Autism Spectrum Disorder Prediction and Support System

Presented to

Dr. Hoang Nguyen

MSDS 545: Intro to Computer Software Engineering

Prepared by

Julian Broughton

Gabrielle Dawkins

Micaiah Mcdonald

Destiny Pounds

Jean-Hus Theodore

5 August 2024

**Problem Statement**

As healthcare providers, we see many young patients exhibiting symptoms that may suggest autism spectrum disorder (ASD), such as delayed speech, inattentiveness, and behavioral issues. Diagnosing ASD can be complex and time-consuming, involving multiple questionnaires and extensive examination. We seek a more efficient and accurate method to support the diagnostic process. By entering patient symptoms into a software system, we hope to streamline the diagnosis of ASD, ensuring that we can provide clear, understandable results to parents. Our goal is to have a tool that not only aids in diagnosis but also supports continuous monitoring and treatment, helping us communicate effectively with parents and manage each patient’s care comprehensively.

**Goals, Requirements, and Analysis**

*Business Goals:*

Goal 1: Improve ASD Diagnosis Accuracy

* Utilize standardized ASD questionnaires
* Integrate machine learning model

Goal 2: Enhance Efficiency of Diagnosis Process

* Streamline data input and processing
* Automate risk assessment and treatment suggestions

Goal 3: Improve Communication with Parents

* Provide clear, understandable diagnostic results
* Offer accessible treatment suggestions

Goal 4: Support Continuous Monitoring and Treatment

* Maintain comprehensive patient profiles
* Enable ongoing tracking of patient progress

*Enumerated Functional Requirements:*

|  |  |  |
| --- | --- | --- |
| **Requirement ID** | **Priority** | **Description** |
| REQ-1 | High | The system shall provide a secure web interface for inputting screening responses. |
| REQ-2 | High | The system shall preprocess collected data to handle missing values and outliers. |
| REQ-3 | High | The system shall integrate a machine learning model to predict ASD risk. |
| REQ-4 | High | The system shall generate a detailed risk assessment score for physicians. |
| REQ-5 | Medium | The system shall generate a basic risk assessment score for parents. |
| REQ-6 | Medium | The system shall provide treatment suggestions based on the risk assessment. |
| REQ-7 | High | The system shall allow physicians to modify treatment suggestions. |
| REQ-8 | High | The system shall provide a user-friendly interface for data input and result interpretation. |
| REQ-9 | High | The system shall integrate with existing EHR systems for data interoperability. |
| REQ-10 | Medium | The system shall create and store individualized patient profiles. |
| REQ-11 | High | The system shall provide role-based access control to ensure appropriate access levels. |
| REQ-12 | High | The system shall allow physicians to access patient medical records. |

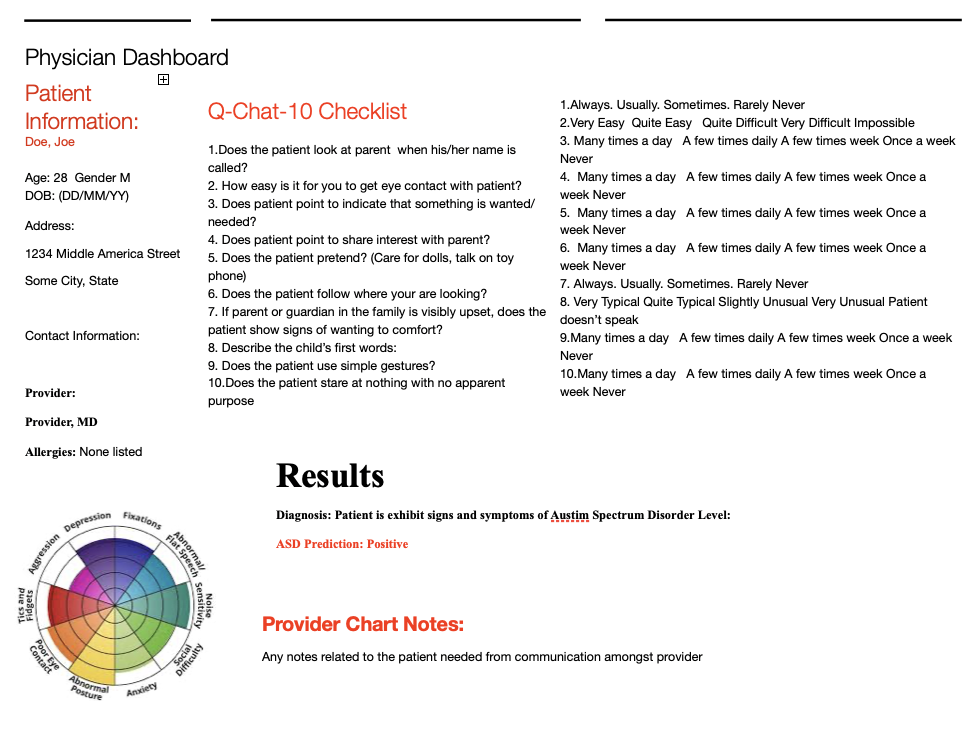
*Enumerated Nonfunctional Requirements:*

