Table of Contents

1. Project Overview

This AI-powered oncology platform analyzes integrated genomic, proteomic, and metabolomic data to predict disease outcomes and generate personalized treatment recommendations. The system utilizes Microsoft Fabric for multimodal data orchestration, Azure AI Search for biomedical evidence retrieval, and Azure OpenAI for clinical insights generation. Specifically designed for breast cancer research, it identifies pathogenic mutation patterns, detects clinically significant genomic variants, and synthesizes comprehensive reports highlighting therapeutic implications derived from multi-omics analysis.

The project includes: -

1. **Data Collection Model -** breast cancer recorder for collecting patients data for insight analysis and prevention
2. **Ms Fabric Model** - Data Ingesting, Processing, Training, Analysis : Insight Analysis, Pipelining etc
3. **RAG Interface Model** - querying insights from the Model

Data Collection Model

The Breast Cancer Patient Data Management System is a comprehensive web application designed to streamline the collection, management, and analysis of breast cancer patient information. This React-based solution enables medical professionals to record critical patient data including demographic information, cancer staging, and biometric measurements while ensuring data integrity through robust validation mechanisms.

The system addresses several critical needs in oncology care:

- Standardized data collection for breast cancer patients

- Immediate validation of clinical data entry

- Secure storage and export of patient records

- Foundation for epidemiological research

- Integration with downstream analytics pipelines

Built with modern web technologies, the application features a responsive design that works across devices, from desktop computers in hospital settings to tablets used during patient rounds. The collected data exports in CSV format for compatibility with statistical analysis tools and hospital information systems, while the architecture supports potential expansion to include API integrations with electronic health record (EHR) systems.

2. System Architecture

The platform's architecture (Figure 1) integrates the following components:

Data Orchestration:

Microsoft Fabric Lakehouse: Harmonizes genomic data, clinical records, and research repositories.

Medallion Pipeline: Ingests raw data (bronze), cleans/validates it (silver), and enriches it with analytical features (gold).

AI/ML Components:

Biospecimen Classifier: ML model for genomic data classification.

RAG System: Retrieval-Augmented Generation for dynamic report creation using Azure OpenAI.

Frontend/Backend:

React Frontend: Collects patient data (age, stage, location) with validation and CSV export.

FastAPI Backend: Hosts RAG workflows and integrates with Azure services.

The application follows a client-server architecture with the following components:

Frontend Layer (RAG interface, Data Collection Model):

- React.js for dynamic user interface components

- CSS Grid/Flexbox for responsive layouts

- Formik-like validation patterns for data integrity

- CSV export via react-csv library

Data Layer:

- In-memory state management using React hooks

- UUID generation for record identification

- Temporal data storage until CSV export

- Client-side data validation pipeline

Architectural Patterns:

- Component-based design for UI elements

- Unidirectional data flow

- Stateless functional components

- Context API for potential state sharing

The architecture diagram illustrates:

[User Interface] → [Validation Layer] → [State Management] → [CSV Generator]

Key design decisions:

1. Client-Side Processing: All operations occur in the browser for privacy and reduced server requirements

2. Progressive Enhancement: Core functionality works without JavaScript

3. Modular Design: Components can be reused or replaced independently

4. Validation-First Approach: Data integrity checks precede storage

The lightweight architecture enables deployment in resource-constrained environments while maintaining capability for expansion through:

- Future API integrations

- Cloud storage options

- Authentication layers

- Multi-user collaboration features

3. Technical Implementation

Core Technologies

Python 3.8+

MS Fabric:-

1. **Azure OpenAI:**

Set up an OpenAI resource and deploy a GPT-4 model.

2. **PowerBI:**

For visualization, Reporting and Insight Analysis

React Framework:

- Functional components with hooks

- useState for form state management

- useEffect for side effects

- Custom hooks for validation logic

Dependencies:

- react-csv: 2.0.3+ for CSV generation

- uuid: 8.3.2+ for unique identifiers

- validator: 13.7.0+ for email validation

Implementation Details

Form Handling:

- Controlled components for all inputs

- Dynamic error messaging

- Conditional rendering of validation states

- Progressive enhancement for accessibility

Validation System:

```
const validate = () => {

  const newErrors = {};
```

```
  // Age validation
  if (!formData.age || formData.age < 18 || formData.age > 120) {
    newErrors.age = 'Age must be 18-120';
  }
  // Email validation regex
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
  if (!emailRegex.test(formData.email)) {
    newErrors.email = 'Invalid email format';
  }
  // Weight validation
  if (formData.weight <= 0) {
    newErrors.weight = 'Weight must be positive';
  }
  setErrors(newErrors);
  return Object.keys(newErrors).length === 0;
};
```

CSV Generation:

- Header row with internationalized labels

- ISO date formatting for timestamps

- Numeric precision control

- UTF-8 encoding support

Performance Considerations:

- Memoization of computed values

- Virtualized lists for large datasets

- Chunked processing for CSV generation

- Lazy loading of non-critical components

The implementation prioritizes:

