**PROJECT TITLE: IRIS TUMOUR DETECTION USING CONVOLUTIONAL NEURAL NETWORKS (CNN)**

**TEAM NUMBER: 02**

**Team Members:**

1. **Manogna Bandlamudi**
2. **Keerthana G.S**
3. **Hari Meghana Kavala**

**1.PROJECT OVERVIEW:**

**WHAT IS THE GOAL?**

The primary goal of this project is to develop a Convolutional Neural Network (CNN) model specifically designed for detecting tumors in iris images, helping to facilitate early diagnosis. The project also aims to create a user-friendly web platform that hosts the model, allowing users to easily upload iris images and receive real-time tumor detection results.

**WHY IS IT IMPORTANT?**

1. **Early Detection**

* Enable faster identification of potential iris tumors
* Reduce time between initial screening and specialist referral

1. **Accessibility**

* Reduce geographical barriers to specialist eye care
* Enable preliminary screening in remote or underserved areas

1. **Efficiency**

* Automate initial screening process
* Decrease waiting times for tumor detection

**WHO’S INVOLVED?**

1. **Intern Co-ordinator/ Leader**: Responsible for overseeing the project timeline, resource allocation, and ensuring the project goals align with clinical and technical standards.
2. **Interns:**  Handle front-end and back-end integration.
3. **End-Users**: Primarily healthcare professionals and possibly patients who use the tool to receive preliminary insights into iris health, with results aiding in professional consultations.

**2. REQUIREMENTS DOCUMENTATION:**

**WHAT DOES IT NEED TO DO?**

The project aims to develop a Convolutional Neural Network (CNN) for tumor detection in iris images and present the results via an interactive web platform. The core features and functions of the system include:

1. **Image Upload:**
   * Users can upload iris images through an easy-to-use interface.
   * The system supports multiple image formats (e.g., JPEG, PNG).
2. **Tumor Detection:**
   * The CNN model analyzes the uploaded iris images to detect the presence of any tumors.
3. **Result Display:**
   * The website provides a clear, user-friendly result page displaying the tumor detection outcome.
4. **Model Performance Feedback:**
   * The system provides feedback to users, such as "tumor detected" or "no tumor detected."
5. **Secure User Access:**
   * Users must have an account to submit images, with options for user authentication (e.g., login/register).

**HOW WELL DOES IT NEED TO WORK?**

To ensure the system’s effectiveness and reliability, the following performance, security, and usability standards must be met:

1. **Performance:**
   * The CNN model should provide accurate and timely results within a reasonable time frame (e.g., within 10 seconds per image).
2. **Security:**
   * The system must ensure the confidentiality and integrity of users.
3. **Usability:**
   * The website should be intuitive and user-friendly, requiring minimal training for healthcare professionals to operate effectively.
4. **Speed**

* Image upload response: < 4 seconds
* Analysis processing time: < 5 seconds
* Page load time: < 3 seconds

**WHY IS IT NEEDED?**

This project supports several business objectives and aligns with the company's goals in the following ways:

1. **Early Detection of Tumors:**
   * By providing an automated tumor detection tool, the project helps healthcare professionals in early diagnosis, leading to better patient outcomes through faster intervention.
2. **Increased Accessibility:**
   * The platform provides a user-friendly tool for non-expert users (e.g., patients or general practitioners) to analyze iris images for early signs of tumors, increasing healthcare accessibility, especially in rural or under-resourced areas.
3. **Enhanced Decision-Making:**
   * It supports medical professionals in decision-making by providing a second opinion tool that aids in detecting tumors they may not notice in initial exams.

**HOW WILL USERS USE IT?**

1. **Login/Register:**
   * **Process:**
     + New users will need to register by providing basic information (username, password) to create an account.
     + Existing users can simply log in using their credentials (username and password).
     + User authentication will ensure secure access to the platform.
2. **Upload Image:**
   * **Process:**
     + Users will click the "Upload Image" button and select the image file (in supported formats like JPEG, PNG) from their device.
     + The platform includes file validation to ensure only image files are uploaded.
3. **Wait for Analysis:**
   * **Process:**
     + This process will take a few seconds.
4. **Receive Result:**
   * **Process:**
     + The system will display the result on the screen, indicating whether a tumor has been detected or not.

