Test Plan Document

UI/UX AUTOMATION USING LLM

Project Code:

Internal Advisor: Dr. Fahad Maqbool

External Advisor:

Project Manager: Dr. Muhammad Ilyas

Project Team: Muhammad Dawood (BSCS51F21S089)
Ghulam Rasool (BSCS51F21S080)
Muhammad Rashid (BSCS51F21S084)

Submission Date: **May 21, 2025**

____

Project Manager's Signature

Document Information

Category	Information
Customer	UI/UX Designers, Design Teams, Developers, University of Sargodha
Project	UI/UX AUTOMATION USING LLM
Document	Test Plan
Document Version	1.0
Identifier	PGBH01-2024-TP
Status	Draft
Author(s)	Muhammad Dawood, Muhammad Rashid, Ghulam Rasool
Approver(s)	Dr. Muhammad Ilyas
Issue Date	MAY 21, 2025
Document Location	
Distribution	1. Dr. Fahad Maqbool 2. Dr. Muhammad Ilyas 3. University of Sargodha

Definition of Terms, Acronyms and Abbreviations

Term	Description
LLM	Large Language Model, used for generating designs and contextual suggestions.
GraphRAG	Graph-based Retrieval-Augmented Generation for contextual responses
OWL	Web Ontology Language for semantic consistency.
Neo4j	Graph database used for storing and querying UI/UX design knowledge.
Figma	Collaborative design tool for exporting high-fidelity designs.

Table of Contents

1. Introduction

This Test Plan outlines the strategy to validate the **UI/UX Automation Using LLM** system, which automates design workflows using AI and knowledge graphs. The system generates wireframes from natural language, validates designs against UI/UX principles, and exports them to Figma. Testing ensures these features meet functional, performance, and security requirements for designers, developers, and stakeholders.

1.1 Purpose of Document

- This Test Plan outlines the strategy, scope, and execution of testing activities for the **UI/UX Automation Using LLM** system. It ensures that all functional and non-functional requirements are validated against the SRS and SDS.

1.2 Project Overview

- The system automates UI/UX workflows using LLMs, Neo4j knowledge graphs, and OWL ontologies. Key features include wireframe generation, design validation, and Figma export. Testing ensures these features meet performance, security, and usability standards.

2. Scope of Testing

In-Scope:

- Functional Testing: Wireframe generation, design validation, Figma export, knowledge graph management.
- Non-functional Testing: Performance (5-second response time), security (AES-256 encryption), scalability (1,000 concurrent users).
- Integration Testing: Interaction between LLM, Neo4j, and Figma API.

Out-of-Scope:

- Domain-specific UI/UX requirements outside the ontology.
- Prototypes with interactive elements.

3. Test Plan Strategy

The testing strategy ensures the **UI/UX Automation Using LLM** system meets all functional and non-functional requirements through three phases:

3.1 Unit Testing

Participants: Developers, Test Team

Methodology:

- Test individual components (e.g. UIComponent, InteractiveComponent, LayoutComponent classes).

- Validate Cypher query execution in Neo4j and OWL ontology reasoning.

3.2 Integration Testing

Participants: Test Team, Developers

- Methodology: Validate interaction between subsystems (e.g., Wireframe Generation ↔ Validation ↔ Figma Export).
- Test REST API communication between LLM and GraphRAG.

3.3 System Testing

Participants: Test Team, Stakeholders

- Methodology: End-to-end workflows (e.g., natural language input → wireframe → Figma export).
- Validate compliance with UI/UX principles (Visibility, Accessibility, Consistency).

4. Test Environment

Hardware:

- Standard PC/laptop (8GB RAM, 256GB storage).

Software:

- Neo4j 5.0+, Protégé, OpenAI API, Figma API.

Network:

- Stable internet for API calls and cloud infrastructure (Google Cloud Platform).

5. Schedule

Testing Activities	Begin	End
Unit Testing	01-Mar-2025	15-Mar-2025
Integration Testing	16-Mar-2025	30-Mar-2025
System Testing	01-Apr-2025	15-Apr-2025
Final Review	16-Apr-2025	20-Apr-2025

6. Control Activities

- Weekly review meetings with the Project Manager and Test Team.
- Bug-tracking using JIRA.

