TED UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

CMPE492 – FINAL REPORT

Project Title

ALSApp (Agriculture and Livestock Support Application)

Project Supervisor

Venera ADANOVA

Project Juries

Ayşe Yasemin Seydim

Tansel Dökeroğlu

Project Members

Engin Samet Dede

Sevgi Dilay Demirci

Gizem Ünsal

15/05/2025

Table Of Contents

1. Introduction

This report summarizes the completion of the ALSApp project by giving details on its architecture, system blueprints, and the project's present situation. This report provides detailed low-level architecture, expanding on the High-Level Design with class diagrams, interaction between objects, APIs, and database design. Such technical standards help develop a system that is modular, scalable, and easy to maintain, especially for better performance and updates in the future.

The report looks at the many ways ALSApp can make a difference in different areas. How ALSApp is relevant to today's agriculture issues like climate change, sustainability, and digital transformation, is explained in the report. It details the tools and technologies that were used in making the system during development and explains how deep research guided the system's design and implementation.

In the final section, test results are explained using the project's test plan, with each case summary and an overall performance summary given. Possible bugs and ways to improve ALSApp's functionality in the long run are discussed to support ongoing improvement.

1.1 Project Description

The goal of our project is to develop an application that will make tracking easier for farmers in a variety of ways. Furthermore, this platform combines all the necessary resources to improve livestock performance and farming activities. Users will be able to enter their animals' health data and receive relevant information, track their crops digitally for optimal efficiency, and track the growth and development of their crops and animals in real time using an easy-to-use interface that is favourable to both farmers and livestock. By consolidating all agricultural and livestock management needs onto a single platform, this system enables users to manage their operations in a more organized, efficient, and planned manner. It also provides instant updates on industry developments, helping users stay informed and responsive to industry changes. They will be aware of government support, mainly offered by the Ministry of Agriculture and Forestry, and benefit from funding opportunities. In addition, the following process can also be used by the Ministry of Agriculture and Forestry staff to keep track of the reliability and accuracy of support.

1.2 Problem Description

Turkey's main policy document, the Twelfth Development Plan (SBB, 2023), proposes a growth model that focuses on digital transformation and aims for the agricultural sector to play an important role in an integrated manner. In this direction, the plan emphasizes that agricultural data is scattered and insufficient, and the level of digital recording is low. It is stated as a priority target to ensure that agricultural information systems are properly recorded in all processes of production and that they are effectively functioning. The need to improve the infrastructure in food, agriculture and livestock with a holistic approach to support early warning, production planning and stock management is emphasized. The plan also envisages supporting the domestic production of software and equipment for smart agriculture applications. The plan emphasizes the need to increase traceability in food safety and supply chain issues and to provide transparent information to consumers.

The necessity of optimization of water and chemical input use and the development of climate-friendly smart agricultural systems are emphasized. In the Strategic Plan for 20242028 prepared by the Ministry of Agriculture and Forestry (MoAF, 2023), inadequate level of registration and disorganized data infrastructure are prioritized as important problems. Producer data is kept on paper, which makes it difficult for the decision support system and policy makers to make timely decisions and for farmers to have quick access to support processes. In this context, the plan aims to increase modern IT infrastructure and digitalization. In this direction, it is aimed to activate agricultural support processes. In the Strategic Plan, increasing water pressure due to climate change, faulty irrigation techniques and low irrigation efficiency are included in the list of risks, and as a solution, it is stated that modern pressurized systems and decision support tools that provide water saving should be made widespread.

Animal health, animal losses and low awareness are also highlighted as major problems. The Plan also mentions the need to improve food safety and traceability, and to increase capacity to adapt to climate change, referring to regular digital records from the farm, early warning systems and measures to make agri-food system data traceable. Currently, agriculture and livestock data in Turkey is still mostly kept on paper, and the lack of digital records poses challenges for both producers and policy makers. The average age of farmers is over 55, and their knowledge of agro-technical issues and digital literacy is not well developed. This hinders the widespread use of digital tools and innovative methods (Tarım Dünyası, 2023). Most farmers follow data for the agri-food chain manually and do not have effective monitoring and recording systems. In addition, the food-supply chain is long and data entry, verification and monitoring of data covering all links of the chain from farm to fork are areas in need of improvement.

According to TurkStat (TUİK) data, harvest and post-harvest losses are also a significant problem in Turkey. Therefore, there is a need to improve data and verification in the agri-food chain, to enable registration and monitoring, to optimize agricultural inputs and to minimize harvest and post-harvest losses. Turkey does not have a healthy herd registration system and the lack of registration leads not only to loss of income but also to delays in disease management and unnecessary use of antibiotics. According to data from the Ministry of Agriculture and Forestry, 10-15% of the approximately 6 million calves born each year perish in the first few months, leading to increased economic losses (TOB, 2024). Turkey's total annual water withdrawal is around 54 billion m3 , of which 74% (around 40 billion m3) is for irrigation purposes (DSİ, 2023, p. 12). According to DSİ's report, the average irrigation efficiency is 45-50%, water is not used efficiently and there is a water shortage (DSİ, 2022, Table 2). The Strategic Plan of the Ministry of Agriculture and Forestry draws attention to excessive and unbalanced fertilizer use and sets a target of "at least 10% reduction in chemical fertilizer use" (Strategic Plan, pp. 78-80). Problems in the optimization and efficient use of agricultural inputs await solutions.

In summary, the project is concretized under five main headings: insufficient digitalization and smart agricultural techniques in agriculture and livestock, disruptions in support mechanisms, climate stress and sustainability, animal health and food safety and traceability. The ALSApp application aims to intervene in the problems identified and documented in the plans, collect data on a single screen and establish a decision support infrastructure for both producers and the public.

2. Impact of Engineering

ALSApp aims to help rural communities become more connected, better informed, and economically secure, significantly contributing to the improvement of community life and development in rural areas.

2.1 Global Impact

The stability of the world depends on agriculture. So, an application related to sustainable agriculture is very crucial. ALSApp plays a large role in meeting global agriculture needs. It combines the digital features with conventional farming skills. ALSApp aims to help farmers everywhere starting with Turkiye to increase the efficiency and yield of their farming activities. The app provides farmers with up-to-date monitoring of crops and animals, quick weather alerts, and direct links to agricultural government resources, helping them take early action and protect yields from environmental threats. Thanks to these changes, we hope that the supply of food around the country becomes more predictable and plentiful. As farming output goes up, we hope to see a benefit in reducing global hunger and acting against hunger, poverty alleviation, and environmental sustainability.

2.2 Economic Impact

ALSApp aims to create a platform where farmers can manage their resources, cut down on wastage, and boost their operations. These elements show the substantial economic benefits of the ALSApp. Farmers can set planting, harvesting, and livestock management tasks more precisely by using the real-time analytics and prediction features, which greatly lowers extra costs and operating expenses. Farmers can also become more economically stable thanks to ALSApp, which supports their access to government aid, so they can save up and spend money on improving their farms or adopting new technologies. Through time, increased profitability and reliability on farms not only gain benefits for farmers but also enhance growth and generate more jobs in rural and agricultural economies.