**3.PROJECT PLAN:**

**PROJECT TIMELINE:**

1. **Data Collection & Preparation** (1 week): Collect and preprocess iris images for model training.
2. **Model Development & Training** (2 weeks): Design, train, and fine-tune the CNN model for tumor detection.
3. **Web Interface Development** (3 weeks): Create a user interface with login, image upload, and result display pages.
4. **Testing & Optimization** (1 week): Test and refine the app for accuracy, usability, and stability.
5. **Deployment & Maintenance** (1 week): Deploy the app and set up ongoing support.

**PROJECT RESOURCES**

1. **Team:**
   * Interns: Handle front-end and back-end integration.
   * Intern Coordinator: Coordinates the project timeline and ensures deliverables.
2. **Software:**
   * Development Environment: Python for model and backend development.
   * ML Libraries: TensorFlow and Keras for building and training the CNN model.
   * Web Framework: Django for backend services, with HTML/CSS for the UI.
3. **Dataset:**
   * High-Resolution Iris Images: Essential for training the model with accurate data.

**4. ARCHITECTURE AND DESIGN DOCUMENTATION FOR IRIS TUMOR DETECTION WEB APPLICATION**

**HOW IS IT BUILT?**

This iris tumor detection application is structured as a **client-server web application** with distinct components for image processing, data handling, and user interaction:

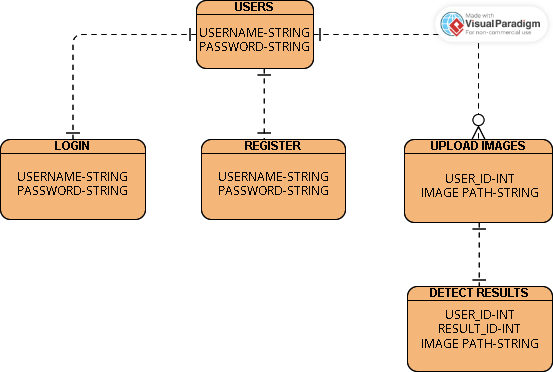
1. **Frontend (Client Side)**:
   * **User Interface**: Created with HTML, CSS, allowing users to register, log in, upload images, and view results.
2. **Backend (Server Side)**:
   * **Web Framework**: Built using Django to handle user requests, manage sessions, process images, and retrieve results.
   * **Machine Learning Module**: A Convolutional Neural Network (CNN) model, implemented in Python with TensorFlow or Keras, to detect tumors from iris images.
3. **Database**:
   * **User Data**: User credentials, login information are stored securely.
   * **Iris Image Data**: High-resolution iris images are processed temporarily**.**

**WHAT DOES IT LOOK LIKE?**

**4.3 How is Data Stored?**

* **Database Management System**: A relational database (such as MySQL or PostgreSQL) stores user information, login credentials, and detection logs.
* **File Storage**: Processed images may be stored temporarily in a secure directory on the server.

**HOW IS DATA STORED?**

**HOW IS DATA STORED:**

**5.TESTING AND QUALITY ASSURANCE:**

**HOW WILL WE ENSURE IT WORKS?**

1. **Unit Testing:**
   * Testing individual components, such as user registration, image upload functionality, and CNN model predictions.
   * Example: Ensure the image upload endpoint properly handles different image formats (e.g., JPEG, PNG) and edge cases (e.g., invalid file formats).
2. **Integration Testing:**
   * Verifying that different system components work together as expected. For example, testing the interaction between the frontend (UI), backend (Django server), and the model API.
3. **System Testing:**
   * Testing the entire web application to verify it meets all functional requirements and works in the intended environment.

**WHAT MAKES IT COMPLETE?**

To consider the project finished and accepted by stakeholders, the following conditions must be met:

* All core functionalities (registration, login, image upload, and result display) are implemented and tested.
* Image upload and processing times should be within an acceptable range.
* Integration and system tests are executed with successful outcomes.
* Documentation is complete, including user manuals and technical documentation for future maintenance.
* The user interface should be intuitive and responsive .
* All identified bugs and issues are resolved or documented with a clear action plan for future resolution.

