ACore API Manual — Comprehensive Guide (Frontend, Admin & Deployment)

This document is a systematic summary of all ACore API endpoints, access rules, request formats, and deployment guidelines for Frontend Devs, Admin Devs, Testers, and DevOps Engineers.

---

1. ⚙ Common Setup & Access Rules

- **Local Base URL (Dev):** http://localhost:8080
- **Admin Access:** JWT Token is mandatory for all /api/admin/** endpoints.P
- **Authorization Header:** Authorization: Bearer <token>
- **Content-Type (JSON):** application/json (For standard POST/PUT requests)
- **Content-Type (File Upload):** multipart/form-data (For Projects, Blogs, Job Applications)

API Route Prefix / Access Level / Notes

- /api/auth/** — Public — Login, Forgot/Reset Password.
- /api/public/** — Public — Contact Form, Job Application Submission.
- /api/chat/** — Public — Chatbot Q&A.
- /uploads/** — Public — To retrieve stored files (Images, Resumes).
- /api/admin/** — Admin Only — JWT Token required.

---

2. 🔐 Authentication & Password Flow

2.1 Login

- **Method:** POST
- **Endpoint:** /api/auth/login
- **Body (JSON):**

```
{
  "username": "user@example.com",
  "password": "userPassword"
}
```

- **Success Response:**

```
{
  "token": "JWT_STRING",
  "role": "ROLE_ADMIN"
```

}

- **Frontend action:** token ko secure storage me save karo (localStorage/sessionStorage/cookie as per security policy).

2.2 Forgot Password (OTP-based flow)

We use a 3-step OTP flow for password reset (recommended UX):

1. User requests an OTP to their registered email.
2. User enters OTP in frontend → verify it via /verify-otp.
3. Once OTP is verified, user submits new password via /reset-password (no OTP in this final request).

Endpoints & example payloads:

1️⃣ **Send OTP**

- **Method:** POST
- **Endpoint:** /api/auth/forgot-password
- **Body:**

{ "email": "test@gmail.com" }

- **Response:** { "message": "OTP sent to your email." }

2️⃣ **Verify OTP**

- **Method:** POST
- **Endpoint:** /api/auth/verify-otp
- **Body:**

{ "email": "test@gmail.com", "otp": "123456" }

- **Response:** { "message": "OTP verified successfully." }

3️⃣ **Reset Password (only email + newPassword)**

- **Method:** POST
- **Endpoint:** /api/auth/reset-password
- **Body:**

{ "email": "test@gmail.com", "newPassword": "new123" }

- **Notes:** No OTP field required here. Backend will **check that OTP was verified and still valid (expiry not passed)** — then update the password and clear stored OTP.
- **Response:** { "message": "Password reset successfully." }

✅ UX note: OTP must be short-lived (e.g., 5 minutes). Frontend should show three steps: enter email → enter OTP → set new password.

---

3. 🌐 Public Facing APIs (No Token Required)

Feature	Method	Endpoint	Content-Type	Request Body / Params
Contact Form	POST	/api/public/contact	application/json	{ fullName, email, phoneNumber, subject, message }
Chatbot Q&A	POST	/api/chat/send	(URL param)	?query=user's question
Job Application	POST	/api/public/applications	multipart/form-data	1. applicationData (JSON string), 2. resumeFile (file)

---

4. 🔒 Admin CRUD Operations (Requires Authorization: Bearer <token>)

4.1 Contact Inquiries & Job Applications (Admin)

- **List Inquiries:** GET /api/admin/inquiries (filter: ?status=NEW or ?type=hiring)
- **Change Status:** PUT /api/admin/inquiries/{id}/status?newStatus=READ
- **Delete Inquiry:** DELETE /api/admin/inquiries/{id}
- **List Job Applications:** GET /api/admin/applications
- **Delete Application:** DELETE /api/admin/applications/{id}

4.2 Job Management (JSON)

- **Create:** POST /api/admin/jobs — full job JSON (includes requirements[], responsibilities[])
- **List:** GET /api/admin/jobs
- **Update:** PUT /api/admin/jobs/{id} (same JSON)
- **Delete:** DELETE /api/admin/jobs/{id}