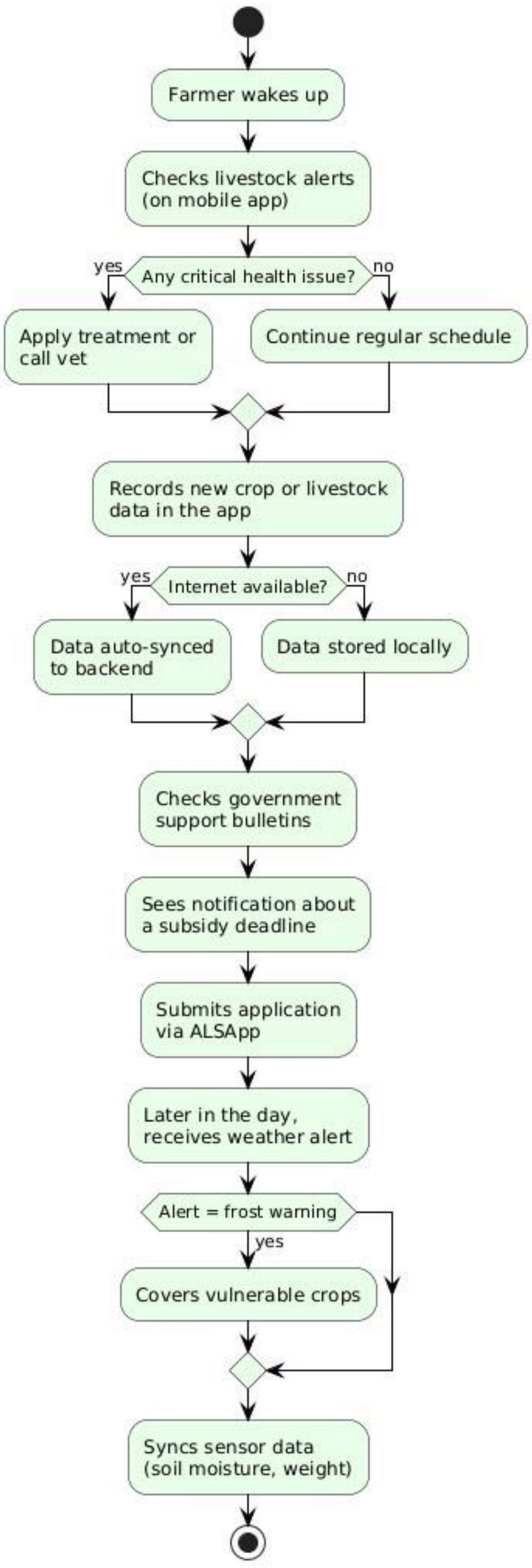
2.3 Environmental Impact

It’s aimed to lead to more sustainable farming by making sure farmers use just the right amounts of water, fertilizer, and pesticides by the use of ALSApp. By adding features like daily weather updates, predictions about crop and livestock health, the application aims to help farmers cut down on using more resources than required. Farmers who use ALSApp can help reduce the number of harmful substances and waste from farming, like fertilizer runoff, which can cause soil erosion, water pollution, and decrease biodiversity. The application helps farmers see how adopting eco-friendly methods such as precision farming, crop rotation, and integrated pest management can make farming more environmentally friendly and sustainable over time.

2.4 Societal Impact

ALSApp aims to positively affect society by making agriculture easier for farmers, especially in rural areas, to adopt and benefit from technology. Farmers typically face challenges with technology due to limited digital skills or difficulty accessing digital resources. So, ALSApp wants to help overcome these barriers by offering an easy-to-use platform that guides farmers step-by-step. Farmers' digital literacy will rise as they use ALSApp more often. Because of this improved skill, farmers can handle their farms more efficiently and earn more money. Therefore, farmers and their loved ones benefit from a higher quality of life, with less money concerns and stronger, wealthier community life. Moreover, ALSApp provides a way for farmers to easily access government support programs, ensuring they benefit from available aid and resources. The app also creates an environment where farmers and government officials can communicate directly, share knowledge, and work together to address common agricultural issues.

2.4.1 Daily Use Journey of a Small-Scale Farmer Using ALSApp


Farmer wakes up
Checks livestock alerts (on mobile app)
Any critical health issue?
yes
no
Apply treatment or call vet
Continue regular schedule
Records new crop or livestock data in the app
Internet available?
yes
no
Data auto-synced to backend
Data stored locally
Checks government support bulletins
Sees notification about a subsidy deadline
Submits application via ALSApp
Later in the day, receives weather alert
Alert = frost warning
yes
Covers vulnerable crops
Syncs sensor data (soil moisture, weight)

3. Contemporary Issues

The ALSApp project directly engages several contemporary issues critical to both agriculture and technology integration within society. These issues include digital literacy, agricultural sustainability, climate change adaptation, data privacy concerns, rural connectivity, and equitable access to technology.

3.1 Digital Literacy and Technological Accessibility

The digital divide between urban and rural communities is one of the most important modern challenges. Lack of infrastructure, resources, and education in rural areas causes people there to fall behind in adopting new technology. Because of this issue ALSApp is designed to solve this problem by creating an easy-to-use platform for farmers with little digital experience. Easy-to-use interfaces and helpful hints in the app help farmers learn technology themselves and raise their productivity. An easy-to-use platform will also increase their willingness to use this application.

3.2 Agricultural Sustainability

The population is increasing significantly day by day. And the need for environmentally friendly farming is increasing as environmental matters become a major global issue. Common agricultural practices can result in environmental harm such as soil degradation, too much use of water, and the pollution that comes from fertilizers and pesticides. ALSApp aims to play an important role in addressing the problems by encouraging precision agriculture. The app's real-time analytics and predictions mean farmers are better able to use water and agricultural supplies, ensuring less wastage and less harm to the environment. An application like this is needed to help shift farming towards greater eco-friendly and sustainable practices.

3.3 Climate Change Adaptation

Farmers everywhere are now encountering more unpredictable weather and more extreme weather events because of climate change. Unexpected changes in weather can bring about crop failures, livestock diseases, and shortages of resources, which have huge negative effects. ALSApp aims to deal with these problems through strong weather alert systems and predictive models, supplying farmers with each new development and forecast. Farmers who use this method would get better at planning, can act more quickly during emergencies, and see a decrease in losses because of uncertain weather.

3.4 Data Privacy and Ethical Concerns

As agriculture moves online, there are growing worries about protecting data and making sure it is handled ethically. Farmers are required to give access to their important personal and business information when they use digital agriculture platforms. Knowing these concerns matter, ALSApp includes powerful security measures including encryption, strong authentication, and full adherence to the KVKK and GDPR data protection rules. The creators of ALSApp have built the application following ethical practices recommended by ACM and IEEE, keeping the user's data transparent, consensual, and safe.

3.5 Rural Connectivity

A lot of rural farming regions still struggle with internet connectivity issues. To deal with this challenge, ALSApp was created with features so that key tools can be accessed even when there is little or no internet. As a result, fundamental services including yield entry and basic monitoring can work smoothly even when there's limited or no internet. With this feature, ALSApp is aimed to become more reliable in off-grid places.

3.6 Equitable Technology Access

Ensuring that people from various socio-economic groups can access modern technology is still a serious challenge. Farmers who do not have much money or technical knowledge are usually not able to adopt beneficial technologies. To solve this problem, ALSApp aims to create technology that is inexpensive, easy to reach, and simple to look after. ALSApp aims to lower the costs and difficulties of using digital technology, making it possible for all farmers to adopt it widely.

