grade 1,2010-05-15
STU002,Jane Smith,Grade 2,2010-06-20

// Response
{
  "message": "Successfully imported 2 students",
  "imported": 2,
  "failed": 0
}
```

#### *Update Vaccination Status*

```
PUT /api/students/:id/vaccination
Authorization: Bearer <token>
Content-Type: application/json

{
  "driveId": "drive_id",
  "status": "completed"
}

// Response
{
  "message": "Vaccination status updated",
  "student": {
    "_id": "123",
    "vaccinations": [
      {
        "vaccineName": "MMR",
        "driveId": "456",
        "date": "2024-03-15",
        "status": "completed"
      }
    ]
  }
}
```

#### *Delete Student*

```
DELETE /api/students/:id
Authorization: Bearer <token>

// Response
{
  "message": "Student deleted successfully"
}
```

---

### Vaccination Drive Endpoints

#### *Get All Drives*

```
GET /api/drives
Authorization: Bearer <token>

// Query Parameters
?status=<status>
?startDate=<date>
```

```
?endDate=<date>

// Response
{
  "drives": [
    {
      "_id": "456",
      "vaccineName": "MMR",
      "date": "2024-04-15",
      "availableDoses": 100,
      "applicableClasses": ["Grade 1", "Grade 2"],
      "status": "scheduled"
    }
  ]
}
```

### *Create Drive*

```
POST /api/drives
Authorization: Bearer <token>
Content-Type: application/json

{
  "vaccineName": "MMR",
  "date": "2024-04-15",
  "availableDoses": 100,
  "applicableClasses": ["Grade 1", "Grade 2"]
}

// Response
{
  "message": "Drive created successfully",
  "drive": {
    "_id": "456",
    "vaccineName": "MMR",
    "date": "2024-04-15",
    "availableDoses": 100,
    "applicableClasses": ["Grade 1", "Grade 2"],
    "status": "scheduled"
  }
}
```

### *Get Dashboard Metrics*

```
GET /api/drives/dashboard
Authorization: Bearer <token>

// Response
{
  "totalDrives": 5,
  "drivesNext30Days": [
    {
      "_id": "456",
      "vaccineName": "MMR",
      "date": "2024-04-15",
      "applicableClasses": ["Grade 1", "Grade 2"]
    }
  ],
  "upcomingDrives": [
    {
      "_id": "789",
      "vaccineName": "Polio",
      "date": "2024-05-01",
      "availableDoses": 80
    }
  ]
}
```

---

## Database Schema

### Users Collection

```
{
  _id: ObjectId,
  username: String,      // Required, unique
  password: String,      // Required, hashed
  role: String,         // Enum: ['admin']
  createdAt: Date,      // Auto-generated
  updatedAt: Date       // Auto-updated
}
```

### Students Collection

```
{
  _id: ObjectId,
  studentId: String,    // Required, unique
  name: String,         // Required
  class: String,        // Required
  dateOfBirth: Date,    // Required
  vaccinations: [{
    vaccineName: String,
    driveId: ObjectId,  // Reference to VaccinationDrive
    date: Date,
    status: String      // Enum: ['pending', 'completed']
  }],
  createdAt: Date,      // Auto-generated
  updatedAt: Date       // Auto-updated
}
```

### VaccinationDrive Collection

```
{
  _id: ObjectId,
  vaccineName: String,  // Required
  date: Date,          // Required
  availableDoses: Number,
  applicableClasses: [String],
  status: String,      // Enum: ['scheduled', 'completed', 'cancelled']
  createdAt: Date,     // Auto-generated
  updatedAt: Date      // Auto-updated
}
```

---

## UI/UX Documentation

### Design Principles

- Clean and intuitive interface
- Mobile-first responsive design
- Consistent color scheme
- Clear navigation

### Component Library

- Material-UI components
- Custom styled components
- Reusable form elements
- Data tables

### User Flows

1. **Admin Login**
   - Sign in with admin credentials
   - Access dashboard
   - Manage students and drives
2. **Student Management**
   - View all students
   - Add students individually
   - Bulk import students via CSV
   - Update vaccination status
3. **Vaccination Drive Management**
   - Create new drives
   - View upcoming drives
   - Track drive metrics
4. **Reports**
   - Generate vaccination reports
   - Filter by vaccine type
   - Export to Excel

---

# Setup Guide

## Prerequisites

- Node.js (v14 or higher)
- MongoDB (v4.4 or higher)
- npm or yarn

## Installation Steps

1. **Clone Repository**

```
git clone https://github.com/your-org/school-vaccination-portal.git
cd school-vaccination-portal
```

2. **Install Dependencies**

```
# Install root dependencies
npm install

# Install frontend dependencies
cd frontend
npm install

# Install backend dependencies
cd ../backend
npm install
```

3. **Environment Configuration**

```
# Frontend (.env)
REACT_APP_API_URL=http://localhost:5000
REACT_APP_ENV=development

# Backend (.env)
PORT=5000
MONGODB_URI=mongodb://localhost:27017/vaccination-portal
JWT_SECRET=your-secret-key
NODE_ENV=development
```

4. **Start Application**

```
# Start backend
cd backend
npm run dev

# Start frontend
cd frontend
npm start
```

## Testing

```
# Frontend tests
cd frontend
npm test
```

```
# Backend tests
cd backend
npm test
```

---

## API Testing Guide

### Postman Collection

1. Import the Postman collection from `docs/postman_collection.json`
2. Set up environment variables:
   - `base_url`: http://localhost:5000
   - `token`: JWT token from login response

### Example API Tests

1. **Login**
   - Request: POST /api/auth/login
   - Body: `{ "username": "admin", "password": "admin123" }`
   - Expected: 200 OK with JWT token
2. **Get Students**
   - Request: GET /api/students
   - Headers: `Authorization: Bearer <token>`
   - Expected: 200 OK with student list
3. **Create Drive**
   - Request: POST /api/drives
   - Headers: `Authorization: Bearer <token>`
   - Body: Drive details
   - Expected: 201 Created with drive details

---

## Assumptions and Limitations

### Assumptions

1. Single admin user system
2. CSV format for bulk student import
3. 15-day advance notice for vaccination drives
4. No overlapping drives for same classes
5. Students can’t be deleted if they have completed vaccinations

### Limitations

1. No parent portal
2. No email notifications
3. No mobile app

4. No real-time updates
5. Limited to one school

### Future Enhancements

1. Multi-school support
2. Parent portal
3. Email notifications
4. Mobile app
5. Real-time updates
6. Advanced reporting