Bustracking Backend API Documentation

Base URL: `https://pretty-dogs-dress.loca.lt` (or `http://localhost:4000`)

---

1. Authentication

1-1. Admin Login

Endpoint: `POST /admin/login`
Description: Authenticate an admin user.
Request Body:

```
{
  "user_name": "admin",
  "password": "password123"
}
```

Response: Success message, token, and user details.

---

2. Super Admin

2-1. Register Super Admin

Endpoint: `POST /admin/superadmin/register`
Description: Create a new super admin account.
Request Body:

```
{
  "username": "superadmin",
  "email_address": "admin@example.com",
  "password": "securepassword"
}
```

Response: Success message and account details.

---

3. General Settings

3-1. Add Settings

Endpoint: `POST /admin/general_settings`
Description: Upload global settings and logo.
Request Body (FormData):

- `site_name` : "Bus App"
- `mobile_no` : "1234567890"
- `email_id` : "support@example.com"
- `support_no` : "0987654321"
- `google_api_key` : "key123"
- `logo` : [File] **Response:** Created settings object.

3-2. Get Settings

Endpoint: `GET /admin/general_settings`
Description: Retrieve current settings.
Request Body: None
Response: Settings details.

3-3. Update Settings

Endpoint: `PATCH /admin/general_settings/:id`
Description: Modify existing settings.
Request Body: Fields to update (e.g., `site_name`).
Response: Updated settings.

3-4. Delete Settings

Endpoint: `DELETE /admin/general_settings/:id`
Description: Remove settings.
Request Body: None
Response: Success message.

---

4. School Management

4-1. Add School

Endpoint: `POST /admin/college_school`
Description: Register a new school/college.
Request Body:

```
{
  "name_of_institution": "Tech School",
  "address": "123 Main St",
  "city": "Metropolis",
  "state": "NY",
  "zip_code": "10001",
  "phone_number": "555-0101",
  "email_address": "school@test.com",
  "website": "www.school.com",
  "username": "schooladmin",
  "password": "password123"
}
```

Response: Created school details.

4-2. Get Schools

Endpoint: `GET /admin/college_school`
Description: List all registered schools.
Request Body: None
Response: Array of schools.

4-3. Edit School

Endpoint: `PATCH /admin/college_school/:id`
Description: Update school details.
Request Body: Fields to update.
Response: Updated school object.

4-4. Delete School

Endpoint: `DELETE /admin/college_school/:id`
Description: Remove a school.
Request Body: None
Response: Success message.

---

5. Transport Services

5-1. Add Transport Service

Endpoint: `POST /admin/busdetails`
Description: Register a transport service provider.
Request Body:

```
{
  "name_of_transport_service": "FastTrans",
  "email_address": "trans@test.com",
  "password": "password123"
}
```

Response: Created service details.

5-2. Get Transport Services

Endpoint: `GET /admin/busdetails`
Description: List all transport services.
Request Body: None
Response: List of services.

5-3. Get Single Service

Endpoint: `GET /admin/busdetails/:id`
Description: Get details of one transport service.
Request Body: None
Response: Service object.

5-4. Edit Transport Service

Endpoint: `PATCH /admin/busdetails/:id`
Description: Update transport service details.
Request Body: Fields to update.
Response: Updated service object.

5-5. Delete Transport Service

Endpoint: `DELETE /admin/busdetails/:id`
Description: Remove a transport service.
Request Body: None
Response: Success message.

5-6. Add Credentials

Endpoint: `POST /admin/transport-credentials`
Description: Add login credentials for transport.
Request Body:

```
{
  "school_id": "school_123",
  "email": "login@test.com",
  "password": "password123"
}
```

Response: Created credentials.

5-7. Get Credentials

Endpoint: `GET /admin/transport-credentials`
Description: List all credentials.
Request Body: None
Response: List of credentials.

5-8. Edit Credentials

Endpoint: `PATCH /admin/transport-credentials/:id`
Description: Update credentials.
Request Body: Fields to update.
Response: Updated credentials.

5-9. Delete Credentials

Endpoint: `DELETE /admin/transport-credentials/:id`
Description: Remove credentials.
Request Body: None
Response: Success message.

---

6. School-Transport Linking

6-1. Link School to Transport

Endpoint: `POST /admin/transportchoose`
Description: Select a transport service for a school.
Request Body:

```
{
  "serviceName": "FastTrans",
```

```
  "school_id": "school_123"
}
```

Response: Linking confirmation.

6-2. Get Link

Endpoint: `GET /admin/transportchoose`
Description: Get transport service for a school.
Params: `?school_id=...`
Response: Linked service details.

6-3. Find Schools by Transport

Endpoint: `POST /admin/fetch_school`
Description: List schools using a specific transport service.
Request Body:

```
{
  "transport_service": "FastTrans"
}
```

Response: List of schools.