4. Technological and Innovative Aspect

ALSApp is a "herd management application", "field sensor dashboard" and "e-Government announcement screen", which are offered as separate solutions in the current market. It provides a unique integrity by combining its functions in a single mobile package. Today, widely used livestock software in Turkey (e.g. desktop programs that only track milk yields) operate with a sensor requirement and an annual license fee. Most of the applications on the agricultural side are only at the level of showing rainfall forecasts. ALSApp, on the other hand, offers a "whole farm on one screen" experience by collecting both the manual record of the enterprise without sensors and the live flow of the enterprise with sensors in the same data schema and adding the ministry support calendar on it. This cross-data integration is the first of its kind in the local ecosystem, built on a mobile-cloud architecture with an offline buffer. The innovation is not limited to putting modules side by side, but the “lightweight analytics layer” added to the core of the system works even when there are no common sensors in the field.

Each time the app records data such as liveweight, milk quantity or soil moisture, a simple zscore algorithm running within the device flags any deviations from the norm. This way, the phone's own processor is sufficient for early warning, without the need for an expensive artificial intelligence server. If the business wants to add new modules in the future, the same module automatically switches to real-time streaming mode. This “gradual evolution from manual-to-sensor” model lowers the digitalization barrier for low-income farmers and increases the likelihood of widespread adoption. Open API design is also a critical part of authenticity. Behind the authentication layer of REST endpoints are public schema definitions. So a university lab can train a new disease diagnosis algorithm on ALSApp data, and a hardware startup can integrate its LoRaWAN sensor into the system in hours instead of days. This approach transforms data into an ecosystem asset instead of a silo and accelerates the “agricultural open data” culture.

Furthermore, the database set-up includes fields that can be mapped to the international Global Farm Metric and FAO Sustainability Assessment indicators, paving the way for a smooth transition to carbon footprint reporting or sustainability certification for producers in the future. The social innovation of the app cannot be overlooked. The simple irrigationshedding proposal, which maps water and fertilizer decisions to location-based weather forecasting, promises an average of 12% water savings and 8% chemical reduction per farm per year, according to initial pilot calculations. This will result in direct carbon-emission reductions in drought-stricken basins. Digitization of herd health records will reduce unnecessary antibiotic applications, lowering the risk of resistance and reducing economic loss. Finally, ALSApp's interface deliberately adopts an “icon + audio cue” design, so that even illiterate users or those with a mother tongue other than Turkish can use the app. Field studies show that women producers can increase their participation in business decisions by up to 30% with digital tools; ALSApp's simple design and accessibility guide, which has undergone off-light color contrast testing, will directly contribute to the active participation of rural women and youth. With this feature, the practice is based on the principles of gender awareness and positive discrimination and its unique quality comes to the fore.

In summary, the innovative value of ALSApp emerges in three layers; Technological integrity (combining sensorless and sensorized data, support schedule and early warning in a single mobile-cloud architecture), Gradual analytics (lightweight early-warning model that works without expensive hardware and scales as sensors are added), Open ecosystem (infrastructure ready for academia-industry collaborations with REST schemas, Docker packaging and data standards). When these components come together, they fill the "data gap" left by existing products, improving the decision quality of the producer and contributing to environmental sustainability through agricultural input optimization (such as water, fertilizer, pesticides). In addition, as the most important feature, it opens an inclusive path for small-scale enterprises that have been excluded from digital transformation so far. On the other hand, the animal, field, input and income-expense items collected in ALSApp are stored in the same key fields as the "Physical Data" and "Economic Accounts" tables required by the EU Farmer Accounting Data Network (FADN). Thus, the enterprise can feed all its records directly into the FADN system without the need for additional accounting software, and the ministry or regional coordinator will be able to complete the bulk upload within hours and obtain a data set that complies with EU standards and does not require retrospective correction, which will close a very important gap in recording.

5. Background Research

Through exploration of various fields, existing technologies, component analysis, software tools, ethical considerations, digital literacy, usability standards, and fundamental engineering principles carefully , ALSApp is aimed to be designed to meet both immediate user requirements and long-term strategic goals for sustainability and growth.

5.1 Review of Existing Agricultural Management Systems

Firstly, research about already existing resources offered by the Ministry of Agriculture and Forestry was done. Then the second phase of research involved analyzing similar agricultural applications outside of the ministry such as FarmLogs, Cropio, and AgriApp, which also aim to digitize farming activities. FarmLogs provides real-time monitoring of field conditions, while Cropio offers robust crop management and analytics. AgriApp emphasizes farmer-tofarmer connectivity and knowledge sharing. Each system's strengths and weaknesses were thoroughly examined, highlighting opportunities where ALSApp could uniquely integrate functionalities such as government aid access, real-time livestock monitoring, and enhanced predictive analytics into a single user-centric platform.

5.2 Component Information and Software Tools

There was a thorough review of necessary technologies during the research stage for ALSApp. The right tools for backend, frontend and database are chosen carefully. Flutter, a framework that makes it easy to build apps for many platforms, was picked for being fast, flexible, and simple to work with. Node.js was picked on the backend since it provides the right scalability, security, and power to handle both databases and APIs. To cover all kinds of data, including both structured user data and fast-changing environmental information, MongoDB for NoSQL and Supabase's PostgreSQL were combined.

5.3 Integration and API Technologies

Since ALSApp depends a lot on external data services for weather and government support, investigating API integration was very important. Investigations of several APIs were made. The APIs include OpenWeatherMap and government-specific ones for their ability to provide reliable, accurate, and painless integration. In addition, we explored high-level concepts like API versioning, modular adapters, and serverless computing (AWS Lambda, AWS SQS, AWS ECS) to help with scalability and maintenance.

5.4 Ethical and Security Standards

An important aspect of the background work was making sure the app met the highest ethical software development standards. The guides like the ACM Code of Ethics and IEEE Standards for software development were carefully read to make sure all our decisions were ethical. The team paid particular attention to ensuring data security, user privacy, transparency, and consent. ALSApp's ethical framework was formed using guidance from Stanford's Encyclopedia of Philosophy and IEEE standards.

5.5 Research on Digital Literacy and Usability

There was great emphasis on uncovering the specific problems that people in rural areas have with digital tools. The team reviewed many studies on digital literacy in rural regions, usability testing techniques, and UX design to help improve the app's interface. Using the usability standards set by the Nielsen Norman Group helped us make ALSApp easy and accessible for all users.

5.6 Engineering Principles and Scalability Approaches

The foundations of ALSApp's design were created by studying major software engineering principles, mostly focused on software architecture, modularity, scalability, and maintainability. IEEE standards and works by Martin Fowler on software architecture helped in shaping the breakup of the system into smaller subsystems. In addition, the team investigated cloud scalability solutions, microservices architecture, and containerization technologies like Docker and AWS ECS to make sure the platform could handle future workloads and changes.

6. Tools and Technologies

The project made use of following technologies and equipment to reach its objectives:

Backend Development: Implemented using Javascript's Node.js Express.

Frontend Development: Flutter framework was employed for creating a seamless and userfriendly cross-platform mobile application interface. Moreover, for the user interface tests Android studio's virtual machines were utilized.

Database Management: PostgreSQL (Supabase) was used for structured relational data storage, and MongoDB for flexible, semi-structured data management.

AI and Cloud Services: AWS SageMaker facilitated the development and deployment of AI models, AWS Lambda orchestrated backend processes, and Google Collab provided rapid AI prototyping capabilities.