1. Maintainability: Clear component structure

2. Testability: Decoupled business logic

3. Extensibility: Modular architecture

4. Performance: Optimized rendering

4. Data Management

- **Data collection**
- **Data ingesting**

- **Data wrangling**
- **Data processing**
- **Data Analysis**
- **ML -** Classification, Training, Fine-tuning etc.
- **Visualization**

Data Model - (Data Collection and site adherance)

The system manages patient records with the following schema:

Field	Type	Validation Rules	Example
id	UUID	Auto-generated	"a1b2c3d4..."
location	String	Required, max 100 chars	"Nairobi"
age	Integer	18-120	45
stage	String	Enum: Stage 0-IV	"Stage II"
weight	Float	>0, precision 0.1	68.5
email	String	RFC 5322 compliant	"user@domain.com"
timestamp	ISO8601	Auto-generated	"2025-04-03T..."

Data Flow

1. **Collection:** Through validated form inputs

2. **Storage:** In React state memory

3. **Export:** As CSV with headers

4. **Persistence:** Via browser download

Data Quality Measures

- Client-side validation prevents invalid submissions

- Type checking for numeric fields

- Format enforcement for emails

- Required field marking

- Cross-field validation potential

Security Considerations

- No server transmission (all client-side)

- No persistent storage in browser

- Clear data separation

- Optional PII encryption extension points

The data management approach balances usability with privacy protection, suitable for preliminary clinical data collection before transfer to secured hospital systems.

5. User Interface Design and Visual Design Principles

Color Scheme:

- Primary: Teal (008080) for professionalism

- Secondary: Lavender (6a5acd) for calmness

- **Accent:** Light pink (ffb6c1) for awareness

Typography:

- System font stack for performance

- Responsive font scaling

- Accessible contrast ratios

Layout Components

1. Header Section:

- Application title

- Brief description

- Navigation placeholder

2. Form Section:

- Logical field grouping

- Clear labeling

- Inline validation feedback

- Submission controls

3. Data Display:

- Responsive table

- Sortable columns

- Pagination controls

- Summary statistics

4. Export Panel:

- CSV download button

- Data preview

- Record count

Responsive Behavior

- Single column on mobile

- Two-column layout on tablet

- Side-by-side form/data on desktop

- Adaptive input sizes

- Touch-friendly targets

The interface follows medical UI best practices with high contrast, clear information hierarchy, and minimal distractions for clinical environments.

6. Validation and Security

Comprehensive Validation System

Field-Level Validation:

- **Age range:** 18-120 years

- **Weight:** Positive numbers only

- **Email:** Format verification

- Required fields: Location, stage

Validation Feedback:

- Real-time error detection

- Visual error indicators

- Descriptive error messages

- Form submission prevention

Security Measures

Data Protection:

- Client-side only processing

- No backend persistence

- Optional PII encryption

- Clear data ownership

Application Security:

- React DOM sanitization

- CSRF protection

- XSS prevention

- Dependency auditing

Privacy Features:

- No analytics tracking

- No third-party scripts

- Data never leaves browser

- Explicit export action

The validation system ensures only clinically plausible data enters the system, while the security model protects patient confidentiality through client-side isolation and transparent data handling.

7. Deployment and Scalability

Deployment Options

1. Static Hosting:

- Vercel

2. Containerized:

- Docker image

- Kubernetes deployment

- Serverless functions

3. On-Premises:

- Internal web server

- Electron desktop wrapper

- Local network hosting

Scalability Pathways

- API integration for centralized storage

- Offline capability with service workers

- Database backend adoption

- User authentication layer

- Role-based access control

The lightweight architecture supports everything from single-file deployment to enterprise-scale integration while maintaining core functionality across all scenarios.

8. Usage Scenarios and **Clinical Workflows**

1. New Patient Intake:

- Registrar completes form during consultation
- Data validates automatically
- Record added to local dataset

2. Data Export:

- Administrator exports CSV weekly
- File transfers to hospital EHR
- Data integrates with analytics

3. Research Analysis:

- Epidemiologist imports CSV to SPSS/R
- Demographic trends identified
- Correlation studies performed

User Roles

1. Clinical Staff: Data entry

2. Administrators: Data export

3. Researchers: Data analysis

4. IT Staff: System maintenance

The application supports the complete data lifecycle from collection through analysis while fitting seamlessly into clinical workflows.

9. Maintenance and Future Enhancements

- Enhanced export formats (JSON, XML)

- EHR system integration

- Predictive modeling

- Genomic data linkage

- Maintenance Plan

- Regular dependency updates

- Browser compatibility testing

- Accessibility audits

- Performance monitoring

The project maintains flexibility for future healthcare technology advancements while delivering immediate value in current form.

10. Conclusion

This Breast Cancer Patient Data Management System provides a robust, privacy-conscious solution for recording and managing oncology patient information. Its thoughtful design combines clinical practicality with technical excellence, offering healthcare providers an immediate tool for standardized data collection while establishing foundations for advanced analytics and research integration in cancer care and epidemiology studies.