|  |  |  |
| --- | --- | --- |
| **Requirement ID** | **Priority** | **Description** |
| NFR-1 | High | The system shall comply with HIPAA and other relevant healthcare data privacy regulations. |
| NFR-2 | High | The system shall ensure data encryption during transmission and storage. |
| NFR-3 | Medium | The system shall have an uptime of 99.9% |
| NFR-4 | High | The system shall include backup and recovery mechanisms to prevent data loss. |
| NFR-5 | High | The system shall respond to user inputs within 2 seconds. |
| NFR-6 | Medium | The system shall process and display risk assessment scores within 5 seconds after data submission. |
| NFR-7 | High | The system shall provide an intuitive interface accessible to users with basic computer skills. |
| NFR-8 | Medium | The system shall offer online help and documentation to assist users. |
| NFR-9 | Medium | The system shall log all user activities for auditing purposes. |

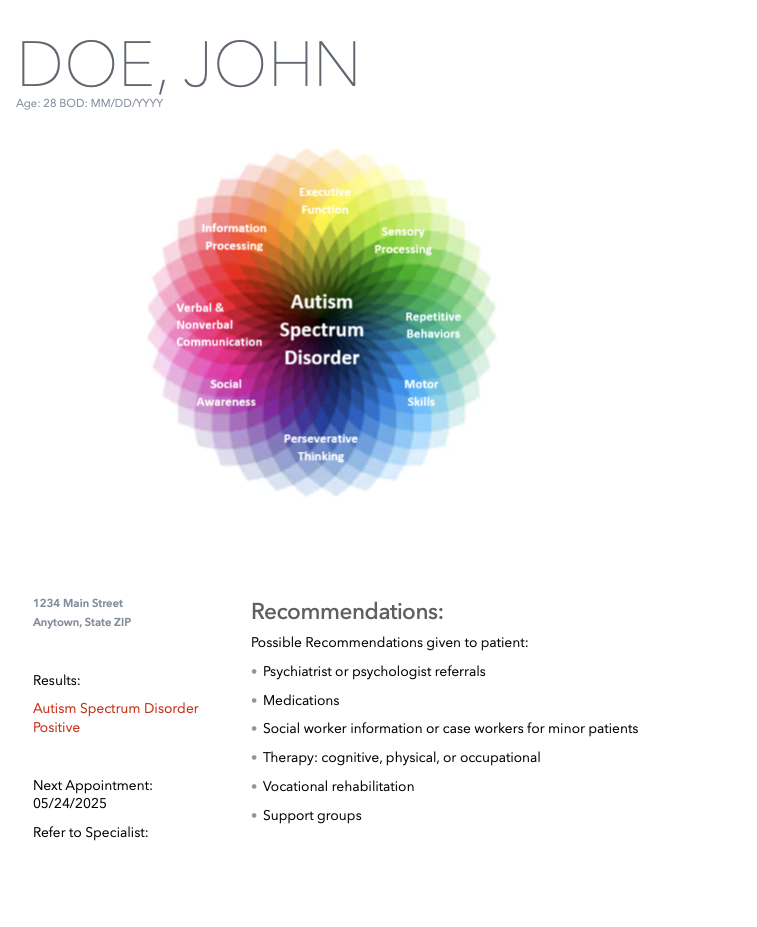
*User Interface Requirements:*

|  |  |  |
| --- | --- | --- |
| **Requirement ID** | **Priority** | **Description** |
| UI-1 | High | The user interface shall provide a clear and intuitive form for inputting screening responses. |
| UI-2 | High | The user interface shall display detailed risk assessment scores for physicians. |
| UI-3 | Medium | The user interface shall display basic risk assessment scores for parents. |
| UI-4 | Medium | The user interface shall provide a section for treatment suggestions. |
| UI-5 | Medium | The user interface shall allow physicians to modify treatment suggestions. |
| UI-6 | High | The user interface shall provide easy navigation to access patient medical records. |
| UI-7 | Medium | The user interface shall offer accessible help and documentation sections. |