---

7. Bus Management

7-1. Add Bus

Endpoint: `POST /admin/add_bus`
Description: Add a new bus.
Request Body:

```
{
  "localid": "school_123",
  "bus_id": "BUS-101",
  "bus_name": "Yellow 1",
  "status": "Active"
}
```

Response: Created bus details.

7-2. Get Buses

Endpoint: `POST /admin/getbus`
Description: List buses for a school.
Request Body:

```
{
  "localid": "school_123"
}
```

Response: List of buses.

7-3. Edit Bus

Endpoint: `PATCH /admin/edit_bus`
Description: Update bus info.
Request Body:

```
{
  "id": "bus_mongo_id",
  "bus_name": "Yellow 2"
}
```

Response: Updated bus details.

7-4. Delete Bus

Endpoint: `POST /admin/delete_bus`
Description: Remove a bus.
Request Body:

```
{
  "id": "bus_mongo_id"
}
```

Response: Success message.

---

8. Driver Management

8-1. Add Driver

Endpoint: `POST /admin/add_driver`
Description: Register a new driver.
Request Body:

```
{
  "driver_name": "Bob",
  "driver_email": "bob@test.com",
  "driver_password": "pass",
  "localid": "school_123"
}
```

Response: Created driver.

8-2. Get Drivers

Endpoint: `POST /admin/getdrivers`
Description: List drivers.
Header: `x-localid: school_123`
Response: List of drivers.

8-3. Edit Driver

Endpoint: `PATCH /admin/edit_driver`
Description: Update driver details.
Request Body:

```
{
  "driver_id": "driver_mongo_id",
  "driver_name": "Bob Updated"
}
```

Response: Updated driver.

8-4. Delete Driver

Endpoint: `POST /admin/delete_driver`
Description: Remove a driver.
Request Body:

```
{
  "id": "driver_mongo_id"
}
```

Response: Success message.

---

9. Student Management

9-1. Add Student

Endpoint: `POST /admin/add_student`
Description: Register a student.
Request Body:

```
{
  "student_name": "Alice",
  "student_id": "STU101",
  "parent_email": "parent@test.com",
  "parent_password": "pass",
  "localid": "school_123"
}
```

Response: Created student.

9-2. Get Students

Endpoint: `POST /admin/getstudents`
Description: List students.
Header: `x-localid: school_123`
Response: List of students.

9-3. Edit Student

Endpoint: `PATCH /admin/edit_student`
Description: Update student info.
Request Body: `id` , `student_name` , etc.
Response: Updated student.

9-4. Delete Student

Endpoint: `POST /admin/delete_student`
Description: Remove a student.
Request Body:

```
{
  "id": "student_mongo_id"
}
```

Response: Success message.

9-5. Mark Attendance

Endpoint: `POST /admin/student_attendance`
Description: Log student attendance.
Request Body:

```
{
  "student_id": "STU101",
  "route_id": "R1",
  "status": "Present"
}
```

Response: Attendance record.

---

10. Parent Management

10-1. Add Parent

Endpoint: `POST /admin/add_parent`
Description: Register a parent account.
Request Body:

```
{
  "father_name": "John Sr",
  "student_id": "STU101",
  "parent_email": "parent@test.com",
  "parent_password": "pass"
}
```

Response: Created parent.

10-2. Edit Parent

Endpoint: `PATCH /admin/edit_parent`
Description: Update parent details.
Request Body: `id` plus fields to update.
Response: Updated parent.

10-3. Delete Parent

Endpoint: `POST /admin/delete_parent`
Description: Remove a parent.
Request Body:

```
{
  "id": "parent_mongo_id"
}
```

Response: Success message.

---

11. Route Management

11-1. Add Route

Endpoint: `POST /admin/add_route`
Description: Create a bus route.
Request Body:

```
{
  "route_name": "Route A",
  "route_id": "R-A",
  "localid": "school_123"
}
```

Response: Created route.

11-2. Get Routes

Endpoint: `POST /admin/getroutedetails`
Description: List all routes.
Request Body: None
Response: List of routes.

11-3. Edit Route

Endpoint: `PATCH /admin/edit_route`
Description: Modify a route.
Request Body: `id` , `route_name` .
Response: Updated route.

11-4. Delete Route

Endpoint: `POST /admin/deleteroutedetails`
Description: Remove a route.
Request Body: `id` .
Response: Success message.

11-5. Assign Route

Endpoint: `POST /admin/asign_route`
Description: Assign driver/bus to route.
Request Body:

```
{
  "route_id": "R-A",
  "driver_id": "driver_mongo_id"
}
```

Response: Assignment details.

---

12. Station Management

12-1. Add Station

Endpoint: `POST /admin/add_stations`
Description: Add a bus stop.
Request Body:

```
{
  "route_id": "R-A",
  "bus_stopname": "Central Park",
  "lattitude": "40.785",
  "longitude": "-73.968"
}
```

Response: Created station.