7. Functions to be Tested

1. **Wireframe Generation** (SRS 4.1)

2. **Design Validation** (SRS 4.2)
3. **Export to Figma** (SRS 4.4)
4. **Knowledge Graph Management** (SRS 4.5)

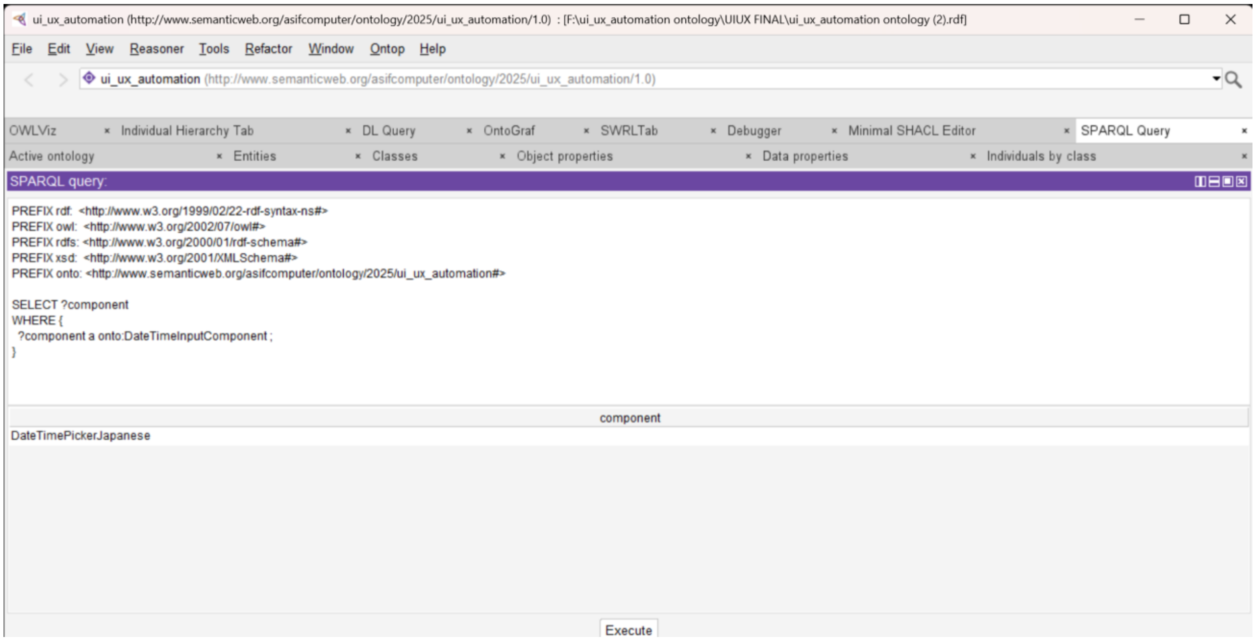
8. Functions not to be Tested

- **Interactive Prototypes** (Out-of-scope per SRS).
- **Domain-Specific Requirements** (Exceeds ontology scope).

9. Test Case Design and Description

Test Case 1:

```
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl:  <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
PREFIX onto:
<http://www.semanticweb.org/asifcomputer/ontology/2025/ui_ux_automation#>


SELECT ?component
WHERE {
  ?component a onto:DateTimeInputComponent ;
}
  Result: DateTimePickerJapanese
```


ui_ux_automation (http://www.semanticweb.org/asifcomputer/ontology/2025/ui_ux_automation/1.0) : [F:\ui_ux_automation ontology\UIUX FINAL\ui_ux_automation ontology (2).rdf]
File Edit View Reasoner Tools Refactor Window Ontop Help
ui_ux_automation (http://www.semanticweb.org/asifcomputer/ontology/2025/ui_ux_automation/1.0)
OWLViz
Individual Hierarchy Tab
DL Query
OntoGraf
SWRLTab
Debugger
Minimal SHACL Editor
SPARQL Query
Active ontology
Entities
Classes
Object properties
Data properties
Individuals by class
SPARQL query:
Execute