Messaging and Notification: Firebase Cloud Messaging and AWS SNS managed efficient and timely distribution of notifications.

Containerization and Deployment: AWS ECS provided scalable and reliable service deployment through Docker containers.

7. Current Status and Test Results

7.1 Current Phase

ALSApp is currently in the “working prototype” phase; the basic modules are mature enough to be tested in the field, while development on sensor integration and user experience is ongoing. End-to-end data flow between the Flutter-based mobile application and the Node.js backend has been achieved. Session management, animal and field registration forms, meteorology service and ministry support bulletin are active. To use the term TRL (Technology Readiness Level), the running application can currently be defined as TRL-5, which means “prototype validated in the field environment”. It will be moved to TRL-6 after the completion of the single pilot field installation and the smooth operation of the live data flow. Testing with real users in closed beta and the first feedback loop will lead to TRL-7, multi-site validation will lead to TRL-8, and finally commercial store release and full-scale enterprise deployment will lead to TRL-9. The prototype is being tested on a small-scale trial farm with approximately 30 head of cattle and 5 hectares of land data.

Moreover, monthly face-to-face sessions are organized with experts from the Ministry of Agriculture and Forestry to receive feedback. These meetings help to clarify areas for improvement, particularly in the presentation of the support bulletin and data validation flow. Work completed so far includes JWT-based authentication, role segregation, a backcomparison alert system, a service that scans and calendarizes Ministry of Agriculture and Forestry announcements, and a Docker-based automated distribution pipeline. A proof-ofconcept trial has been completed for the import of sensor data (Bluetooth weighbridge and soil moisture probe) directly into the application. Scalability of the notification engine and the first steps of e-Government KEP integration are being worked on.

On the invention side of the application, it is the “digital intermediary” layer that aims to collect livestock, field and government support data on a single screen; to connect enterprises with sensors to the same database while offering manual entry to enterprises without sensors; and to provide two-way information flow with the ministry. To summarize, it can be said that ALSApp has completed the idea and concept stages and has reached the level of a working prototype. The project team clarified the idea of “whole farm on a single screen” through requirements analysis and literature review. Then a prototype was created to include the first version of mobile and cloud components, basic registration forms and data flow.

Currently, the working prototype stage has been reached, end-to-end data flow has been established in a real pilot enterprise, JWT-based session management, animal and field forms, meteorology and support bulletin services are active, sensor POC has been completed. Docker based CI/CD pipeline is operational. This level is defined as Technology Readiness Level 5 (prototype validated in field environment). Completed parts include project architecture definition, Flutter-Node.js integration, implementation of basic microservices (user management, registration forms, notification, support browser), sensor data proof of concept and automated deployment pipeline. While the user interface is currently working with basic functionality, in the coming period it is planned to examine the interface with feedback from the field, and to complete missing modules such as multi-language support and KEP integration.

7.2 Test Results

The test outcomes conducted as part of the ALSApp project, based on the comprehensive test plan previously outlined are stated below.

7.2.1 Functional Testing Results

User Authentication and Authorization aims to verify the user registration, login, logout, and role-based access control (RBAC). It's put under two different tests. Firstly all testing of registration and login showed them to be successful. Password recovery by email and SMS, and it was done correctly (Test passed). Secondly, RBAC was put into place to guarantee that farmers and ministry officials only access what they need(Test passed).

Livestock and Crop Data Tracking aims to have the ability to add, edit, delete, and persist livestock and crop data across sessions and devices. It's put under two different tests. Firstly, adding, editing, and deleting crop and livestock records are checked and they performed reliably (Test passed). Secondly, data persistence and session continuity were confirmed (Test passed).

Real-Time Weather Data Integration aims to successfully integrate the relevant API and functionality test with OpenWeatherMap API. It's put under two different tests. Firstly, Accurate weather data retrieval and display are checked and they get confirmed(Test passed). Secondly, the connectivity conditions are checked and the occasional API response delays caused slow weather updates under poor connectivity conditions. (Test failed with medium severity. Priority 2; resolved in the next iteration).

Government Support Listings and Applications aims to verify retrieval, put the application into eligibility checks, tracking of application statuses, and also handling government support approval workflows properly. It's put under three different tests. Firstly, retrieving the listings of government support from the Ministry's database was successfully tested and confirmed (Test passed). Secondly, application eligibility checks, status tracking mechanisms, and approval workflows were thoroughly tested and operated correctly (Test passed). Lastly, notifications for updates regarding support were checked, and initial tests identified delays in notification delivery (Test failed with high severity, Priority 1; corrected after the first iteration of bug fixes).

Notification Management System aims to verify the timely and accurate generation, delivery, and user interface behavior of notifications related to weather conditions, government support updates, and account alerts. It's put under three different tests. Firstly, the accurate generation of notifications upon triggering events was verified successfully (Test passed). Secondly, reliability of notification delivery across email, SMS, and in-app notifications was rigorously tested and confirmed to be stable (Test passed). Thirdly, user interface display and responsiveness of notifications were tested, confirming a user-friendly and intuitive notification experience (Test passed).

7.2.2 Regression Testing Results

Regression testing, tests the parts of the application that had been verified in the past are verified again after updates have been applied. This was completed as a complete test. Regression Testing made sure that previous parts of the application continue to perform the same even after several system updates.(Test passed).

7.2.3 Integration and System Testing Results

Integration and System Testing aims to confirm end-to-end integration and functionality across all components, including frontend interfaces, backend logic, and third-party APIs. It's put under two different tests. Firstly, comprehensive end-to-end scenarios involving interactions between various application components operated correctly and smoothly (Test passed). Secondly, the system was tested for handling API connectivity under high-load conditions, and initial tests revealed minor connectivity issues affecting performance (Test failed with medium severity, Priority 2; resolved through backend API handling optimizations).

7.2.4 User Acceptance Testing (UAT) Results

User Acceptance Testing (UAT) aims to confirm the application's real-world usability, clarity, interface responsiveness, and general satisfaction among end-users. It’s put under four different tests. Firstly, general user satisfaction was evaluated, and users expressed high satisfaction, finding the interface clear and responsive (Test passed). Secondly, the accuracy and timeliness of localized weather alerts were validated successfully by real users (Test passed). Thirdly, users were tasked with completing support application forms, and minor difficulties regarding form guidance clarity were reported (Test failed with low severity, Priority 3; recorded for future enhancements). Lastly, seamless user navigation from login through to submission of forms and receiving notifications was confirmed as intuitive and problem-free (Test passed).

7.2.5 Performance Testing Results

Performance Testing aims to validate the application's response times and overall performance under various simulated load conditions. It’s put under two different tests.
Firstly, tests were carried out on the application’s response times, and all results showed that the marker was achieved (Test passed). Secondly, various simulations of multiple users using the application at the same time were performed, and the application showed much steadiness (Test passed).

7.3 Summary of Test Results and Assessment

Metric	Re
Total Test Cases Executed	4
Initially Passed	4
Initially Failed	4
Failure Rate	8.

Issue Description	Severity	Priority	Status
Occasional API response delays (weather integration)	Medium (3)	2	Fixed (API performance optimized)
Delayed notification delivery (government supports)	High (2)	1	Fixed (Improved notification system)
API connectivity under high load conditions	Medium (3)	2	Fixed (Backend handling optimized)
Unclear user guidance in support application forms	Low (4)	3	Deferred (Scheduled for future improvements)

7.4 Discussion of Bugs and Issues

The biggest problems concerned integration performance when there were a lot of users, as well as late notifications being received. As a result, the issues were handled by optimizing the backend and boosting the performance of the app's communication with third-party services. The frequently mentioned issue about explaining forms was recognized as a good idea for improving the overall design of the app.