12-2. Get Stations

Endpoint: `POST /admin/getstations`
Description: List all stations.
Request Body: None
Response: List of stations.

12-3. Edit Station

Endpoint: `PATCH /admin/edit_stations`
Description: Update station.
Request Body: `id` , `bus_stopname` .
Response: Updated station.

12-4. Delete Station

Endpoint: `POST /admin/delete_stations`
Description: Remove station.
Request Body: `id` .
Response: Success message.

---

13. Waiting Locations

13-1. Get Locations

Endpoint: `GET /admin/waiting_locations`
Description: List waiting locations.
Response: List of locations.

13-2. Add Location

Endpoint: `POST /admin/waiting_locations`
Description: Add a new waiting point.
Request Body:

```
{
  "latitude": "12.34",
  "longitude": "56.78",
  "driver_id": "driver_1"
}
```

Response: Created location.

13-3. Update Location

Endpoint: `PUT /admin/waiting_locations/:id`
Description: Modify location.
Response: Updated location.

13-4. Delete Location

Endpoint: `DELETE /admin/waiting_locations/:id`
Description: Remove location.
Response: Success message.

---

14. Notifications

14-1. Add Notification

Endpoint: `POST /admin/add_notifications`
Description: Send a push notification.
Request Body:

```
{
  "title": "Bus Late",
```

```
  "message": "Arriving in 10 mins"
}
```

Response: Confirmation.

14-2. Log Notification

Endpoint: `POST /admin/notifications`
Description: Save notification to log.
Request Body:

```
{
  "title": "Alert",
  "group": "Parents"
}
```

Response: Created log.

14-3. Get Logs

Endpoint: `GET /admin/notifications`
Description: View history.
Response: List of logs.

---

15. Comments & Public APIs

15-1. Post Comment (Admin)

Endpoint: `POST /admin/newcomment`
Description: Admin comment.
Request Body:

```
{
  "name": "admin",
  "password": "pass",
  "userId": "1",
  "comment": "Nice work"
}
```

Response: Comment details.

15-2. Get Comments (Admin)

Endpoint: `GET /admin/comments`
Description: Fetch comments. **Response:** List of comments.

15-3. Post Comment (Public)

Endpoint: `POST /api/v1/newcomment`
Description: Public user comment.
Request Body: `comment`, `userId`.

15-4. Get Comments (Public)

Endpoint: `GET /api/v1/comments`
Description: Fetch comments publicly.

15-5. Public Login Check

Endpoint: `GET /api/v1/login`
Description: Check credentials.
Request Body: `user_name`, `password`.

15-6. Delete Comment (Admin) (Previously Missing)

Endpoint: `DELETE /admin/comment/:id`
Description: Admin delete comment.
Params: `id` of comment.
Response: Success message.

15-7. Delete Comment (Public) (Previously Missing)

Endpoint: `DELETE /api/v1/comment/:id`
Description: Public delete comment.
Params: `id` of comment.
Response: Success message.

---

16. Mobile App APIs

16-1. Driver Login

Endpoint: `POST /mobile_api/v1/driver_login`
Description: App login for drivers.
Request Body:

```
{
  "email": "driver@test.com",
  "password": "pass"
}
```

Response: Auth token.

16-2. Parent Login

Endpoint: `POST /mobile_api/v1/parent_login`
Description: App login for parents.
Request Body:

```
{
  "email": "parent@test.com",
  "password": "pass"
}
```

Response: Auth token.

16-3. Get Route (Driver)

Endpoint: `POST /mobile_api/v1/fetch_Driver_route`
Description: Get assigned route.
Request Body:

```
{
  "driver_id": "driver_mongo_id"
}
```

Response: Route data.

16-4. Get Bus Stops

Endpoint: `POST /mobile_api/v1/fetch_pickupdrop_stations`
Description: Get stops for a route.
Request Body:

```
{
  "route_id": "R-A"
}
```

Response: List of stops.

16-5. Get Driver Info

Endpoint: `POST /mobile_api/v1/fetch_driver`
Description: Get driver profile.
Request Body:

```
{
  "driver_id": "driver_mongo_id"
}
```

Response: Driver details.

16-6. Mobile Fetch Route (Public) (Previously Missing)

Endpoint: `POST /mobile_api/v1/fetch_route`
Description: General fetch route.
Request Body:

```
{
  "driver_id": "driver_mongo_id"
}
```

16-7. Update Bus Stop Filter (Previously Missing)

Endpoint: `PATCH /mobile_api/v1/filter_bustop`
Description: Filter stops for student/parent.
Request Body:

```
{
  "student_id": "STU101",
  "station_id": "STAT_99"
}
```

Response: Updated filter.

Total Endpoints: 68