**WHEN WILL WE TEST?**

The testing schedule is outlined as follows:

1. **Phase 1: Unit Testing**:
   * Duration: 3 weeks
   * After initial development of core features (login, image upload, CNN model), unit tests will be conducted.
2. **Phase 2: Integration Testing**:
   * Duration: 2 weeks
   * After individual components are developed, integration testing will ensure that all parts of the system work together as expected.
3. **Phase 3: System Testing**:
   * Duration: 2 weeks
   * Comprehensive testing of the full web application with end-to-end testing, including performance and security tests.
4. **Phase 4: User Acceptance Testing (UAT)**:
   * Duration: 1 week
   * After system testing, stakeholders will perform UAT to verify the system meets business requirements.

**WHAT IF SOMETHING GOES WRONG?**

If any issues are detected during testing or post-deployment, we will follow this process:

1. **Issue Reporting**:
   * Users and testers will report issues through a centralized issue-tracking system.
   * Each issue will be assigned a priority level (e.g., low, medium, high) based on its severity and impact on the system.
2. **Issue Tracking**:
   * Each issue will have a ticket created with clear steps to reproduce the issue, logs, and screenshots if necessary.
3. **Issue Fixing**:
   * Developers will work on fixing high-priority issues first and deploy bug fixes promptly.
4. **Verification**:
   * Once an issue is resolved, the fix will be verified through re-testing and integration testing.

**6. DEPLOYMENT AND IMPLEMENTATION PLAN**

**WHERE WILL IT LIVE?**

**Deployment Environment**:

* **Cloud-Based Deployment**: The application cam be hosted on a cloud platform such as AWS, Google Cloud Platform (GCP), or Microsoft Azure. This choice allows for scalability, flexibility, and ease of management.

**HOW WILL WE GET IT THERE?**

**Deployment Strategy**:

* **Phased Rollout**:
  + **Phase 1**: Deploy the application to a staging environment for internal testing and validation.
  + **Phase 2**: Conduct User Acceptance Testing (UAT) with a select group of stakeholders to gather feedback and make necessary adjustments.
  + **Phase 3**: Roll out the application to a limited user base (beta testing) to identify any remaining issues before full deployment.
  + **Phase 4**: Full production deployment after successful testing and feedback from users.

**WHAT IF SOMETHING GOES WRONG?**

**Backup Plans and Rollback Strategies**:

* **Backup Plans**:
  + Regularly scheduled backups of the application database and user data to a secure location
* **Rollback Strategies**:
  + Maintain a versioning system for the application. In case of a critical failure during deployment, the team can quickly revert to the last stable version.

**HOW WILL USERS LEARN TO USE IT?**

**Training and Onboarding Processes**:

1. **User Documentation**:
   * Create comprehensive user manuals and online help documentation that detail how to use each feature of the application.
   * Include FAQs and troubleshooting sections to assist users with common issues.
2. **Training Sessions**:
   * If possible, we can conduct live training sessions (webinars or in-person) for end-users to walk them through the application’s features and functionalities.
3. **Support Channels**:
   * Establish a support system (e.g., email support, or chat support) to assist users with questions or issues they may encounter while using the application.

**7.MAINTENANCE AND SUPPORT PLAN:**

**WHO'S IN CHARGE OF KEEPING IT RUNNING?**

Ongoing maintenance and support will be handled by a dedicated **Maintenance and Support Team**, which will consist of the following roles:

1. **System Administrators**: Responsible for managing servers, databases, and cloud infrastructure.
2. **Developers**: Responsible for maintaining and improving the web application, including implementing bug fixes, security patches, and adding new features based on user feedback.
3. **Customer Support Team**: A team dedicated to responding to user inquiries, issues, and feedback.
4. **Project Manager**: Oversees the coordination of the maintenance process, prioritizing tasks, ensuring efficient communication, and aligning resources with ongoing support needs.