*SKETCH OF PHYSICIAN DASHBOARD*



*SKETCH OF PARENT DASHBOARD*



**Use cases**

*Use Case 1: Input Screening Responses*

* Actors: Physicians, Nurses
* Description: Healthcare providers (Physicians or Nurses) input Q-CHAT-10 responses into the system through a secure web interface.
* Associated Requirements: REQ-1, UI-1

*Use Case 2: Preprocess Data*

* Actors: System
* Description: The system preprocesses the collected responses, handling missing values and outliers, and normalizing the data for compatibility with the machine learning model.
* Associated Requirements: REQ-2

*Use Case 3: Predict ASD Risk Using the Machine Learning Model*

* Actors: System
* Description: The system uses a machine learning model to predict the risk of ASD based on the preprocessed data.
* Associated Requirements: REQ-3

*Use Case 4: Generate Detailed Risk Assessment for Physicians*

* Actors: Physicians
* Description: The system generates a detailed risk assessment score and provides it to physicians for review.
* Associated Requirements: REQ-4, UI-2

*Use Case 5: Generate Basic Risk Assessment for Parents*

* Actors: Parents
* Description: The system generates a basic risk assessment score and provides it to parents for review.
* Associated Requirements: REQ-5, UI-3

*Use Case 6: Provide Treatment Suggestions*

* Actors: Physicians, Parents
* Description: The system provides treatment suggestions based on the risk assessment.
* Associated Requirements: REQ-6, UI-4

*Use Case 7: Modify Treatment Suggestions*

* Actors: Physicians
* Description: Physicians can modify the treatment suggestions provided by the system.
* Associated Requirements: REQ-7, UI-5

*Use Case 8: Access Patient Medical Records*

* Actors: Physicians
* Description: Physicians can access patient medical records through the system.
* Associated Requirements: REQ-12, UI-6

*Use Case 9: Manage Patient Profiles*

* Actors: Physicians, System
* Description: The system creates and stores individualized patient profiles containing screening responses, risk assessment scores, and treatment suggestions. Physicians can update and manage these profiles.
* Associated Requirements: REQ-10, REQ-8, REQ-9