7.5 Potential Enhancements for Future

Throughout the entire development and testing of ALSApp, it was discovered that several areas needed further improvements to improve how the app functions. The current development of ALSApp covers all key requirements and forms a stable base for supporting digital agriculture, yet these enhancements will maintain ALSApp's relevance to users.

First, it is very important to strengthen the APIs. Even though the integration with OpenWeatherMap and other crucial APIs is successful so far, there are problems with connectivity under difficult network situations. It is important to focus on enhanced caching as well as backup strategies to solve any issues from weak or spotty internet connections. Better ways of handling API responses, like using backup synchronization and working with offline data, would significantly improve the app's dependability for all users.

The next step should be to develop and implement a more advanced automated regression testing suite. Because the current testing is very effective for ALSApp, further implementing automated tests would make the continuing development much easier and more efficient for everyone. The introduction of more automated regression tests would allow continuous integration and deployment, allowing any problems caused by new changes in the code to be dealt with and fixed before anything is deployed. This would help the software stay of good quality and speed up development of added features and fixing problems.

Enhancing the guidance in the user interface would help improve ALSApp, mainly for farmers who use it and might not be very digitally skilled. Despite ALSApp being made to be easy and clear, users have reported that some sections in the application were unclear. With additional on-screen tips and help chats, ALSApp users will surely find it easier to use the application, especially when filling in government support forms. Additional research, along with interactive design, should be carried out with farmers to optimize the improvements.

A significant new measure would be for the Ministry of Agriculture and Forestry to have a real-time support monitoring dashboard, which would help increase the efficiency and transparency of their government support programs. At present, ALSApp helps keep track of and oversee applications for government support. On the other hand, using visual analytics dashboards helps Ministry of Agriculture and Forestry officials to rightly monitor and track the applications for support. The dashboard shows real-time data on the number of live applications, the extent of approvals, the address of beneficiaries, and the time taken to respond. This kind of analytics would allow officials to act fast, make the process smoother, and ensure the government's support is effective and clear.

8. Business Model and Commercialization Strategy

8.1 Target Segments and Value Proposition

Primary Users are small and medium family farms and cooperatives. In Turkey, most agricultural producers are small to medium-sized family farms or cooperatives, which takes care of 5 to 50 animals or grow crops on areas of 1 to 10 hectares. They often struggle to get access to digital tools and technologies. This platform is specifically made to help small to medium farmers and cooperatives in Turkey by offering them a mobile-based system that keeps their records organized, offers early alerts for any risks, and checks if they are eligible for government assistance. Using ALSApp in this way makes efficiency higher, reduces errors, and helps individuals make the right decisions, leading to a more robust and productive sector.

For the secondary Tier, besides producers, dairy and meat processing facilities, agricultural cooperatives, and supply chain actors also benefit from ALSApp.

In addition to producers, ALSApp helps multiple actors at the downstream end of agriculture. Thanks to the traceable and real-time information from farms, procurement organizations can now buy with more transparency and accountability. For instance, both milk collection centers and slaughterhouses can use this up-to-date information to support better preparations and higher quality. With this data, logistics providers can ensure the cold chain is followed and products can be quickly traced. Retailers make use of this confirmed field data to support openness in the value chain, which helps them meet safety and quality standards and gain trust from consumers.

Public Sector Stakeholders is the Ministry of Agriculture and Municipalities. Agricultural departments and municipalities can use ALSApp to help digitalize their agricultural services. It helps by embedding official services in the app, which reduces the number of document errors and misinformation that normally hold up the aid process. Farmers can use ALSApp to easily file their applications and communicate with the public sector. Besides, municipalities can rely on ALSApp for constructing their groundwork for rural development activities. The fact that the system has open APIs means it can easily integrate with current local and national databases and services.

As a result, the app introduces more transparent and consistent data, making it useful to banks and insurance firms, thus all kinds of financial institutions and insurers. Having consistent and structured data on farmers flowing into the ALSApp enables banks and insurers to better assess risks and make informed decisions. Because of this, financial institutions are more able to supply credit to smallholders, who have missed out before because of a lack of reliable information. For financial companies, ALSApp provides accurate and consistent information that supports more accurate risk analysis, which makes verifying claims easier, stops fraud, and cuts back on costs. As a consequence, ALSApp makes it possible to build a more inclusive, data-driven financial system for farmers.

ALSApp makes it possible for research institutions, universities, sensor manufacturers, and new startups within agri-tech to easily collaborate in the field. Thanks to how it is designed, the platform actively supports testing by research groups, universities, sensor groups, and startups in smart agriculture. Thanks to open data and Docker-based architecture, ALSApp allows machine learning models to be tested and validated on real user information. It is possible for hardware manufacturers to connect new sensor solutions to the ALSApp system in a matter of hours, which greatly aids rapid prototyping and field use. As a result of its integration capabilities, ALSApp serves as a hub for joint research and innovation that speeds up the creation and use of modern farming technologies.

8.2 Revenue Model and Growth Strategy

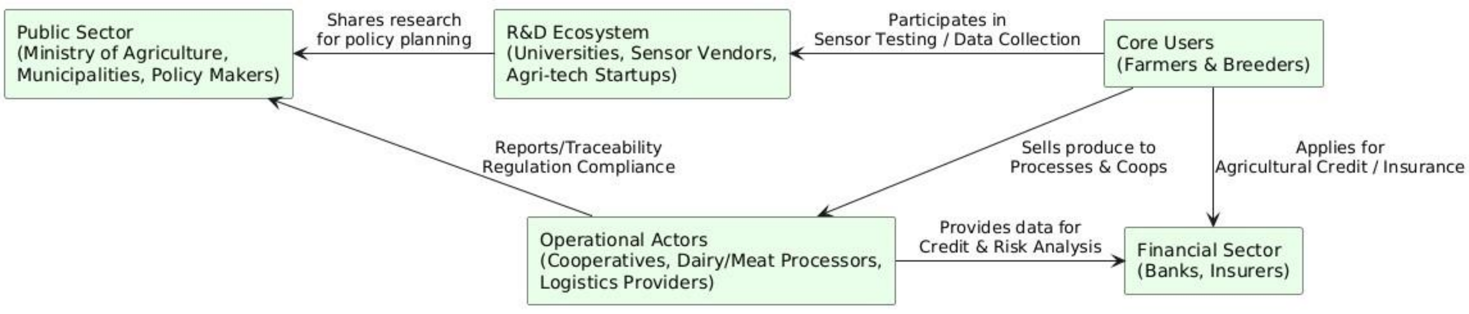
To transform ALSApp from a “working prototype” to a scalable product, a three-phase sustainability plan was developed. In phase 1 (0-12 months), the goal is to build a bug-free “core version” of the existing prototype, collecting real data from two pilot farms and around 50 closed-beta users. This version can be installed on the farmers' cooperatives' own servers or on low-budget cloud layers with a few commands thanks to Docker-based packages. In Phase 2 (12-24 months), the goal is to move the software to a “Platform as a Service (PaaS)” model and start generating subscription-based revenue, while sensor packages and online training modules will be offered as optional add-on packages in the form of hardware+service. Target in Phase 3 (24-36 months): When at least 2 000 enterprises in Turkey are reached, a regional dealership network and white-label licensing will be considered, so that the software can be distributed even under the brand name of local stakeholders. There are around 1.6 million farms in Turkey with herds of 1-50 heads and 1-10 hectares of land. It is estimated that only 10% of these producers are able to keep digital records; the remaining 90% market gap is considered as the primary target of ALSApp. The second target is the integrated dairy/meat plants and agricultural credit cooperatives that purchase in the target segment; when the quality and finance score based on data from the producer is provided through ALSApp, these institutions will meet their homogeneous data needs without the need for separate software.

