Phase 1: Problem Understanding & Industry Analysis

- **Requirement Gathering**: Collect requirements from professors (rubric-based grading, general grading without rubric, feedback, re-evaluation requests, reporting).
- **Stakeholder Analysis**: Professors (define marking scheme, approve/reject AI grading), Students (submit answers, view grades), Admins (manage org, permissions), TAs (review flagged answers).
- **Business Process Mapping**:
    1. Student uploads answer → stored in Salesforce.
    2. Auto-grading triggered (via ML API/Einstein).
    3. Score, rationale, and confidence returned.
    4. Professors validate or override.
    5. Results published to students.
- **Industry-specific Use Case Analysis**: Similar to EdTech tools like Gradescope but integrated with Salesforce's workflow, audit trail, and reporting. Useful in schools, universities, and corporate training.
- **AppExchange Exploration**: Explore apps like "Grade Capture", "Document Automation", or "Einstein Analytics" dashboards to see reusable components.

Phase 2: Org Setup & Configuration

🔹 **Step 1: Sign Up for Salesforce Developer Edition**

1. Go to the signup page:
   🔗 https://developer.salesforce.com/signup
2. Fill in the details as shown:
    - First Name: *Mohd Umar*
    - Last Name: *Khan*
    - Job Title: **Developer**
    - Email: umar.22bce7693@vitapstudent.ac.in
    - Company: *Vellore Institute of Technology - Andhra Pradesh*
    - Country/Region: India
3. Click **Sign Me Up**.
4. Verified my email via the link Salesforce sends.

🔹 **Step 2: Log in to Salesforce Developer Org**

1. Once verified, go to:
   🔗 https://login.salesforce.com
2. Enter your registered email + password.
3. You'll be redirected to the **Salesforce Lightning Experience Dashboard**.

🔹 **Step 3: Explore the Org Setup**

1. Click the **gear icon (⚙️)** in the top-right corner.
2. Select **Setup** → You'll enter **Setup Home**.
   - Here you can configure apps, objects, security, and automation.

🔹 **Step 4: Do the Following Configuration Steps**

These steps align with your **Automated Answer Sheet Grading System**:

1. **Create a New App**
   - Navigate to **App Manager** → **New Lightning App**.
   - App Name: *Automated Grading System*.
   - Add Navigation Items (Submissions, Rubrics, Results).
2. **Create Custom Objects**
   - Go to **Object Manager** → **Create** → **Custom Object**.
   - Example objects:
     - **Answer Submission** (stores student answers).
     - **Marking Rubric** (criteria from professor).
     - **AutoGrade Result** (AI-generated marks + confidence).
3. **Create Custom Fields**
   - In each object, add fields like:
     - *Answer Submission*: Student Name (Lookup), Answer Text (Long Text), Exam Type (Picklist).
     - *Marking Rubric*: Criteria (Text), Marks (Number).
     - *AutoGrade Result*: Score (Number), Confidence % (Percent), Feedback (Rich Text).
4. **Create Tabs for Custom Objects**

- Go to **Tabs** in Setup.
- Create tabs for each custom object so users can access them from App Launcher.

5. **Assign Profiles / Permission Sets**
   - *Students*: Submit answers only.
   - *Professors*: Full CRUD access on Rubrics, Results, and Submissions.
   - *TAs*: Read/Update access for verification.

🔹 **Step 5: Document Your Work**

Take screenshots of:
- Salesforce Dashboard after login
- Created App (*Automated Grading System*)
- Custom Objects (Submissions, Rubrics, Results)
- Custom Fields inside objects
- Tabs added for navigation

📂 Save them in your repository under
`https://github.com/Maus-313/Automated_Answer_Sheet_Evaluation_System_Using_Salesforce/upload/master`

Phase 3: Data Modeling & Relationships

- **Objects**:
  - *Student* (Standard: Contact)
  - *Professor* (Standard: User)
  - *Answer Submission* (Custom)
  - *Marking Rubric* (Custom)
  - *AutoGrade Result* (Custom)
  - *Re-evaluation Request* (Custom)
- **Fields**: Answer text, score, confidence, feedback, rubric criteria, status.
- **Record Types**: Different exam types (MCQ, Short Answer, Essay).
- **Page Layouts / Compact Layouts**: Tailored for professors vs students.