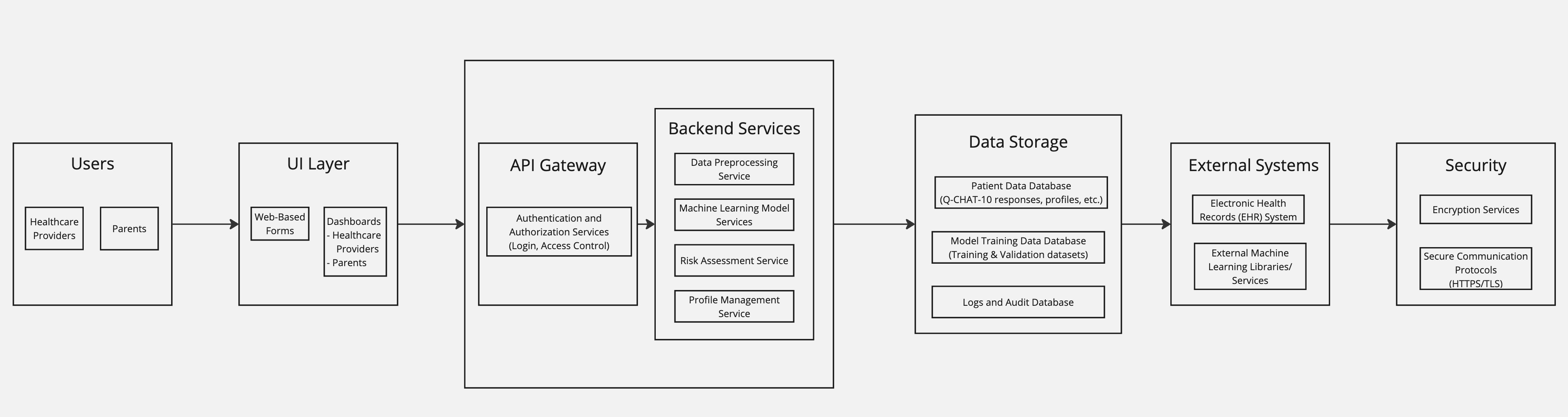
*Use Case 10: Manage User Accounts*

* Actors: System Administrators
* Description: System administrators manage user accounts, providing role-based access control to ensure appropriate access levels.
* Associated Requirements: REQ-11

*Use Case 11: Integrate with EHR Systems*

* Actors: System
* Description: The system integrates with existing EHR systems to import/export patient data.
* Associated Requirements: REQ-9

**System Architectures**



*Major Components:*

* *User Interface (UI)*:
* Web-Based Forms for Data Input: Used by healthcare providers (physicians and nurses) to input screening responses.
* Dashboard for Physicians: Allows physicians to view detailed risk assessments, access medical records, and modify treatment suggestions.
* Dashboard for Nurses: Allows nurses to input screening responses and view relevant data.
* Dashboard for Parents: Provides parents with basic risk assessments and treatment suggestions.
* *Backend Services:*
* API Gateway: Manages API requests and routes them to appropriate services.
* Authentication and Authorization Service: Handles user logins and access control.
* Data Preprocessing Service: Cleans, normalizes, and encodes the data for compatibility with the machine learning model.
* Machine Learning Model Service: Integrates and utilizes the pre-built model to predict ASD risk.
* Risk Assessment Service: Computes the risk assessment score based on model predictions.
* Treatment Suggestion Service: Generates and allows modification of treatment suggestions.
* Profile Management Service: Manages the creation, storage, and retrieval of individualized patient profiles.
* *Data Storage:*
* Patient Data Database: Stores screening responses, risk assessments, treatment suggestions, and other patient-related information.
* Model Data Database: Contains data related to the pre-built machine learning model.
* Logs and Audit Database: Stores logs for system activities and user actions for auditing purposes.
* *External Systems:*
* Electronic Health Records (EHR) System: Integrates with EHR systems for importing/exporting patient data.
* External Machine Learning Libraries/Services: Utilizes external libraries and services for advanced machine learning functionalities.
* *Security:*
* Encryption Services: Ensures data encryption during storage and transmission.
* Secure Communication Protocols (HTTPS, TLS): Provides secure communication channels between users and the system.

**Testing**

The Autism Spectrum Disorder (ASD) Prediction System aims to provide accurate and efficient diagnosis support through the integration of Q-CHAT-10 responses and a machine learning model. Given the sensitivity and importance of this system, thorough testing is required to ensure reliability, accuracy, and usability. Our testing strategy includes unit tests, integration tests, and user acceptance tests to validate each component of the system.

**Testing Requirements:**

1. **Accuracy Testing**: Verify the accuracy of the machine learning model in predicting ASD risk.
2. **Functionality Testing**: Ensure all features work as intended.
3. **Usability Testing**: Validate that the user interface is intuitive and accessible to healthcare providers and parents.
4. **Performance Testing**: Confirm the system responds to user inputs within acceptable time frames and handles multiple concurrent users without performance degradation.
5. **Security Testing**: Ensure data encryption during transmission and storage, and compliance with HIPAA and other relevant healthcare data privacy regulations.
6. **Integration Testing**: Validate the system's ability to integrate with Electronic Health Records (EHR) systems.

*Examples of Tests to be conducted:*

**1. Unit Test for Data Preprocessing Service**

*Objective*: Ensure the data preprocessing service correctly handles missing values and outliers and normalizes the data.

