Phase 8: Data Management & Deployment

Effective data management and deployment are crucial in Salesforce to ensure that the application runs smoothly, data integrity is maintained, and updates are efficiently moved between environments. For the **Student Accommodation Finder** project, the following tools and methods were utilized:

1. Data Import Wizard

- **Purpose:** To import data into Salesforce for standard and custom objects without writing code.
- **Objects Used:** Student, Landlord, Accommodation, Booking, Payment.
- **Steps:**
    1. Navigate to **Setup → Data → Data Import Wizard**.
    2. Select the object to import (e.g., Student).
    3. Upload the CSV file containing records.
    4. Map the CSV columns to Salesforce fields.
    5. Start the import and monitor progress.
- **Outcome:** Successfully imported initial datasets such as students' information, accommodation listings, and landlord details.

2. Data Loader

- **Purpose:** To handle large volumes of data import, update, or export, especially when exceeding the limits of the Data Import Wizard.
- **Steps:**
    1. Install Salesforce Data Loader and log in with Salesforce credentials.
    2. Choose operation: Insert, Update, Upsert, or Export.

3. Select the object (e.g., Booking).
4. Upload the CSV file and map fields.
5. Execute the operation and download success/error logs.

- **Usage:** Imported bulk bookings and payments, and updated accommodation availability statuses efficiently.

3. Duplicate Rules

- **Purpose:** To prevent duplicate records and maintain data integrity.
- **Implementation:**
  - Configured duplicate rules for **Student** (email, phone), **Landlord** (email, phone), and **Accommodation** (name + location).
  - Matching rules were defined to identify potential duplicates during data import or record creation.
- **Outcome:** Ensured that no duplicate student registrations, landlord entries, or accommodation listings exist in the system.

4. Data Export & Backup

- **Purpose:** To secure Salesforce data and ensure recovery in case of data loss.
- **Steps:**
  1. Navigate to **Setup → Data → Data Export**.
  2. Select the objects to export (e.g., all custom objects).
  3. Schedule regular weekly or monthly exports.
  4. Download exported ZIP files containing CSVs for all records.
- **Outcome:** Maintained a reliable backup of all project-related data.

5. Change Sets

- **Purpose:** To deploy metadata changes (custom objects, fields, workflows, flows) between Salesforce orgs.

- **Steps:**
    1. Create an **Outbound Change Set** in the source org (sandbox).
    2. Add components like objects, fields, page layouts, and flows.
    3. Upload the change set to the target org (production).
    4. Validate and deploy the change set in the target org.

- **Outcome:** Smooth deployment of configurations and customizations from development to production.

6. Unmanaged vs Managed Packages

- **Unmanaged Packages:**
    - Used to distribute open-source components.
    - Components can be modified in the target org.

- **Managed Packages:**
    - Used for distributing apps or components with version control.
    - Components are protected and cannot be modified in the target org.

- **Usage in Project:**
    - Unmanaged packages were used to share reusable components like flows and Lightning pages between sandbox orgs.

7. ANT Migration Tool

- **Purpose:** A command-line tool used for deploying and retrieving metadata in bulk.

- **Steps:**
    1. Configure the `build.xml` and `build.properties` files with Salesforce credentials.

2. Use commands such as `retrieve` and `deploy` for metadata operations.

3. Validate deployments before applying to production.

- **Outcome:** Enabled automated deployment of complex metadata components for Student Accommodation Finder.

8. VS Code & SFDX

- **Purpose:** Salesforce Developer Experience (SFDX) in VS Code allows source-driven development and seamless deployment.

- **Steps:**
  1. Install **Salesforce Extension Pack** in VS Code.

  2. Authorize the org using `sfdx auth:web:login`.

  3. Retrieve metadata from the org using `sfdx force:source:retrieve`.

  4. Make changes locally and deploy using `sfdx force:source:deploy`.

  5. Use **Scratch Orgs** for testing new features before deploying to production.

- **Outcome:** Improved development workflow and version control for all project metadata.