1.Digital Platform for Agricultural Feedback & Resource Coordination (Team 2)

Mentor: Riziki Kagabo **Email**: kagaboriziki@gmail.com Tel: 0787664342
Team leader: Fideli Nsanzumuhire **Email:**fidelensanzumuhire9@gmail.com
Duration: 2 Weeks
Team Size: 3–5 members

Challenge Overview

Develop a **Digital Agricultural Platform** that connects farmers/livestock keepers with district officials to streamline concerns, harvest reports, and resource requests (seeds, fertilizers, veterinary services).

Key Features

1. **Farmer/Livestock Keeper Portal**
   - Submit concerns (pests, diseases, drought) with photos/location tags.
   - Report harvests/livestock production metrics (yield, milk/meat production).
   - Request resources (seeds, fertilizers, equipment, vet services).

2. Real-Time District Dashboard

- Officials view submissions on a map with priority alerts (e.g., disease outbreaks).
- Assign responses (e.g., dispatch vet, approve seed requests).
- Track resolution status (Pending → Approved → Delivered).

3. Resource Coordination System

- Inventory management for district agricultural stores.
- Automated SMS alerts to farmers when inputs are ready for pickup.

4. **Tasks for Data Science and AI & ML**

Real-Time Dashboard (Dashboard: Plotly Dash or streamlit (more control over layout, interactivity) shows, but not limited to, the following insights:

1. **Header / KPIs**
   - **Show key metrics as cards:**
     - Total Harvest Reported
     - Total Livestock Production
     - Pending Resource Requests
     - Delivered Requests
2. **Live Map (Folium)**
   - Shows farmer issue locations with color-coded markers.
   - Clicking a marker displays the issue type and photo link (optional).

3. **Analytics Charts (Dynamic)**
    - ○ Line Chart → Yield trends over time.
    - ○ Bar Chart → Most reported issues (e.g., pests, drought).
    - ○ Pie Chart → Resource request statuses (Pending/Approved/Delivered).
4. **Prediction Panel (Visual)**
    - ○ A highlighted card with Predicted Next Yield.
    - ○ Mini forecast line chart showing predicted values.
5. **Export Section**
    - ○ Button to download PDF analytics report.

Bonus Features (Optional)

- **Mobile Offline Mode**: Submit reports without internet (syncs when connected).
- **IoT Integration**: Soil sensor data for personalized recommendations.
- **Chatbot**: AI assistant to answer common farming queries.

Technical Requirements

- **Frontend**: React.js / Vue.js / Flutter (for mobile).
- **Backend**: PHP (Laravel) / Node.js / Python.
- **Database**: PostgreSQL (for relational data) + Firebase (for real-time alerts).
- **Maps**: Leaflet.js or Google Maps API for location tracking.
- **Authentication**: SMS OTP for farmers, JWT for officials.
- **Deployment**: AWS/Azure or Laravel Forge.
- **Data science**

Deliverables

1. Working Prototype with:

- Farmer submission forms + district dashboard.
- Basic resource request workflow.

2. Dashboard: Plotly Dash or streamlit

3. Demo Video (3-5 mins) showing key features.

4. Pitch Deck (5 slides): Problem, solution, tech stack, impact.

Judging Criteria

✅ **Functionality**: Does it solve farmers’ communication gaps?

✅ **Usability**: Simple for farmers (low-literacy UX?), actionable for officials.

✅ **Innovation**: IoT/offline features?

✅ **Technical Soundness**: Scalable backend, secure data.

Timeline
Days 1-3: User research, wireframes, backend setup.
Days 4-7: Core features (submissions, dashboard).
Days 8-10: Notifications + testing.
Days 11-12: Polish + prep demo.
Days 13-14: Submit & present.