- **Schema Builder**: Visualize object relationships.
- **Relationships**:
    - Student → Answer Submission (Lookup).
    - Answer Submission → AutoGrade Result (Master-Detail).
    - Professor → Marking Rubric (Lookup).
- **Junction Objects**: Link Rubric to multiple Submissions.
- **External Objects**: Reference ML system results via Salesforce Connect.

Phase 4: Process Automation (Admin)

- **Validation Rules**: Ensure students can't submit empty answers.
- **Workflow Rules**: Notify professor when grading is complete.
- **Process Builder**: Update status field (Submitted → Graded → Published).
- **Approval Process**: Professor approval required for low-confidence grades.
- **Flow Builder**:
    - Screen Flow for professors to input rubrics.
    - Record-triggered Flow to call ML grading API.
    - Auto-launched Flow to update grading result.
- **Email Alerts**: Grade published → notify student.
- **Tasks & Notifications**: Remind professors about pending evaluations.

Phase 5: Apex Programming (Developer)

- **Classes & Objects**: `GradingService.cls`, `RubricParser.cls`.
- **Apex Triggers**: On *Answer Submission* insert → send to ML API.

- **SOQL & SOSL**: Fetch rubric, past grades for training.
- **Collections**: Use Maps for rubric → score mapping.
- **Asynchronous Processing**:
    - *Future Method*: Call external ML API asynchronously.
    - *Queueable Apex*: Handle batch grading.
    - *Scheduled Apex*: Auto-grade overnight.
    - *Batch Apex*: Process large exam sets.
- **Exception Handling**: Catch ML API failures → fallback to manual grading.
- **Test Classes**: Mock ML API response for unit testing.

Phase 6: User Interface Development

- **Lightning App Builder**: Build “Grading Console” app.
- **Record Pages**: Student view vs Professor view.
- **Tabs**: Submissions, Rubrics, Results.
- **Home Page Layout**: Pending grading stats.
- **Utility Bar**: Quick grading shortcut.
- **LWC (Lightning Web Components)**:
    - LWC for answer submission (student).
    - LWC for professor rubric input.
    - LWC to display AI-generated score + rationale.
- **Apex with LWC**: Call ML API.
- **Wire Adapters / Imperative Apex Calls**: Fetch results dynamically.
- **Navigation Service**: Easy navigation between submissions.

Phase 7: Integration & External Access

- **Named Credentials**: Store ML API keys securely.
- **External Services**: Define schema for ML service.
- **Web Services**: REST callouts to grading engine.
- **Platform Events**: Trigger grading workflows asynchronously.
- **Change Data Capture**: Sync updated grading results.
- **Salesforce Connect**: Expose ML service logs as external objects.
- **OAuth & Authentication**: Secure professor login.
- **API Limits**: Batch grading to avoid exceeding limits.
- **Remote Site Settings**: Allow ML API calls.

Phase 8: Data Management & Deployment

- **Data Import Wizard / Data Loader**: Import student lists & rubrics.
- **Duplicate Rules**: Prevent duplicate submissions.
- **Data Export & Backup**: Backup results for compliance.
- **Change Sets**: Deploy objects, fields, and flows.
- **Packages**: Create managed package for reuse by other institutions.
- **VS Code & SFDX**: Local development for LWC + Apex.

Phase 9: Reporting, Dashboards & Security Review

- **Reports**:
    - Average scores by class.
    - Low-confidence grading percentage.

    - Student performance trends.
- **Report Types**: Custom report types linking Submissions & Results.
- **Dashboards**: Professor dashboard with grading status.
- **Dynamic Dashboards**: Personalized student dashboards.
- **Sharing Settings / Field Level Security**: Students only see their own grades.
- **Session Settings / Login IP Ranges**: Restrict professor logins to campus IPs.
- **Audit Trail**: Track grading overrides.

Phase 10: Final Presentation & Demo Day

- **Pitch Presentation**: Problem statement → Salesforce-powered solution → AI integration.
- **Demo Walkthrough**:
    1. Student submits an answer.
    2. ML auto-grades via Salesforce callout.
    3. Professor reviews result.
    4. Final grade published.
- **Feedback Collection**: Gather inputs from professors.
- **Handoff Documentation**: Admin/developer guide, API docs.
- **LinkedIn/Portfolio Showcase**: Highlight project as “Automated Grading System on Salesforce with AI Integration.”