**WHAT NEEDS TO BE DONE TO KEEP IT RUNNING?**

1. **Software Updates**:
   * Regularly update the application’s dependencies, libraries, and frameworks to ensure security and performance.
2. **Bug Fixes**:
   * Monitor issue tracking systems for reported bugs and prioritize fixes based on severity.
3. **Performance Monitoring**:
   * Conduct regular performance reviews and optimize the application as necessary.
4. **Data Backups**:
   * Schedule regular backups of the application’s database and user data to prevent data loss.
5. **User Support**:
   * Provide ongoing support to users through helpdesk systems, ensuring timely responses to inquiries and issues.

**HOW WILL WE KNOW WHAT USERS THINK?**

1. **User Surveys**:
   * Conduct periodic user satisfaction surveys to gather feedback on the application’s usability, features, and overall experience.
   * Use tools like Google Forms to create and distribute surveys.
2. **In-App Feedback Mechanism**:
   * Implement a feedback feature within the application, allowing users to submit comments or suggestions directly.
3. **User Interviews and Focus Groups**:
   * Organize interviews or focus groups with a representative sample of users to gather qualitative feedback on their experiences and suggestions for improvement.

**WHAT ARE OUR SERVICE COMMITMENTS?**

**Service-Level Agreements (SLAs)**:

1. **Response Times**:
   * **Critical Issues** (e.g., application downtime, data loss): Response within 1 hour, resolution within 4 hours.
   * **High-Priority Issues** (e.g., major functionality failure): Response within 2 hours, resolution within 1 business day.
   * **Medium-Priority Issues** (e.g., minor bugs affecting usability): Response within 4 hours, resolution within 3 business days.
   * **Low-Priority Issues** (e.g., feature requests, minor enhancements): Response within 1 business day, resolution based on prioritization.
2. **Performance Standards**:
   * **Uptime Commitment**: The application will maintain a minimum uptime of 99.9% per month, excluding scheduled maintenance.
   * **Performance Monitoring**: Continuous monitoring of application performance metrics, with alerts for any deviations from established thresholds.

**8. RISK MANAGEMENT:**

**WHAT COULD GO WRONG?**

1. **Technical Risks**:
   * **Model Performance**: The CNN model may not perform as expected, leading to inaccurate tumor detection results.
   * **Integration Issues**: Problems may arise when integrating different components of the application, such as the front-end and back-end systems.
2. **Operational Risks**:
   * **Staffing Issues**: Key personnel may leave the project, leading to a loss of knowledge and continuity.
   * **User Adoption**: Users may resist adopting the new system due to a lack of training or perceived complexity.

**HOW CAN WE PREVENT PROBLEMS?**

**Technical Risks:**

* **Model Accuracy**: Perform cross-validation and retrain with new, high-quality data to improve accuracy.
* **Security**: Conduct penetration testing, use data encryption, implement strong authentication.

**Operational Risks:**

* **User Adoption**: Invest in user-friendly design, conduct user testing, and provide clear onboarding and tutorials.

**WHAT'S OUR BACKUP PLAN?**

1. **Model Performance Issues**:
   * If the CNN model fails to meet performance expectations, revert to a previously validated model while a new model is developed and tested.
2. **Integration Failures**:
   * Maintain a backup of the last stable version of the application to roll back to if integration issues cannot be resolved quickly.
3. **User Adoption Challenges**:
   * If user adoption is low, implement targeted outreach initiatives, such as additional training sessions, tutorials, and one-on-one support.

**WHO'S WATCHING FOR PROBLEMS?**

* **Risk Management Officer**: A designated risk management officer will oversee the identification, assessment, and mitigation of risks throughout the project.
* **Project Manager**: The project manager will work closely with the risk management officer to ensure risks are monitored and communicated to the team.
* **Cross-Functional Team**: A cross-functional team comprising representatives from technical, operational, and financial areas will meet regularly to review risk status and discuss emerging issues.
* **Regular Reporting**: The risk management officer will provide regular updates to stakeholders on the status of risks and mitigation strategies, ensuring transparency and accountability.

