Full Stack Developer Task: Job Portal with Flutter

Day 1: Backend Development (Node.js + Express + MongoDB)

1. **Authentication**:
    - Implement login and register with **JWT** for role-based access (**admin** and **user** roles).
2. **Candidate Management**:
    - Create an API to upload candidate details (name, email, contact, education, role, etc).
    - Add functionality to upload a candidate's profile image and resume.
3. **File Storage**:
    - Store uploaded images and resumes in **AWS S3**.
    - Save file URLs and candidate details in **MongoDB**.
4. **Deployment**:
    - Deploy the backend to **AWS EC2** or **Elastic Beanstalk**.

Day 2: Mobile App Development (Flutter)

1. **Login UI**:
    - Build a login screen with role-based navigation (Admin/User).
    - Make the registration page (**optional)**.
2. **Candidate Management**:
    - Create a UI for adding candidate details, including profile image and resume upload.
    - Display a candidate list fetched from the backend.
3. **Candidate Details View**:
    - Show full candidate details with their uploaded image and a button to view the resume (PDF or document viewer).
4. **Integration**:
    - Use **http package** to connect the Flutter app with the backend API.

Before submitting the project, make sure that both the mobile app and the server are thoroughly tested for functionality and performance. Ensure that all features, such as user authentication, file uploads, and candidate management, are working as expected. It is important to test the app on multiple devices and verify API responses. To assist you, a list of recommended tools is provided below for help.

Essential Tools for Project Development

Backend Tools:

1. **Node.js**: For server-side scripting and API development.
2. **Express**: To create RESTful APIs.
3. **Multer**: For handling file uploads (images and resumes).
4. **AWS SDK**: To upload images and resumes to **AWS S3**.
5. **jsonwebtoken (JWT)**: For authentication and role-based access control.
6. **MongoDB**: For storing candidate data and file URLs.
7. **Mongoose**: To interact with MongoDB using schemas.
8. **dotenv**: To manage environment variables (like AWS keys).
9. **bcrypt**: To hash user passwords for secure storage.

AWS Services:

1. **AWS S3**: To store uploaded files (images and resumes).
2. **AWS EC2**: To host the backend application.
3. **AWS Elastic Beanstalk**: For easier deployment and scaling.
4. **AWS IAM**: To manage permissions for secure S3 access.

Flutter Tools:

1. **Flutter SDK**: For building the mobile app.
2. **http**: To make API calls to the backend.
3. **image_picker**: To allow users to select and upload images.
4. **file_picker**: For selecting and uploading resumes (PDF or DOC).
5. **flutter_secure_storage**: To store authentication tokens securely on the device.
6. **provider**: For state management (optional but useful).
7. **url_launcher**: To open resume files (PDF or DOC) in external viewers.

Testing and Debugging Tools:

1. **Postman**: To test API endpoints.
2. **Flutter DevTools**: For debugging the mobile app.