4.3 Testimonials (JSON)

- **Create:** POST /api/admin/testimonials — { quote, personName, personPosition }
- **List:** GET /api/admin/testimonials
- **Update:** PUT /api/admin/testimonials/{id}
- **Delete:** DELETE /api/admin/testimonials/{id}

4.4 News (Articles) (JSON)

- **Create:** POST /api/admin/news — { title, description, content, authorName, readTime (int), category }
- **List:** GET /api/admin/news
- **Get by ID:** GET /api/admin/news/{id}

- **Update:** PUT /api/admin/news/{id} (same JSON)
- **Delete:** DELETE /api/admin/news/{id} — expected 204 No Content

4.5 Projects & Blogs (Multipart/Form-data)

- **Projects Create:** POST /api/admin/projects — projectData (JSON string) + imageFile
- **Projects List:** GET /api/admin/projects
- **Projects Update/Delete:** PUT / DELETE /api/admin/projects/{id} (imageFile optional in PUT)
- **Blogs Create:** POST /api/admin/blogs — blogData (JSON string) + imageFile
- **Blogs List:** GET /api/admin/blogs
- **Blogs Update/Delete:** PUT / DELETE /api/admin/blogs/{id} (imageFile optional in PUT)

---

5. 🗂️ File Upload & Access Structure

- **Uploads Folder Structure:** uploads/projects/, uploads/blogs/, uploads/resumes/
- **Backend Storage:** store relative path in DB (e.g., projects/demo.jpg)
- **Frontend Access:** construct full URL: http://localhost:8080/uploads/<relative-path>
    - Example: http://localhost:8080/uploads/projects/demo.jpg

---

6. ⚠️ Common Errors & Troubleshooting

Error Code / Issue	Meaning	Frontend Action	Backend Check (Admin/DevOps)
403 Forbidden	Token Missing/Expired or Invalid Role	Re-login or check token refresh logic	JWT filter config, allowed public routes
404 Not Found	Incorrect ID in URL path	Verify URL/ID	Check Entity ID in DB
400 Bad Request	Invalid Request Body / JSON Format	Correct data types (e.g., readTime must be integer)	Add request validation
500 Server Error (File Upload)	File system error or path issue	Check file type/size	Verify FILE_UPLOAD_PATH_ENV and permissions (775)

---

7. ☁️ Deployment Guide (DevOps)

Area / Production Setting / Notes

- **Config File:** Use application-prod.properties — do not hardcode secrets.

- **Run Command:** java -jar acoreapi.jar --spring.profiles.active=prod
- **Environment Vars:** DB_USERNAME, JWT_SECRET_KEY_ENV, FILE_UPLOAD_PATH_ENV, etc. — set on server.
- **Linux Uploads Path:** e.g., /var/www/acore-uploads — set FILE_UPLOAD_PATH_ENV and ensure 775 permissions.
- **CORS Fix:** Centralized CORS bean in SecurityConfig.java — remove controller-level @CrossOrigin.
- **Email Fix:** Allow outbound port 587 or use transactional provider (SendGrid/Mailgun).

---

8. ✅ Quick Postman Testing Checklist

1. POST /api/auth/login → get token.
2. Add header Authorization: Bearer <token> to admin calls.
3. Test GET /api/admin/jobs (should return 200).
4. Test file uploads: POST /api/admin/projects with form-data: projectData (text) + imageFile (file).
5. Test job application submission: POST /api/public/applications form-data: applicationData + resumeFile.
6. Test forgot-password flow: POST /api/auth/forgot-password → POST /api/auth/verify-otp → POST /api/auth/reset-password.

---

9. Appendix — Examples & Tips

- Use multipart/form-data **only** where specified.
- Dates: use YYYY-MM-DD format.
- Always check API response codes (200, 201, 204, 400, 403, 404, 500) and handle them in frontend.
- OTP flow summary (again):
    1. /forgot-password sends OTP to email.
    2. /verify-otp verifies it.
    3. /reset-password accepts { email, newPassword } (no OTP) and resets password if OTP was previously verified and not expired.