*Example Code*:

import unittest

from preprocessing\_service import preprocess\_data

class TestPreprocessData(unittest.TestCase):

def test\_missing\_values(self):

input\_data = {'q1': [1, 2, None], 'q2': [3, None, 4]}

expected\_output = {'q1': [1, 2, 1.5], 'q2': [3, 3.5, 4]} # Assuming mean imputation

self.assertEqual(preprocess\_data(input\_data), expected\_output)

def test\_outliers(self):

input\_data = {'q1': [1, 2, 1000], 'q2': [3, -999, 4]}

expected\_output = {'q1': [1, 2, 2], 'q2': [3, 3, 4]} # Assuming outlier removal

self.assertEqual(preprocess\_data(input\_data), expected\_output)

def test\_normalization(self):

input\_data = {'q1': [1, 2, 3], 'q2': [4, 5, 6]}

expected\_output = {'q1': [0.0, 0.5, 1.0], 'q2': [0.0, 0.5, 1.0]} # Min-max normalization

self.assertEqual(preprocess\_data(input\_data), expected\_output)

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

**2. Unit Test for Risk Assessment Service**

*Objective*: Ensure the risk assessment service correctly generates a risk score using the pre-built machine learning model.

Example Code:

import unittest

from risk\_assessment\_service import generate\_risk\_score

class TestRiskAssessmentService(unittest.TestCase):

def test\_risk\_score(self):

input\_data = {'q1': 1, 'q2': 3, 'q3': 2, 'q4': 1, 'q5': 3, 'q6': 2, 'q7': 1, 'q8': 3, 'q9': 2, 'q10': 1}

expected\_output = 0.85 # Assuming model output

self.assertAlmostEqual(generate\_risk\_score(input\_data), expected\_output, places=2)

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

**3. Integration Test for System with EHR**

*Objective*: Ensure the system correctly integrates with the EHR system for data import/export.

*Example Code*:

import unittest

from ehr\_integration import import\_patient\_data, export\_patient\_data

class TestEHRIntegration(unittest.TestCase):

def test\_import\_patient\_data(self):

ehr\_data = {'patient\_id': 123, 'name': 'John Doe', 'q\_chat\_10': {'q1': 1, 'q2': 2, 'q3': 3, 'q4': 4, 'q5': 5, 'q6': 6, 'q7': 7, 'q8': 8, 'q9': 9, 'q10': 10}}

expected\_output = {'q1': 1, 'q2': 2, 'q3': 3, 'q4': 4, 'q5': 5, 'q6': 6, 'q7': 7, 'q8': 8, 'q9': 9, 'q10': 10}

self.assertEqual(import\_patient\_data(ehr\_data), expected\_output)

def test\_export\_patient\_data(self):

patient\_data = {'patient\_id': 123, 'risk\_score': 0.85, 'treatment\_suggestions': 'Behavioral therapy recommended.'}

expected\_output = {'status': 'success', 'ehr\_id': 456}

self.assertEqual(export\_patient\_data(patient\_data), expected\_output)

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

**4. User Interface Usability Test**

*Objective*: Validate the user interface for intuitiveness and accessibility.

*Method*: Conduct a usability test with a group of healthcare providers and parents, asking them to perform common tasks (e.g., inputting Q-CHAT-10 responses, viewing risk assessments) and gathering feedback on ease of use, clarity of information, and overall experience.

**5. Performance Test**

*Objective*: Ensure the system can handle up to 1,000 concurrent users without performance degradation.

*Method*: Use a performance testing tool to simulate 1,000 concurrent users performing various tasks on the system and measure response times, throughput, and resource utilization.

**6. Security Test**

*Objective*: Validate compliance with HIPAA and ensure data encryption during transmission and storage.

*Method*: Conduct penetration testing to identify potential security vulnerabilities, verify data encryption methods, and ensure access controls are properly implemented.

This comprehensive testing strategy ensures that the ASD Prediction System is robust, accurate, and user-friendly, meeting the needs of healthcare providers and parents while complying with regulatory requirements.

**Project Management Plan:**

*Approaches to Software Engineering*

**1. Agile Methodology:**

* **Iterative Development**: The project will be divided into several sprints, each lasting 2-3 weeks. Each sprint will focus on developing a specific module or component of the system.
* **Weekly Check-ins:** Short weekly meetings will be held to discuss progress, identify blockers, and plan for the upcoming week.
* **Sprint Reviews and Retrospectives**: At the end of each sprint, we will review what was accomplished, gather feedback, and identify areas for improvement.

**2. Requirements Engineering:**

* **Requirement Gathering**: Consult with stakeholders (healthcare providers, parents, system administrators) to gather detailed requirements.
* **Requirement Analysis**: Prioritize requirements based on their importance and feasibility and create detailed user stories and acceptance criteria.
* **Requirement Documentation**: Maintain a living document that captures all requirements, changes, and decisions.

