Project Design Phase-II

Technology Stack (Architecture & Stack)

Date: 31 January 2025

Team ID: [Your Team ID]

Project Name: GrainPalette - A Deep Learning Odyssey In Rice Type Classification Through Transfer Learning

Maximum Marks: 4 Marks

Technical Architecture:

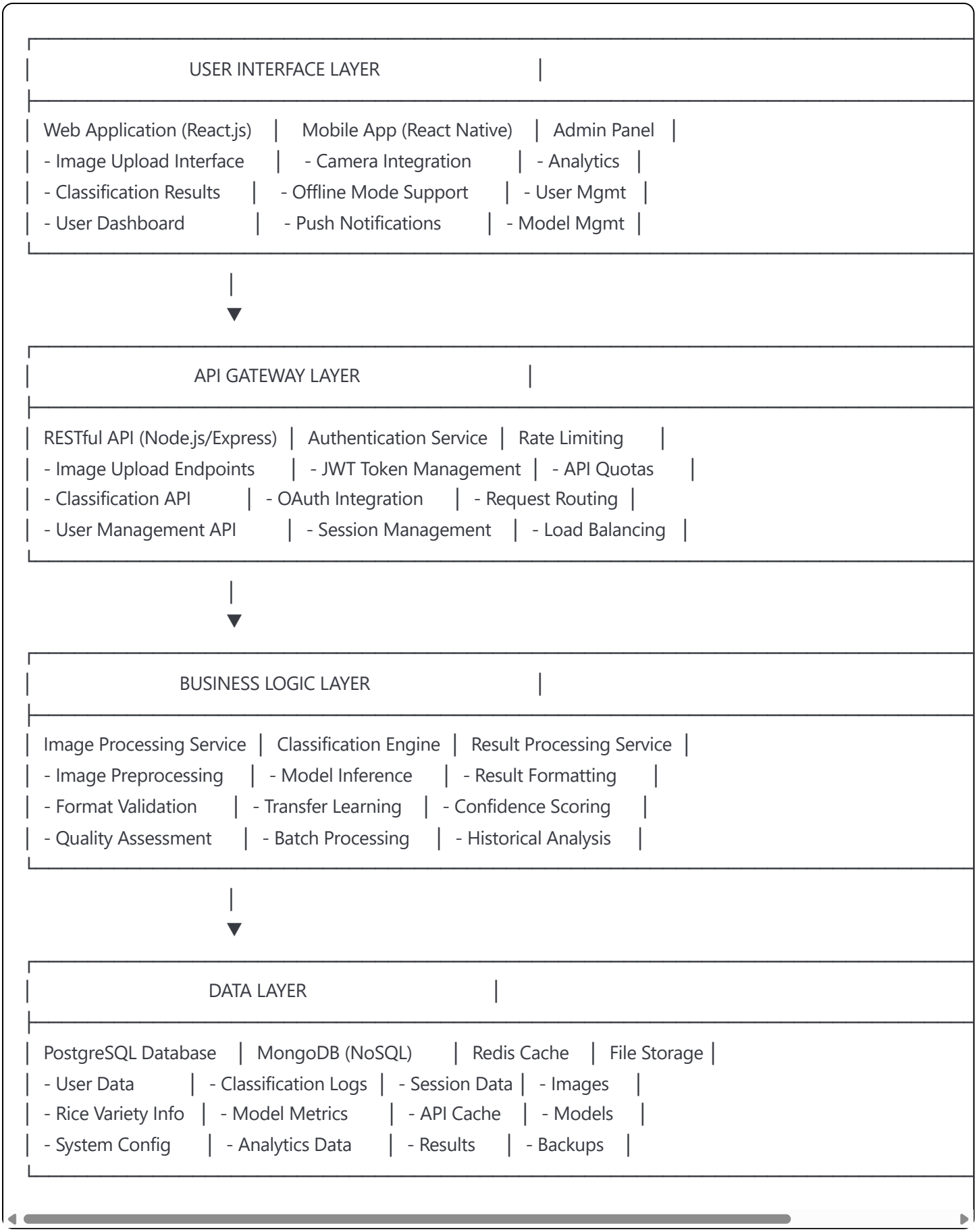

Table-1: Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web application for rice classification	React.js, HTML5, CSS3, JavaScript, Material-UI
2.	Mobile Application	Cross-platform mobile app	React Native, Expo SDK
3.	API Gateway	RESTful API services and routing	Node.js, Express.js, API Gateway
4.	Authentication Service	User authentication and authorization	JWT, OAuth 2.0, Passport.js
5.	Image Processing Service	Image preprocessing and validation	Python, OpenCV, PIL (Pillow)
6.	Classification Engine	Deep learning model for rice classification	TensorFlow, Keras, Transfer Learning (ResNet50, VGG16)
7.	Result Processing Service	Classification result formatting and analysis	Python, NumPy, Pandas
8.	Primary Database	Relational data storage	PostgreSQL
9.	Document Database	NoSQL data storage for logs and analytics	MongoDB
10.	Cache Layer	High-speed data caching	Redis
11.	File Storage	Image and model file storage	AWS S3, Google Cloud Storage
12.	Model Training Pipeline	ML model training and deployment	Python, TensorFlow, MLflow, Kubeflow
13.	Monitoring & Analytics	System monitoring and performance tracking	Prometheus, Grafana, ELK Stack
14.	External API-1	Image enhancement and processing	Google Vision API
15.	External API-2	Nutritional data integration	USDA Food Data Central API
16.	Container Orchestration	Application deployment and scaling	Docker, Kubernetes
17.	Cloud Infrastructure	Cloud hosting and services	AWS EC2, Google Cloud Platform, Azure

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Deep learning framework for model development Web framework for API development Frontend library for UI development Database management system	TensorFlow, Keras Express.js, Node.js React.js, Material-UI PostgreSQL, MongoDB
2.	Security Implementations	JWT token-based authentication Data encryption at rest and in transit Input validation and sanitization API rate limiting and DDoS protection OWASP security compliance	JWT, bcrypt SSL/TLS, AES-256 Joi validation, express-validator express-rate-limit, helmet.js OWASP ZAP, security audits