2. Citizen Engagement Platform(Team 4)

Mentor: Fiston Bakunzi Email:bakunzifiston@gmail.com Tel: 0783092757
Team Leader: MURAGIJIMANA Alpha **Email:** alphamuragijim@gmail.com Tel: 0727437987
Duration: 2 Weeks
Team Size: 3–5 members

Challenge Overview

Design and develop a **Citizen Engagement Platform** to improve communication between citizens and local government. The platform should streamline issue reporting, enhance transparency, and ensure accountability across administrative levels (Cell → Sector → District).

Key Features

1. **Issue Submission & Tracking**
   - Citizens submit complaints/requests (location, category, description, images).
   - Real-time status tracking (Pending → In Progress → Resolved).
   - Visibility of delays (e.g., stuck at Sector level for 2 weeks).

2. **Multi-Level Government Visibility**
   - Automatic escalation of unresolved issues to higher authorities (Cell → Sector → District).
   - Dashboard for officials at each level to view/act on issues.

3. **Feedback on Public Services**
   - Star ratings (1-5) and comments for district services.
   - Feedback visible to the Mayor/department heads.

4. **Department Directory**
   - Searchable list of departments, services, and contacts.

Tasks for Data Science and AI & ML
Real-Time Dashboard (Dashboard: Plotly Dash or streamlit (more control over layout, interactivity) shows, but not limited to, the following insights:

1. Issue Analytics Dashboard (For Officials)

Build a real-time, dynamic dashboard (using Plotly Dash or Streamlit) that shows:

- **Issue Volume Trends:**
    - Number of new issues submitted daily/weekly by category.
    - Heatmaps showing issue density by location (Cell → Sector → District).
- **Resolution Metrics:**
    - Average resolution time per category and administrative level.
    - Percentage of issues escalated vs resolved at each level.
- **Delay Tracking & Escalations:**
    - Visual alerts for issues stuck unresolved beyond a threshold (e.g., 2 weeks).
    - Automatic highlighting of bottleneck departments or levels.
- **Citizen Feedback Summary:**
    - Average star ratings by department.
    - Sentiment analysis of comments (basic NLP to gauge positive/negative feedback).
    -

2. Predictive & Automation Features

- Use simple ML models (e.g., decision trees or logistic regression) to predict the likelihood of issue escalation based on past data (issue type, submission time, department workload).
- Automatically prioritize issues that need urgent attention or likely to escalate.
- Implement text classification to categorize issues submitted via free-text descriptions for faster routing.

Bonus Features (Optional)

- Automated SMS/Email notifications for status updates.
- Analytics dashboard (e.g., resolution time, frequent issues).
- Mobile app (React Native/Flutter) alongside the web platform.

Technical Requirements

- **Frontend**: React.js / Vue.js / Angular (or mobile frameworks).
- **Backend**: PHP (Laravel) / Node.js / Python / Java.
- **Database**: PostgreSQL / MySQL / MongoDB.
- **Authentication**: Auth for citizens & officials.
- **Deployment**: you can use any hosting service of your choice
- **Data science**

Deliverables

1. **Working Prototype** (Web/Mobile App) with core features.
2. Dashboard: Plotly Dash or streamlit
3. **Demo Video** (3-5 mins) showcasing functionality (optional).
4. **Pitch Presentation** (5-7 mins) covering: Problem, solution, tech stack, and scalability.

Judging Criteria

✅ **Functionality** (Does it solve the problem?)
✅ **UX/UI** (Intuitive for citizens & officials?)
✅ **Innovation** (Unique features?)
✅ **Technical Execution** (Code quality, scalability).
✅ **Presentation** (Clarity, demo effectiveness).

Timeline

Days 1-3: Ideation, wire framing, backend setup.
Days 4-10: Core features implementation Testing + feedback integration.
Days 11-12: Polish + prep demo.
Days 13-14: Final submission & presentations.

