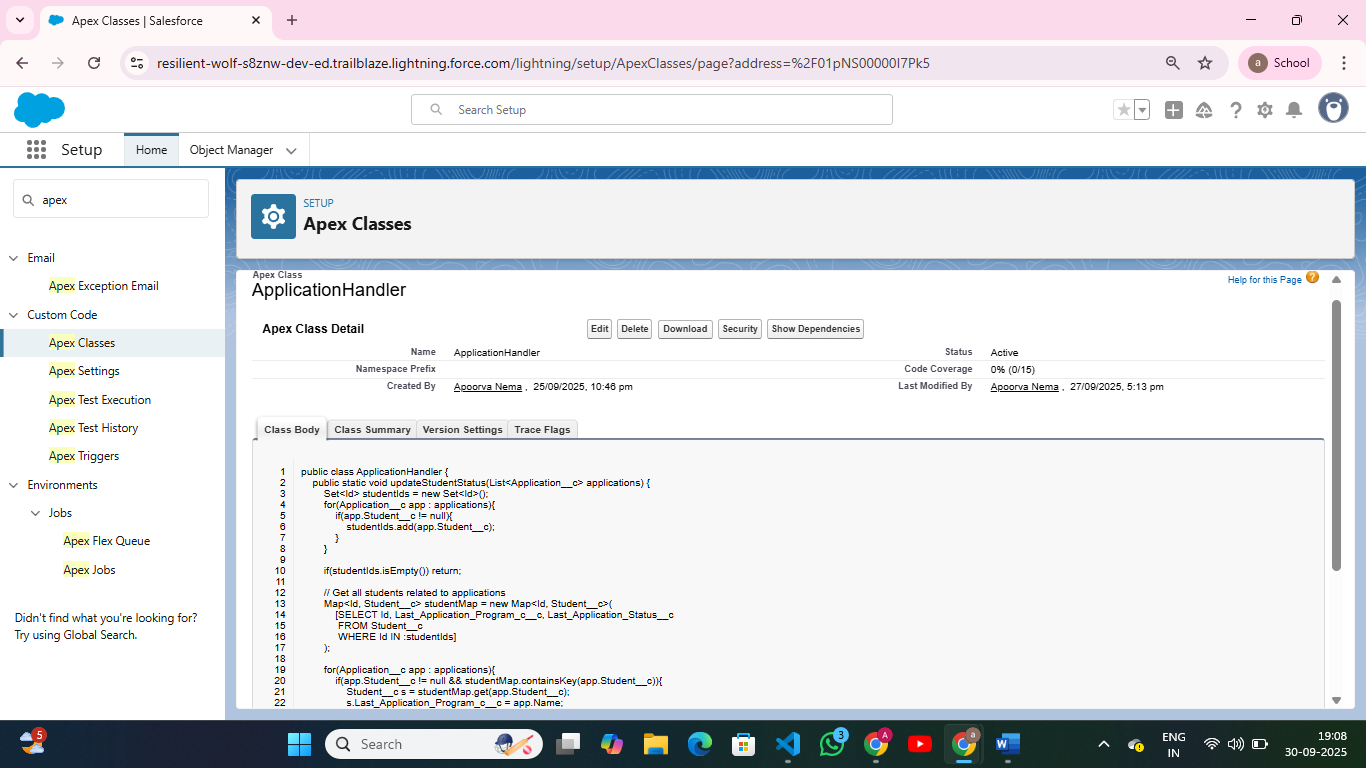
**Phase 5: Apex Programming (Developer)**