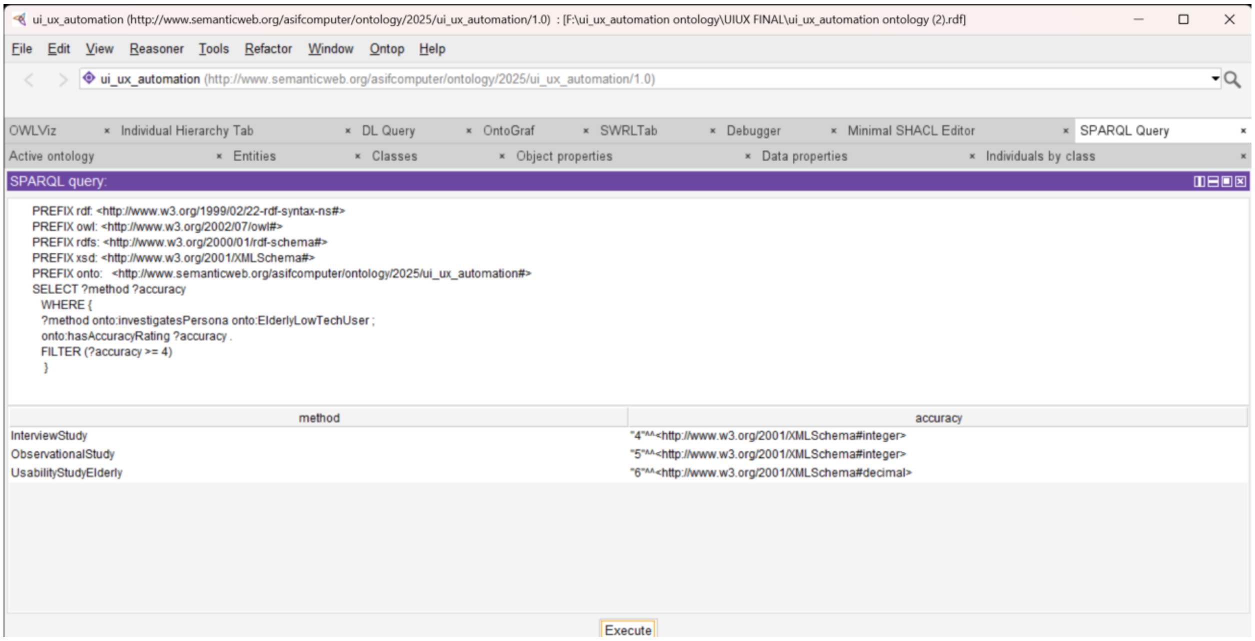
```
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl:  <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
PREFIX onto: <http://www.semanticweb.org/asifcomputer/ontology/2025/ui_ux_automation#>

SELECT ?component
WHERE {
 ?component a onto:DateTimeInputComponent ;
}
```

component
DateTimePickerJapanese

Test Case 2:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX onto:
<http://www.semanticweb.org/asifcomputer/ontology/2025/ui_ux_automation#>
SELECT ?method ?accuracy
  WHERE {
  ?method onto:investigatesPersona onto:ElderlyLowTechUser ;
  onto:hasAccuracyRating ?accuracy .
  FILTER (?accuracy >= 4)
   }
```

Result:


ui_ux_automation (http://www.semanticweb.org/asifcomputer/ontology/2025/ui_ux_automation/1.0) : [F:\ui_ux_automation ontology\UIUX FINAL\ui_ux_automation ontology (2).rdf]
File Edit View Reasoner Tools Refactor Window Ontop Help
ui_ux_automation (http://www.semanticweb.org/asifcomputer/ontology/2025/ui_ux_automation/1.0)
OWLViz × Individual Hierarchy Tab × DL Query × OntoGraf × SWRLTab × Debugger × Minimal SHACL Editor × SPARQL Query ×
Active ontology × Entities × Classes × Object properties × Data properties × Individuals by class ×
SPARQL query:
Execute

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX onto:   <http://www.semanticweb.org/asifcomputer/ontology/2025/ui_ux_automation#>
SELECT ?method ?accuracy
  WHERE {
  ?method onto:investigatesPersona onto:ElderlyLowTechUser ;
  onto:hasAccuracyRating ?accuracy .
  FILTER (?accuracy >= 4)
   }
```

method	accuracy
InterviewStudy	"4"^^<http://www.w3.org/2001/XMLSchema#integer>
ObservationalStudy	"5"^^<http://www.w3.org/2001/XMLSchema#integer>
UsabilityStudyElderly	"6"^^<http://www.w3.org/2001/XMLSchema#decimal>

10. Traceability Matrix

1. **Design Principles** ↔ **Contexts** | Object Property: appliesToContext | Example: ConsistencyPrinciple1 → CrossPlatformEnterpriseContext
2. **UI Components** ↔ **Feedback** | Object Property: hasFeedback | Example: ButtonComponent1 → TactileFeedback
3. **Metrics** ↔ **Components/Personas** | Object Property: evaluatedBy | Example: TimeOnTaskMetric2 → DesktopDevice1
4. **Platforms** ↔ **Benchmarks** | Object Property: hasBenchmark | Example: DesktopDevice1 → TimeOnTaskMetric2
5. **Prototypes** ↔ **Fidelity** | Object Property: hasFidelityLevel | Example: PrototypeA → HighFidelity