3.	Scalable Architecture	Microservices architecture for independent scaling Container-based deployment Auto-scaling based on demand Load balancing across multiple instances	Docker containers Kubernetes orchestration AWS Auto Scaling Groups NGINX load balancer
4.	Availability	Multi-region deployment for high availability Database replication and backup Health checks and monitoring Disaster recovery mechanisms	AWS Multi-AZ deployment PostgreSQL streaming replication Prometheus health checks Automated backup systems
5.	Performance	Content Delivery Network for global access Database query optimization Image processing optimization Caching strategies Asynchronous processing	AWS CloudFront CDN Database indexing, query optimization OpenCV optimizations Redis caching Message queues (RabbitMQ)
6.	Machine Learning Operations	Model versioning and deployment Continuous integration for ML Model monitoring and drift detection A/B testing for model performance	MLflow for model management Jenkins CI/CD pipeline Data drift monitoring Feature flags for model deployment
7.	Data Privacy & Compliance	GDPR compliance implementation Data anonymization techniques User consent management Data retention policies	Privacy-by-design architecture Data masking algorithms Consent management platform Automated data deletion
8.	Mobile Optimization	Progressive Web App capabilities Offline functionality Push notifications Camera integration	Service workers Local storage, IndexedDB Firebase Cloud Messaging React Native Camera API

Infrastructure Configuration:

Local Server Configuration:

- **Development Environment:** Docker containers running on local machine
- **CPU:** Intel i7 or AMD Ryzen 7 (minimum 8 cores)
- **RAM:** 16GB minimum, 32GB recommended
- **Storage:** 500GB SSD for development data
- **GPU:** NVIDIA RTX 3060 or better for model training

Cloud Server Configuration:

- **Production Environment:** AWS/GCP/Azure cloud infrastructure
- **Compute:** Auto-scaling groups with t3.medium to c5.xlarge instances
- **Storage:** S3/Cloud Storage for files, EBS for database storage
- **Database:** RDS PostgreSQL (Multi-AZ), MongoDB Atlas
- **CDN:** CloudFront/Cloud CDN for global content delivery
- **Load Balancer:** Application Load Balancer with SSL termination

---

References:

- https://c4model.com/
- https://tensorflow.org/guide/keras/transfer_learning
- https://aws.amazon.com/architecture/
- https://cloud.google.com/architecture/
- https://kubernetes.io/docs/concepts/architecture/
- https://mlflow.org/docs/latest/index.html