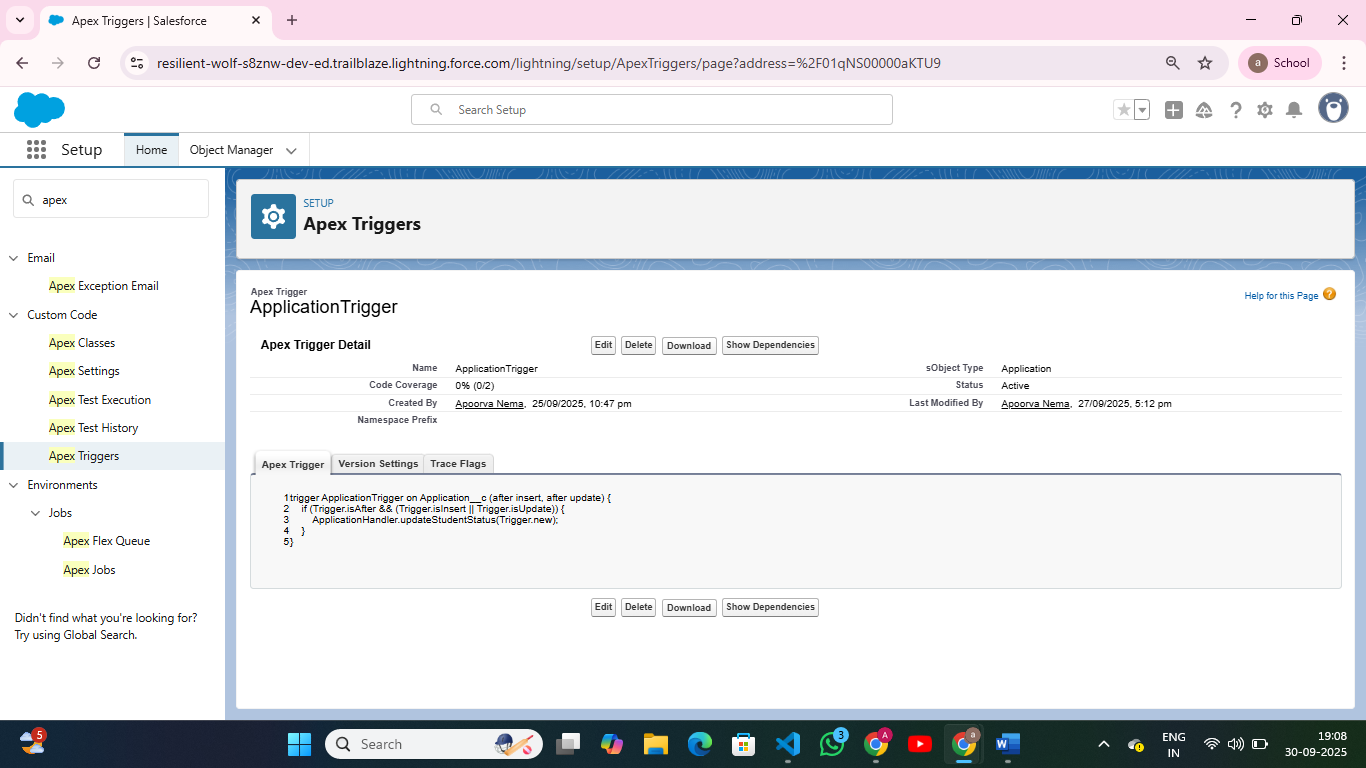
**1. Classes & Objects**

* Apex classes define reusable code blocks.
* **Navigation**: Setup → Apex Classes → New → Write class code → Save.



**Apex Triggers (before/after insert/update/delete)**

* Automate actions on object records before or after database operations.
* **Navigation**: Setup → Apex Triggers → New → Select Object → Define Trigger Event.



**3. Trigger Design Pattern**

**Navigation**: Setup → Apex Triggers → Ensure logic is in separate handler class.

**4. SOQL & SOSL**

* Query Salesforce records efficiently.
* **Navigation**: Execute anonymous window (Setup → Developer Console → Query Editor).
* Example: SELECT Name, Status\_\_c FROM Application\_\_c WHERE Student\_\_c = 'STU001'.

**7. Batch Apex**

* Process large data volumes asynchronously.
* Navigation: Setup → Apex Classes → Implement Database.Batchable interface → Schedule job.

**8. Queueable Apex**

* Lightweight asynchronous processing.
* Navigation: Setup → Apex Classes → Implement Queueable interface → System.enqueueJob(new MyJob());

**9. Scheduled Apex**

* Run Apex code at specific times.
* Navigation: Setup → Apex Classes → Schedule Apex → Select class → Set frequency.

**11. Exception Handling**

* Handle errors using try-catch blocks.
* Example:
* try {
* insert student;
* } catch (DmlException e) {
* System.debug('Error: ' + e.getMessage());
* }

**12. Test Classes**

* Ensure code coverage ≥ 75% for deployment.
* Navigation: Setup → Apex Classes → New Test Class → Annotate with @isTest.
* Example:
* @isTest
* private class TestStudentController { …}

**13. Asynchronous Processing**

* Use Batch Apex, Queueable Apex, Scheduled Apex, and Future Methods for heavy operations.
* Navigation: Setup → Apex Classes → Monitor Jobs (Setup → Apex Jobs).