The third link is the agricultural banking and insurance sector, where a steady flow of records will speed up risk scoring and claims verification, reducing operational costs. As small-scale producers in the Balkans, Central Asia and North Africa face similar digitization challenges abroad, the modular and low-bandwidth ALSApp model has direct export potential. On the hardware side, a “sensor package” model was adopted. Bluetooth weighbridges and soilmoisture probes are recognized in the application with ready-made drivers. When the manufacturer orders the package online and scans the QR code, the device is automatically paired. On the software side, a “multi-tenant” cloud architecture allows environment cloning for each new customer in seconds, enabling rapid growth without operational bottlenecks.

Competitive analysis Product Strength Weakness Position Relative to ALSApp Herdwatch (Ireland) Rich herd recording features, mobile AR integration Fully livestock focused, no field and support module ALSApp more holistic with field + support + offline support

AgriWebb (Australia) Detailed sensor integration, advanced analytics Annual license + sensor obligation, high price ALSApp, low entry barrier with gradual transition from "manual to sensor" Tarfin (TR) Input financing and marketplace No record keeping and early warning function ALSApp, e-Farm (TR) GDS-compliant desktop registration No cloud and mobile, data synchronization problematic ALSApp provides the advantage of real-time sharing with mobile-cloud architecture The common weakness of competitors is that they either focus only on the livestock or only on the farming side and do not integrate into formal support processes. The unique value of ALSApp is that it connects both lines of production to the same data schema, automatically summarizes government support announcements, and avoids rural network issues with offline buffering.

Moreover, thanks to open APIs, external sensor manufacturers or financial institutions can integrate within hours; this flexibility opens the door to a "localize-distribute" strategy in the international market. Pilot calculations show that even in sensor-less mode, a 12% reduction in water use and an 8% reduction in chemical fertilizers can be achieved with clever rule settings. If 100 000 businesses nationwide achieve this level of savings, it would mean a saving of 45 hm3 of water and 18 000 tons of fertilizer per year. This environmental impact is in full compliance with the EU Green Deal and "Zero Emission" targets, raising the possibility of accessing international grant and carbon credit funds. In short, with a clear sensor package plan for mass production, low-cost cloud scaling, a multi-tiered customer segment and internal and external market analysis, ALSApp has the potential to become a financially sustainable, environmentally positive and competitively unique solution. **8.4 Multi-Stakeholder Impact Map**


Public Sector (Ministry of Agriculture, Municipalities, Policy Makers)
Shares research for policy planning
R&D Ecosystem (Universities, Sensor Vendors, Agri-tech Startups)
Participates in Sensor Testing / Data Collection
Core Users (Farmers & Breeders)
Reports/Traceability Regulation Compliance
Sells produce to Processes & Coops
Applies for Agricultural Credit / Insurance
Operational Actors (Cooperatives, Dairy/Meat Processors, Logistics Providers)
Provides data for Credit & Risk Analysis
Financial Sector (Banks, Insurers)

9. Final Architecture and Design

9.1 System Architecture

The mobile application of ALSApp is built using Flutter so that it can run on both Android and iOS devices using just one codebase. The user interface on the mobile layer is made simple and with users in mind, helping small and medium-size farmers use the app with any level of technical skills. RESTful APIs are used to make all the requests from the frontend to the backend as well. Importantly, users can access and update the app, even if they are not connected to the internet, thanks to data being stored by packages such as Hive and shared_preferences. Because of this, users in rural areas can make use of the application and save their data, even without a constant connection to the internet.

Services in the backend use Node.js and a microservices approach to make the application flexible, scalable, and easy to work with. Modules are in place to take care of user and role management, crop and livestock details, sensor data processing, handling notifications, and tying in with government support bulletins. Using docker, the services are packaged in containers that support fast and automated updates and ensure scalability. Security relies on JWT for handling user session management and user rights. Using Docker, each backend service can be deployed the same way in all environments and orchestrated easily as it is containerized.

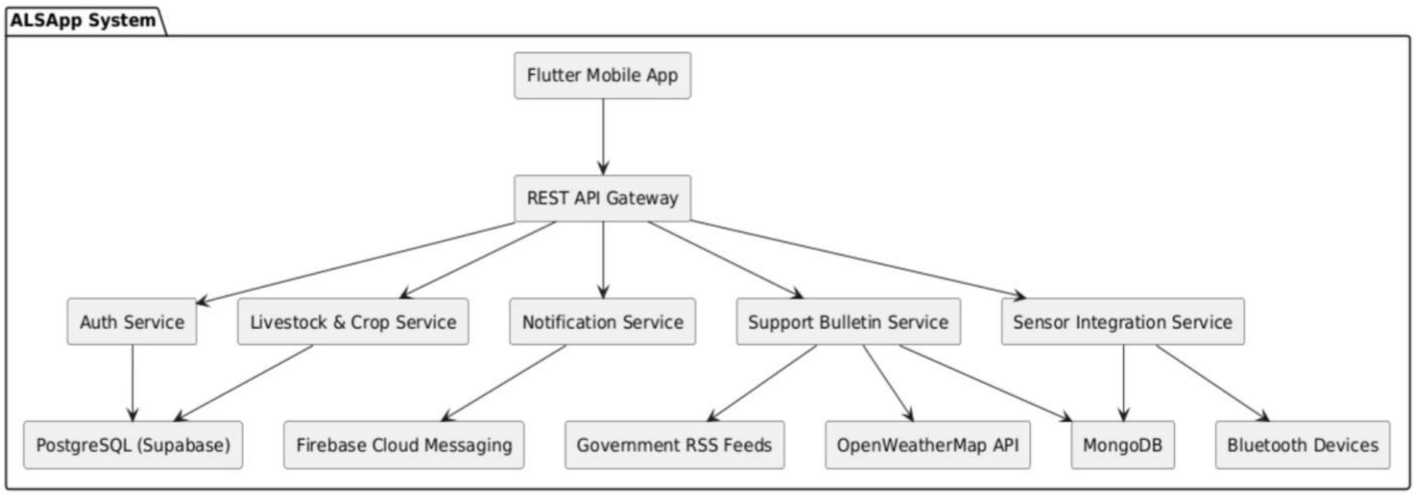
For ALSApp it has been chosen as a way to structure their database that prioritizes structure, can scale, and offers flexibility. For this type of data, the system makes use of PostgreSQL via Supabase for both structure and relational purposes. Furthermore, ALSApp includes MongoDB to keep a record of the collected data from sensor and environmental sources. This layer of the system ensures both easy access to fixed records and flexible gathering of realtime sensor data. The use of indexing strategies and checking the data schema helps to achieve good performance and safety with the database.