**3. Design and Architecture:**

* **High-Level Architecture**: Define a high-level architecture diagram that outlines the major components and their interactions.
* **Modular Design**: Break down the system into modular components to ensure scalability, maintainability, and ease of integration.
* **Design Reviews**: Conduct design reviews with the team and stakeholders to ensure alignment and address any concerns early.

**4. Development:**

* **Version Control**: Use Git for version control to manage code changes and collaborate effectively.
* **Code Reviews**: Implement regular code reviews to ensure code quality and adherence to coding standards.
* **Automated Testing**: Develop unit tests, integration tests, and system tests to ensure the reliability of the system.

**5. Testing:**

* **Unit Testing**: Test individual components to ensure they function correctly.
* **Integration Testing**: Test interactions between different components to identify and fix integration issues.
* **System Testing**: Conduct end-to-end testing to ensure the entire system works as intended.
* **User Acceptance Testing**: Involve stakeholders in testing the system to validate that it meets their needs and expectations.

**6. Deployment:**

* **Staging Environment**: Deploy the system to a staging environment for final testing before going live.
* **Production Deployment**: Plan and execute the deployment of the system in a live production environment.
* **Monitoring and Maintenance**: Establish monitoring and maintenance routines to ensure the system remains functional and up to date.

*Roadmap with Projected Milestones*

|  |  |  |
| --- | --- | --- |
| **Milestone** | **Description** | **Date** |
| Project Initiation | Define project scope, objectives, and team roles. | May 23, 2024 |
| Requirement Gathering and Analysis | Collect and analyze requirements from stakeholders. | June 27, 2024 |
| System Design | Create high-level design and architecture diagrams. | August 1, 2024 |
| Sprint 1: Data Collection Module | Develop and test the data collection module (Q-CHAT-10 responses). | August 15, 2024 |
| Sprint 2: Preprocessing Service | Develop and test the data preprocessing service. | September 1, 2024 |
| Sprint 3: Machine Learning Model | Integrate and test the pre-built machine learning model. | September 15, 2024 |
| Sprint 4: Risk Assessment Service | Develop and test the risk assessment service. | October 1, 2024 |
| Sprint 5: User Interface Development | Design and develop user interfaces for physicians, nurses, and parents. | October 15, 2024 |
| Integration Testing | Conduct integration testing for all components. | November 1, 2024 |
| System Testing | Perform end-to-end system testing. | November 15, 2024 |
| User Acceptance Testing | Conduct user acceptance testing with stakeholders. | December 1, 2024 |
| Deployment Preparation | Prepare for deployment in the staging environment. | December 15, 2024 |
| Staging Deployment | Deploy the system in the staging environment for final testing. | January 1, 2025 |
| Production Deployment | Deploy the system in the live production environment. | January 15, 2025 |
| Post-Deployment Monitoring | Monitor the system and address any issues. | February 1, 2025 |

**Implementation Results**

As we are entering the implementation phase, this section will be updated upon completion of each milestone. We plan to document all development activities, challenges faced, solutions implemented, and feedback received during testing and deployment phases.

**References**

1. Alves, C.L., Toutain, T.G.L.d.O., de Carvalho Aguiar, P. *et al.* Diagnosis of autism spectrum disorder based on functional brain networks and machine learning. *Sci Rep* **13**, 8072 (2023). <https://doi.org/10.1038/s41598-023-34650-6>
2. Bahathiq, R.A., Banjar, H., Bamaga, A.K. and Jarraya, S.K. Machine learning for autism spectrum disorder diagnosis using structural magnetic resonance imaging: Promising but challenging. Frontiers in Neuroinformatics, 16, 2022.
3. Farooq, M.S., Tehseen, R., Sabir, M. *et al.* Detection of autism spectrum disorder (ASD) in children and adults using machine learning. *Sci Rep* **13**, 9605 (2023). <https://doi.org/10.1038/s41598-023-35910-1>
4. Q-CHAT-10 Questionnaire: [<https://www.autismalert.org/uploads/PDF/SCREENING--AUTISM--QCHAT-10%20Question%20Autism%20Survey%20for%20Toddlers.pdf>]
5. HIPAA Compliance Guidelines: [<https://www.hhs.gov/hipaa/for-professionals/security/laws-regulations/index.html>]
6. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications