EYouth X DEPI Project Proposal

Submitted by: EYouth

Table of content:

I. Group Members & Roles:

1- Each one of the group is responsible ,mainly, for 2 requirements.
(Review, Implement TCs, Manual & Automated Execution)
2-Then the group gathers all test cases and reviews them.
3- Other Parts will be worked on by the whole group.

Team Member	**Role**
Enas AbdElkader	Responsible for Requirement 1 & 10
Hana Yassin	Responsible for Requirement 2 & 12
Nour Ahmed	Responsible for Requirement 3 & 11
Samar farag	Responsible for Requirement 4 & 9
Zeinab Sakran	Responsible for Requirement 5 & 6
Fatma AboElfotoh	Responsible for Requirement 7 & 8

II. Team Leader: Hana Yassin Tawfik

III. Project Description

The **EYouth X DEPI** project aims to develop a robust, automated testing framework for a web-based e-learning platform. The platform includes core functionalities such as user registration, login, course browsing, enrollment, payment processing, content access, and reviews. This project will focus on testing these functionalities to ensure seamless performance, security, and user experience.

Testing Scope:

The testing will cover the following key features and pages:

- **User Management:** Registration, login, and password reset functionalities.
- **Course Browsing & Enrollment:** Search, filtering, and enrollment in both free and paid courses.
- **Cart & Payment Processing:** Adding courses to the cart, checkout process, and payment gateway validation.
- **Content Access:** Viewing course materials, progress tracking, and certification.
- **User Reviews & Ratings:** Submission and visibility of course ratings and reviews.
- **Notifications & Updates:** Course reminders and schedule changes.
- **App Download Feature:** Verification of download buttons and redirections to respective app stores.
- **Cross-Browser & Device Compatibility:** Ensuring functionality across Chrome, Firefox, Safari, Edge, and various devices.

IV. Objectives

Full coverage for website from 2 sides (GUT / API)

- Ensure smooth user registration, login, and authentication.
- Validate course browsing, filtering, and enrollment functionality.
- Test shopping cart features and secure payment processing.
- Verify content access, progress tracking, and certification generation.
- Check user interaction features like course reviews and notifications.
- Confirm cross-browser and device compatibility.
- Identify and document software bugs and issues for resolution.
- Provide performance analysis to optimize system efficiency

V. Tools & Technologies

- **Test Management:** JIRA
- **Implementation:** Google sheet or JIRA
- **Automation Testing:** Selenium WebDriver / Java
- **API Testing:** Postman
- **Bug Tracking & Reporting:** JIRA, TestNG
- **CI/CD Integration:** Jenkins

VI. Milestones & Deadlines → timeline ??

Week 1: Setup and Test Case Development

Tasks:

• Set up testing environment: Install and configure Selenium WebDriver or an equivalent framework for automated testing.

• Project structure setup: Organize the folder structure for test scripts and reports.

• Create sample test cases: Develop initial test cases for basic UI functionality (e.g., navigation, button clicks, form submissions).

• Postman integration: Set up Postman for API testing and write basic test cases to verify API responses.

• Test management integration: Integrate Jira for managing test cases and tracking issues found during testing.

Week 2: Functional Testing and Report Generation

Tasks:

• Develop functional test cases: Write detailed test cases to cover key functional workflows (e.g., user login, form submission, data processing).

• Automated test scripts: Automate the execution of functional tests using Selenium.

• Implement report generation: Integrate tools like TestNG or JUnit to generate detailed test reports with test results.

• API test cases: Expand API testing in Postman by adding more endpoints and verifying data.

• Performance testing setup: Install and configure tools for performance testing (e.g., JMeter, Gatling).

Week 3: Performance and UI Testing

Tasks:

• Develop performance test cases: Write test cases for load and stress testing using tools like JMeter or Gatling.

• Execute performance tests: Run performance tests on the web application to analyze response times, load handling, and scalability.

• UI testing: Enhance UI test cases to cover more complex scenarios (e.g., responsive design, cross-browser compatibility).

• Bug tracking: Log and track issues in Jira discovered during the execution of

performance and UI tests.

Week 4: Final Testing, Optimization, and Deployment

Tasks:

• Optimize test framework: Refactor the test framework to optimize performance and reusability (e.g., using page object models).

• Execute full testing suite: Run the complete suite of tests (UI, functional, API,

performance) on the web application.

• Analyze test results: Review and analyze the results, identifying any remaining issues.

• Generate final reports: Create detailed reports summarizing the results of all tests, including performance metrics.

• Documentation: Write detailed documentation on the automated testing framework, how to use it, and how to add new test cases.

• Deploy the framework: Make the testing framework ready for continuous integration and deployment (CI/CD) pipelines (e.g., integration with Jenkins).

Note:

For all requirements:

1- Review Points (24 Jan - 7 Feb) / 2- Implementation (7 Feb - 7 Mar)

3- Execution “Manual & Automation” (8 Mar - 8 May)

VII. KPIs (Key Performance Indicators)

Full coverage through all possible positive and negative scenarios. Detects all the bugs existing in the website.

1. **Bug Detection Rate:** Number of defects found per test cycle.
2. **Test Coverage Percentage:** Percentage of features covered by test cases.
3. **Response Time for API Requests:** Average response time of API calls.
4. **Successful Payment Transactions:** Ratio of completed payments to total payment attempts.
5. **System Uptime During Testing:** Percentage of time the system remains operational.