The system uses a number of cloud services and integrations to provide updated information and reliable service. The mobile application uses AWS Lambda and ECS for running and handling its scalable microservices. The integration of FCM in ALSApp quickly sends push notifications to mobile devices about upcoming weather and government news. The ALSApp uses the OpenWeatherMap API to gather environmental information that helps with warning others and deciding on suitable actions. The ALSApp uses a simple solution to ensure

Bluetooth sensors for livestock scales and the moisture of soil can connect and send data automatically, without much setup.

To finish, the ALSApp team uses GitHub Actions and Docker to automate every step of the CI/CD process. A CI/CD pipeline, backed by GitHub Actions and Docker, is used to ensure that updates are checked correctly and put into use easily. By using modern DevOps methods, the product's development is swifter and more reliable for supporting multiple integrations.

9.2 System Component Diagram



9.3 Key Design Features

With an offline-first design, ALSApp is dependable in areas where the internet is not consistently available. When offline, the app collects data from users and its sensors and keeps it ready, so that it can send it to the back-end server when a reliable connection is found. Once you can connect again, the data is automatically synchronized in the backend using retry methods. Thanks to this, the system is reliable and guarantees that data is not lost, even when there are challenges in the fields.

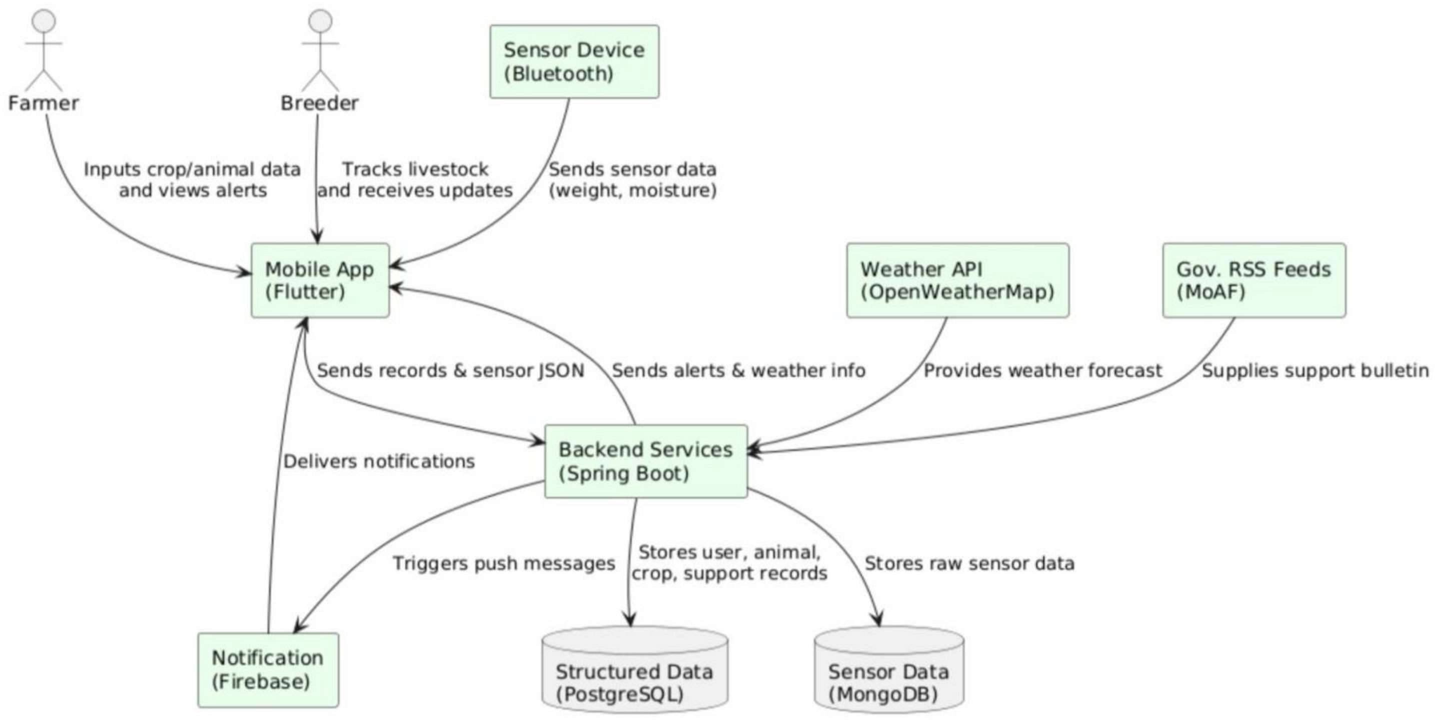
The system allows developers from different organizations to easily incorporate their services. It can be used with sensors, official systems, and platforms for research. Developers and organizations can use well-documented endpoints in ALSApp to seamlessly bring their tools into the platform and promote innovation in the agricultural sector.

Additionally, the ALSApp allows users to add new components or tools gradually as needed. If the farm lacks expensive technology, they can use manual processes first and then add hardware tools, analysis software, or official dashboards when their needs become greater. This makes it possible for small farms to transform digitally by adding sensors, analytics, or other important systems as they gain access to them.

Abnormal conditions, such as a sudden drop in soil moisture or a change in livestock weight needs to be identified, so ALSApp runs lightweight analytics on mobile devices to help recognize these changes and make prompt alerts and decisions, even when offline. In this way, Z-score-based calculations on the device detect any sudden changes and alert the system about, for example, changes in the weight of livestock or the moisture levels in the soil. As a result, users can be alerted and make decisions when changes are detected.

So, all core activities are designed in a RESTful approach, allowing any interface to connect and access them, without the need to rebuild the main structure. Because of this, future web dashboards, administrative sites, or even apps created by partners can be added to the system without modifying the main software. All these design decisions ensure that ALSApp is ready for the field, can grow in the future, and is still compatible with various technologies and applications.

9.4 Data Flow Diagram


Farmer
Breeder
Sensor Device (Bluetooth)
Inputs crop/animal data and views alerts
Tracks livestock and receives updates
Sends sensor data (weight, moisture)
Mobile App (Flutter)
Weather API (OpenWeatherMap)
Gov. RSS Feeds (MoAF)
Sends records & sensor JSON
Sends alerts & weather info
Provides weather forecast
Supplies support bulletin
Delivers notifications
Backend Services (Spring Boot)
Triggers push messages
Stores user, animal, crop, support records
Stores raw sensor data
Notification (Firebase)
Structured Data (PostgreSQL)
Sensor Data (MongoDB)

10. Final Status of ALSApp

Once completed, ALSApp is expected to produce measurable outputs and results in scientific, economic, environmental, commercial, social and technical fields. On the scientific front, animal-farm-air data to be collected from two pilot farms will be anonymized and published as open data. It is envisaged that this study will be the subject of scientific articles (early disease diagnosis, water-fertilizer optimization, support-effectiveness analysis) and a local visualization running on a mobile device. As an economic result, the early-warning cycle calibrated with manual and sensorized records is expected to result in a 3% increase in milk yield, 10% reduction in calf loss, 12% savings in irrigation water and 8% savings in chemical fertilizer. For a single family enterprise, this is estimated to equate to an annual net gain of approximately 25 000 TL. If the system is rolled out to 5 000 enterprises within three years, total savings are planned to reach 125 million TL, annual water savings of 45 hm3, fertilizer reduction of 18 000 tons and carbon emission cuts of about 40 000 tons of CO2. Thus, the application will prove to be climate friendly. The commercial potential of the application is estimated to be at least 4.5 million TL per year with a monthly subscription of 100 TL and optional sensor package sales/rentals.