**9. SECURITY AND PRIVACY:**

**DATA PROTECTION:**

1. **Encryption:**

* We can use HTTPS to secure communication between the user’s browser and the server. This ensures data in transit, like images and credentials, cannot be intercepted.

1. **Access Controls:**

* Implement a simple login and registration system with password hashing .
* Allow users to access only their images and results, ensuring user-specific directories or IDs.

1. **Data Privacy:**

* Add a privacy policy page clearly stating: Uploaded images are only used for tumor detection and not shared.

**SECURITY MEASURES**

1. **Firewalls**: A robust firewall will be deployed to protect the application from unauthorized access and potential threats
2. **Regular Security Audits:** The application will undergo regular security audits and vulnerability assessments to identify and mitigate potential security risks.
3. **User Authentication**: A secure login and registration process will be implemented to enhance user security

**INCIDENT RESPONSE PLAN:**

1. **Incident Detection**: The application will implement monitoring tools to detect security incidents, including data breaches and cyberattacks.
2. **Incident Response Team**: A dedicated incident response team will be established to handle security incidents promptly.
3. **Response Procedures**: Upon detection of a security incident, the following procedures will be followed:
   * **Containment**: Immediate measures will be taken to contain the incident and prevent further data loss or damage.
   * **Assessment**: The team will assess the scope and impact of the incident, including identifying affected systems and data.
   * **Notification**: If a data breach occurs, affected users will be notified within the required timeframe. Relevant authorities will also be informed if necessary.
   * **Remediation**: Steps will be taken to remediate the vulnerabilities that led to the incident, including applying patches, updating security measures, and improving policies.
   * **Post-Incident Review**: After resolving the incident, a post-incident review will be conducted to analyze the response, identify lessons learned, and update the incident response plan accordingly.

**10. LEGAL AND COMPLIANCE:**

**LICENSING:**

1. **Software Licenses:**

* Use open-source frameworks and libraries with permissive licenses, such as MIT or Apache 2.0. For example:
  + - Flask/Django: MIT License
    - TensorFlow/PyTorch (CNN algorithm): Apache 2.0 License

1. **Hardware Requirements:**

* No specialized hardware is needed for this project unless deploying locally on edge devices like GPUs.

1. **Image Data:**

* Ensure user-uploaded images are used only for the stated purpose (tumor detection) and not for resale or third-party sharing without explicit consent.

**REGULATORY COMPLIANCE**

1. **Data Usage:**
   * Clearly state that user-uploaded images are used **only for processing and not stored long-term.**
2. **Privacy Policy:**
   * Explain in a few sentences:

"Uploaded images are processed temporarily and deleted after the results are generated. No personal data is shared or stored."

**INTELLECTUAL PROPERTY:**

1. If sharing the project online (e.g., on GitHub):
   * Add a basic license to your repository to let others know how they can use your code.
2. Ensure any logos, names, or designs are original.
3. Make sure to include comments in your code .

**11. ENVIRONMENTAL IMPACT ASSESSMENT:**

**SUSTAINABILITY:**

1. **Minimizing Energy Consumption:**
   * Use cloud-based platforms like Google Colab or AWS for model training to avoid setting up power-intensive local servers.
   * Test and train the CNN model efficiently by limiting the number of epochs and using pre-trained models.
2. **Reducing Waste Generation:**
   * Store results digitally rather than printing them.
   * Avoid unnecessary data duplication by processing images in memory rather than saving multiple copies.

**GREEN IT PRACTICES:**

1. **Energy-Efficient Hardware:**
   * If using local systems, ensure you run the application on laptops or PCs with power-saving modes enabled.
2. **Virtualized Infrastructure:**
   * Deploy the application on cloud services that use **shared resources** (e.g., Heroku, AWS free tier). These providers typically operate energy-efficient, virtualized servers.
3. **Code Optimization:**
   * Write optimized code to reduce unnecessary processing.
   * For example:
     + Resize and compress uploaded images before running the CNN to reduce computational overhead.