11. Major Deliverables

- Test Plan Document
- Test Cases and Reports
- Traceability Matrix

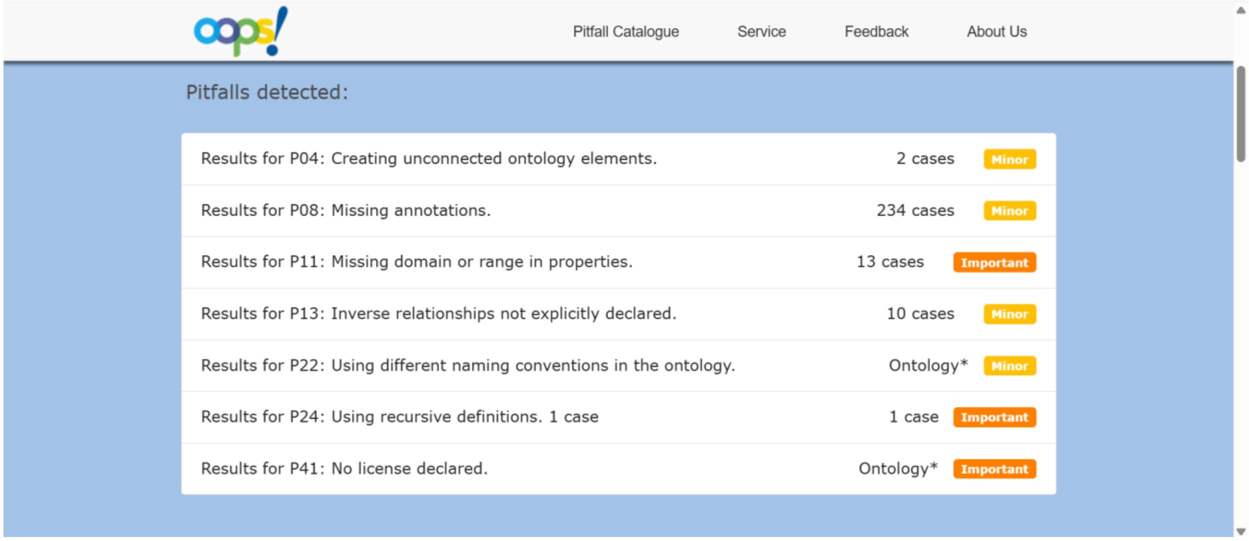
12. Risks and Assumptions

Risks:

- API downtime (OpenAI, Figma).
- Scalability issues with Neo4j.

Assumptions:

- Stable internet and API access.
- Ontology reflects current UI/UX standards.



Pitfalls detected:

Results for P04: Creating unconnected ontology elements.	2 cases	Minor
Results for P08: Missing annotations.	234 cases	Minor
Results for P11: Missing domain or range in properties.	13 cases	Important
Results for P13: Inverse relationships not explicitly declared.	10 cases	Minor
Results for P22: Using different naming conventions in the ontology.	Ontology*	Minor
Results for P24: Using recursive definitions. 1 case	1 case	Important
Results for P41: No license declared.	Ontology*	Important

13. Exit Criteria

- All high-priority test cases passed.
- Performance benchmarks met (5-second response time).
- Security protocols validated (GDPR compliance).

14. References

Ref. No.	Document Title	Date of Release/ Publication	Document Source
RAG2023	GraphRAG Explained: Enhancing RAG with Knowledge Graphs	2023	Medium (https://medium.com/@zilliz_learn/graphrag-explained-enhancing-rag-with-knowledge-graphs-3312065f99e1
OWL200 4	OWL Reference	Feb 10, 2004	W3C (https://www.w3.org/TR/ owl-ref/)
PROTEGE1999	Protégé: A Free, Open- Source Ontology Editor and Framework for Building Intelligent Systems	Nov 11, 1999	wikipedia: http://protege.stanford.edu

Ref. No.	Document Title	Date of Release/ Publication	Document Source
AIUX20 2 3	Artificial Intelligence (AI) for User Experience (UX) Design	Aug 2023	Information Technology and People Journal
UIMOBI LE2022	UI Design Patterns and Ontology Models for Adaptive Mobile Applications	2022	Personal and Ubiquitous Computing Journal

15. Appendices

Appendix:A: Glossary
Appendix B: Workflow Diagrams (from SRS)