3. Dynamic Youth Profiling System for Employment & Skills Mapping (Team 3)

Mentor: Abdoul Nyabyenda Email: abdoulnyabyenda@gmail.com Tel:0780463691
Team Leader: Enock NIYONSABA Email: enockccg28@gmail.com Tel:
Duration: 2 Weeks
Team Size: 3–5 members

Challenge Overview

Build a **Dynamic Youth Profiling System** to map skills, employment status, and entrepreneurial activities of youth in a district. The platform will help policymakers, employers, and training institutions connect with talent efficiently.

Key Features

1. Comprehensive Youth Profiles
- **Unemployed Youth**: Track education, skills, CVs, location, age, and contact info.
- **Employed Youth**: Log job type (permanent/casual), employer details, and skill utilization.
- **Entrepreneurs/Self-Employed**: Record business type, revenue, and support needs.

2. Real-Time Updates
- Youth can update their profiles (new skills, job changes, business growth).
- Automated reminders for profile verification every 6 months.

3. Advanced Search & Matching
- Employers/training programs can filter candidates by skills, education, or location.
- AI-driven job/business opportunity recommendations for youth.

4. Tasks for Data Science and AI & ML

Real-Time Dashboard (Dashboard: Plotly Dash or streamlit (more control over layout, interactivity) shows, but not limited to, the following insights:

1. **KPIs (Cards)**
    - Total youth registered
    - % unemployed vs employed
    - Top 3 demanded skills
    - Number of businesses started this month
2. **Analytics Visuals**
    - Bar Chart – Skill gap: Demand vs. supply
    - Pie Chart – Employment status distribution
    - Line Chart – Youth employment trends over time
    - Map – Location-wise employment heatmap (using Folium)
3. **Prediction Panel**
    - "Job Readiness Index" per candidate
    - Predicted future demand for top skills (line chart with forecast)

Bonus Features (Optional)

- **Mobile App**: For easy profile updates via SMS/USSD in low-connectivity areas.
- **Integration with LinkedIn/Job Portals**: Pull verified work history.
- **Skill Certification**: Partner with training centres to validate competencies.

Technical Requirements

- **Frontend**: React.js / Vue.js / Flutter (for mobile).
- **Backend**: PHP (Laravel) / Node.js / Python (Django/Flask).
- **Database**: PostgreSQL (structured data) + MongoDB (for CV/text analysis).
- **Authentication**: OAuth/JWT + role-based access (youth, employers, admins).
- **AI/ML**: (Optional) Scikit-learn/TensorFlow for skill matching.
- **Deployment**: AWS/Azure, or Laravel Forge for PHP stacks.
- **Data Science**

Deliverables

1. Working Prototype with:
   - Youth profile creation/editing.
   - Search/matching functionality.
   - Admin dashboard.
2. Dashboard: Plotly Dash or streamlit
3. **Demo Video** (3-5 mins) showcasing workflows.
4. **Pitch Deck** (5-7 slides) covering:
   - Problem, solution, tech stack, and scalability.

Judging Criteria

✅ **Functionality**: Does it effectively map youth skills/employment status?

✅ **User Experience**: Intuitive for youth, employers, and officials?

✅ **Innovation**: Unique features (e.g., AI matching, mobile integration)?

✅ **Technical Execution**: Code quality, security, scalability.

✅ **Impact**: Potential to bridge employment gaps.

Timeline

Days 1-3: Research, wireframing, database design.
Days 4-7: Core profiling + search features.

Days 8-10: Dashboard + AI matching (if applicable).
Days 11-12: Testing + polish.
Days 13-14: Final submissions & presentations.

4. Digital Tracking Solution for Health Service Transparency (Team 1)

Mentor: Fiston Munyampeta **Email**: mfiston2020@gmail.com **Tel**: 0782009474
Team Leader: Cyprien Yankurije **Email:** yankurijecyprien76@gmail.com **Tel**: 0789294965
Duration: 2 Weeks
Team Size: 3–5 members

Challenge Overview
Your team will build a **Digital Tracking System** to ensure transparency in the distribution of food aid for malnourished children. The platform will track food from allocation to delivery, preventing diversion and ensuring accountability.