**12. USER DOCUMENTATION:**

**USER MANUALS:**

* Introduction: Briefly explain the purpose of the project (tumor detection in iris images).
* System Requirements: Mention any browser or internet requirements for using the application.
* Steps to Use: Provide clear, numbered steps:
  + Register or log in to the system.
  + Upload an image of the iris.
  + View the detection results on the results page(detected using CNN model)
* Troubleshooting Tips: Include common issues (e.g., "File size too large" or "Unsupported file format”)

**ONLINE HELP AND TUTORIALS:**

1. **FAQs Section:**
   * Create a frequently asked questions section addressing common queries, such as:
     + What types of images are supported?
     + How accurate is the tumor detection?
2. **Tutorials:**
   * Develop step-by-step written tutorials for key tasks, such as:
     + Uploading an image and analyzing results.
     + Understanding the different outputs and what they mean.
3. **Video Guides:**
   * Create short video tutorials that visually guide users through the application. Consider covering:
     + A walkthrough of the user interface.

**USER INTERFACE DESIGN:**

1. **Intuitive Layout:**
   * Use a clean design with clear navigation:
   * Ensure buttons like "Upload Image" and "Get Results" are prominently displayed.
2. **Error Messages:**
   * Provide **clear feedback** for user actions:
     + "Registration successful!"
     + "Error: Please upload a valid image file."

**13. PROJECT MANAGEMENT AND MONITORING:**

**PROJECT PLANNING:**

1. **Task Breakdown**: Divide the project into manageable tasks. For example:
   * Research and review existing literature on iris tumor detection.
   * Design and develop the user interface.
   * Implement the CNN algorithm for tumor detection.
   * Conduct testing and validation of the application.
   * Prepare user documentation and tutorials.
2. **Timelines**: Create a timeline for each task. Use a timeline chart to visualize the project schedule, indicating start and end dates for each task.
3. **Resource Allocation**: Identify the resources required for each task, including:
   * Team members and their roles .
   * Tools and technologies (e.g., programming languages, frameworks, cloud services).

**RISK MANAGEMENT:**

1. **Identify Potential Risks**:
   * **Technical Risks**: Issues related to algorithm accuracy, integration problems, or software bugs.
   * **Time Constraints**: Delays in task completion due to unforeseen challenges or team availability.
   * **Resource Availability**: Lack of access to necessary tools, technologies, or team members.
   * **User Adoption**: Difficulty in attracting users to test and provide feedback on the application.
2. **Develop Mitigation Strategies**:
   * **Technical Risks**:
     + Regularly conduct code reviews and testing to catch issues early.
   * **Time Constraints**:
     + Prioritize tasks based on their importance and dependencies.
   * **Resource Availability**:
     + Ensure all team members have access to necessary tools and resources from the start.
     + Have backup plans for tasks that can be delegated to other team members if someone is unavailable.
   * **User Adoption**:
     + Gather feedback continuously and adapt the application based on user input to improve usability and features.

**14. TESTING AND QUALITY ASSURANCE:**

**UNIT TESTING:**

1. **Login/Registration Module:**
   * Test valid and invalid user inputs.
   * Verify successful login for registered users and error handling for incorrect credentials.
   * Test password validation (e.g., minimum length, special characters).
2. **Image Upload Feature:**
   * Verify that only supported image formats (e.g., .jpg, .png) are accepted.
   * Ensure the image is stored temporarily before processing, and confirm successful uploads.
3. **CNN Model:**
   * Test the model with sample images to ensure it provides correct classifications (tumor vs. no tumor).
   * Check the model's performance (accuracy, false positives/negatives).

**INTEGRATION TESTING:**

1. **Login and Image Upload Workflow:**
   * Test the flow from logging in, uploading an image, and processing it with the CNN model.
   * Ensure the application passes the uploaded image correctly to the CNN model for processing and that the results are returned correctly.
2. **Results Display:**
   * Verify that the results from the CNN model (e.g., tumor detected or not) are displayed on the result page.
3. **End-to-End User Journey:**
   * Test the full user journey. Ensure smooth transitions and that no data is lost between steps.
   * Check that session handling works properly, and the user is not logged out unexpectedly.