**Capstone Project Plan**

**Title:** Test Automation for *The Online Book Store*  
**Application Under Test (AUT):** <http://examples.codecharge.com/Store/Default.php>

**1. Team Organization**

Each team consists of **6 members**, organized into Agile roles:

* **Product Owner (PO):** Defines scope, prioritizes backlog, validates deliverables.
* **Scrum Master (SM):** Facilitates Agile ceremonies, removes blockers, ensures smooth progress.
* **Developers (2–3):** Develop Selenium automation scripts, implement framework utilities, maintain reusable functions.
* **Testers (2–3):** Write manual test cases in Excel/JIRA, perform functional & exploratory testing, validate automation results.

**Agile Execution**

* **Sprint Length:** 2 weeks
* **Ceremonies:** Daily Stand-up, Sprint Planning, Sprint Review, Sprint Retrospective
* **Deliverables per Sprint:** Set of automated test scripts, updated manual test cases, execution reports

**2. Test Strategy**

**Framework Approach**

* **Page Object Model (POM):** Separate class files for each page to ensure maintainability.
* **TestNG Integration:** XML-driven test suite execution with groups, priorities, and parallel testing.
* **Cucumber BDD:** Feature files written in Gherkin for readability and stakeholder collaboration.
* **Data Driven Testing:** Using **Apache POI (Excel)** or **Properties file** for test data input.
* **Reusable Utilities:** For WebDriver initialization, waits, logging, reporting, and screenshot capture.

**Other Strategy**

* **Cross-Browser Execution:** Chrome, Firefox (optionally Edge/IE).
* **Coding Standards:** Consistent naming conventions, comments, and exception handling.
* **Screenshots:** Captured on every test pass/fail step and attached in reports.
* **Extent Reports:** Integrated with TestNG for graphical HTML reports.
* **Synchronization:** Use of **Explicit Waits/Fluent Waits** instead of Thread.sleep().

**3. Case Study 1 – Functional Testing (Manual + Automation)**

**Pre-requisites**

* Test environment is up and stable.
* Perform **smoke & exploratory testing** before automation to understand workflows.

**Test Scenarios for Automation**

(*The Online Book Store*)

1. **Search Functionality**
   * Navigate to homepage and enter book name in the search field.
   * Click on **Search** button.
   * Verify that relevant book details are displayed.
   * Verify error message for invalid or empty search.
2. **Advanced Search Functionality**
   * Select a category and enter a keyword.
   * Apply filters: *Price More Than*, *Price Less Than*.
   * Apply sorting: *Order By* (Price/Name).
   * Validate *Records per page* options.
   * Verify filtered results and counts.
3. **Page Validation**
   * Verify the **Page Title** is correct.
   * Validate navigations are correct across search results.
4. **Shopping Cart Functionality**
   * Add a product to the cart.
   * Verify product details (name, price, quantity) in cart.
   * Update quantity and validate total price.
   * Remove items and validate cart is empty.

**4. Technical Implementation**

**Tools & Technologies**

* **Language:** Java
* **Automation Tool:** Selenium WebDriver
* **Frameworks:** TestNG, Cucumber BDD, POM
* **Data Handling:** Apache POI (Excel), Properties file
* **Build Tool:** Maven
* **Reporting:** Extent Reports + TestNG Reports
* **Version Control:** GitHub/GitLab

**Folder Structure**

OnlineBookStore\_Automation/

│── src/main/java

│ ├── pages/ # Page Object classes

│ ├── utils/ # Utilities (Driver, Excel, Config, Waits)

│── src/test/java

│ ├── testcases/ # Test scripts

│ ├── runners/ # TestNG/Cucumber runners

│── testdata/ # Excel sheets, properties

│── reports/ # Extent reports, logs, screenshots

│── pom.xml # Maven dependencies

│── testng.xml # TestNG suite configuration

**5. Deliverables**

* **Manual Test Cases:** Positive, negative, edge cases (Excel/Jira).
* **Automation Scripts:** Implemented with Selenium + TestNG + POM + Cucumber.
* **Execution Reports:** Extent Reports (with screenshots).
* **Cross-Browser Results:** Evidence of execution on Chrome & Firefox.
* **Snapshots:** Screenshots of executed test steps, logs, and reports.
* **Final Submission Package:** Complete project folder (zipped).

**6. Success Criteria**

* All critical functional flows automated successfully.
* Framework is modular, reusable, and scalable.
* Execution is successful on at least two browsers.
* Reports include detailed logs and screenshots.
* Manual and automated results are validated against each other.