Key Features

1. End-to-End Tracking
- Record food aid from warehouse allocation → distribution centres → beneficiaries.
- Scan QR codes/barcodes at each checkpoint to log location, time, and responsible personnel.

2. Transparency & Audit Trail
- Every transaction/modification is recorded on an immutable ledger (e.g., blockchain or secure database).
- Citizens can verify food aid status via a public portal.

3. Citizen Feedback Loop
- Beneficiaries confirm receipt via SMS/USSD/web app.
- Anonymous reporting for missing/delayed supplies.

4. Leader Dashboard
- Real-time monitoring for officials (Sector/Cell/District levels).
- Alerts for delays or irregularities (e.g., aid stuck at a checkpoint).

Tasks for Data Science and AI & ML
Real-Time Dashboard (Dashboard: Plotly Dash or streamlit (more control over layout, interactivity) shows, but not limited to, the following insights:

● Live tracking map of shipments (using Leaflet or Folium with GIS data), showing current locations of food aid trucks or packages. ● **Supply chain KPIs as cards:**

  - Total aid dispatched vs. delivered
  - Average transit time per checkpoint
  - Number of delayed shipments
  - Number of reported issues (missing/delayed)

- **Trend charts:**
  - Daily shipments over time
  - Delay patterns by location/checkpoint
  - Feedback reports count and categories (anonymous complaints)
- **Alerts panel:**
  - Flag suspicious shipments (e.g., repeated delays or frequent lost reports)
  - Notify officials immediately when irregularities are detected

2. Fraud Detection with Simple Machine Learning

● Use anomaly detection models (Isolation Forest or Local Outlier Factor) to flag shipments with unusual delay times or repeated 'lost' status. ● **Model inputs can include:**

  - Transit time between checkpoints
  - Frequency of reported issues per route
  - Number of times a shipment changes responsible personnel unexpectedly

  This will automate flagging suspicious cases for officials to investigate.

Bonus Features (Optional)

- **GIS Integration:** Map-based tracking of aid trucks/warehouses.
- **Automated SMS Alerts:** Notify beneficiaries when aid is dispatched.
- **Fraud Detection:** Flag suspicious patterns (e.g., repeated "lost" shipments).

Technical Requirements

- **Frontend**: React.js / Vue.js / Flutter (for mobile).

-**Backend**: PHP (Laravel) / Node.js / Python / Java.

- **Database**: PostgreSQL / MySQL (for relational data) or MongoDB (for flexible logs).
- **Authentication**: Role-based access (citizens, distributors, officials).
- **Security**: End-to-end encryption for sensitive data.
- **Deployment**: Hosted demo (AWS, DigitalOcean, Laravel Forge).
- **Data science**

Deliverables

1. **Working Prototype** (Web/Mobile App) with:
   - Food tracking workflow.
   - Citizen feedback mechanism.
   - Admin dashboard.
2. Dashboard: Plotly Dash or streamlit
3. **Demo Video** (3-5 mins) walking through key features.
4. **Pitch Deck** (5-7 slides) covering:
   - Problem, solution, tech stack, and scalability.

Judging Criteria

✅ **Functionality**: Does it prevent diversion and ensure transparency?

✅ **User Experience**: Intuitive for citizens, distributors, and officials?

✅ **Innovation**: Unique tracking/reporting features?

✅ **Technical Execution**: Code quality, security, scalability.

✅ **Impact**: How well does it solve the stated problem?

Timeline

- **Days 1-3**: Research, wireframing, backend setup (Laravel/other).
- **Days 4-7**: Core tracking + feedback features.
- **Days 8-10**: Dashboard + testing.

- **Days 11-12**: Polish + prep demo.
- **Days 13-14**: Final submissions & presentations.