The social and societal impact of the project is expected to increase the participation of women and youth in decision-making processes by 30% with the voice-activated interface, reduce the recall time from days to hours with the batch-ID supply chain, and shorten the update cycle of national statistics by 50% with integration to the ministry's e-agriculture portal. In addition, the application produced by the project will make a significant contribution by increasing digitalization in agriculture, resulting in the use of effective techniques in agriculture. In addition, the originality and localization of the application is also an important result. Thus, the project will provide an impact set that is directly parallel to the objectives of yield increase, input savings, sustainability and digital inclusion, and traceable in numbers. Looking at the conditions required for the development of the project, first of all, reliable infrastructure and hardware support should be provided. In rural areas, instant transmission of data from sensors, GSM/LoRaWAN networks with adequate coverage and redundant internet connection are important requirements. In addition, a stock of spare sensors and maintenance equipment needs to be available for long-term field tests. Another important element for the future of the project is to increase the digital literacy of farmers.

10.1 Technical Readiness and Implementation Status

The technical foundation of ALSApp has been built so it can support more users in the future, work well with other parts as they're added, and handle tough conditions when it's used in remote areas. The current version of the mobile app works offline by using Hive and shared_preferences for storing data. The current version of the platform is a simple and easyto-use starting point, with important features already working from the frontend to the backend and sensors.

The mobile app, made in Flutter, lets farmers and breeders easily enter their data even when there's no internet, shows real-time alerts, and keeps farmers up to date with government support news, all while saving their information offline using Hive and shared_preferences, and automatically sending their data when a connection is back. It helps farmers and ranchers record animal and crop information, see updates right away about their alerts, and find out about any government help related to their farms, all on their phones even if there's no internet connection at the time. The offline capability uses local data storage with Hive and shared_preferences, making sure that any data entered by the farmer or breeder on the app gets sent when there is an internet connection again. The frontend is made to be easy to use, with accessibility features and buttons that are easy to touch, designed to help people in rural areas who might not use phones too much.

The backend is divided into small services built in Node.js. ECS from AWS Elastic Container Service is used to containerize the services and manage their deployment, allowing for easy maintenance and greater elasticity. The key services are handling user authentications and authorizations with JWT, looking after livestock and crop data, dealing with weather and government bulletins, managing notifications, and working with sensor inputs. To remain flexible, the system allows external applications to communicate with each other via RESTful APIs.

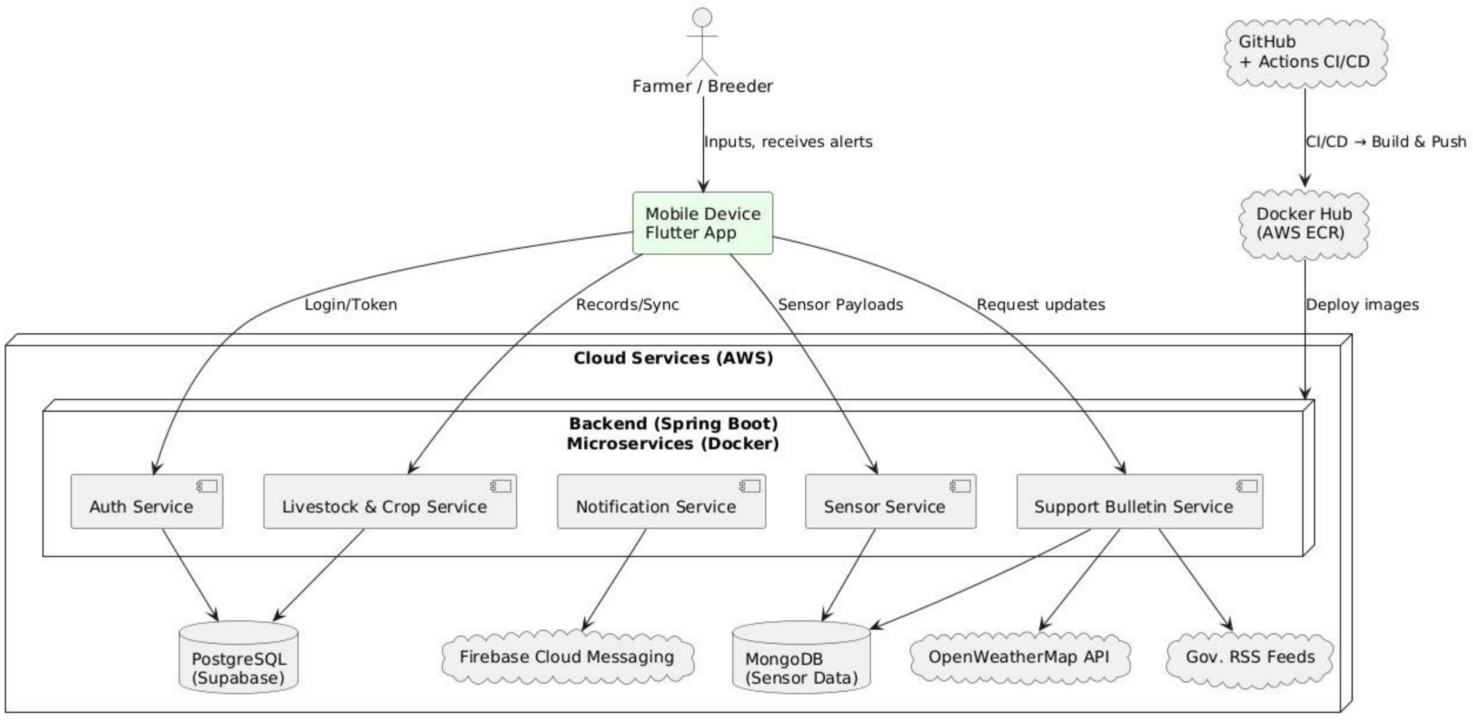
A combination of PostgreSQL (for relational data) and MongoDB (for semi-structured data) has been used to ensure the system can adapt to more sensor data. The logs, user records, and filled in forms are all kept in PostgreSQL via Supabase, whereas MongoDB deals with the semi-structured sensor data and factory output. Using both PostgreSQL (for relational data) and MongoDB (for semi-structured data) allows the system to grow with the number of sensors and still maintain core workflow connections. Querying is made fast and reliable by using indexing and validation to ensure no schema violations happen.

It makes use of various external APIs and services, for example, it retrieves weather data from OpenWeatherMap to help with instant alerts. OpenWeatherMap is used to fetch real-time weather information that is then processed for creating alerts in local languages. RSS feeds are used to fetch updates from the authorities for users to read more easily. Firebase Cloud Messaging is the tool used for sending push notifications when alert limits or application deadlines are crossed.

The platform has also managed to connect solar panels to the system through Bluetooth-based livestock scales and soil moisture sensors. By scanning a QR code, these devices communicate with the mobile app and hand over already-calibrated, time-stamped, and validated data. A module that uses Z-score outlier detection runs on the device, which allows for alerts to be sent even if the app is disconnected from the backend.

All code is tested, images generated, and code deployed to AWS through the use of GitHub Actions. This makes it possible to update the system extremely fast during field testing, thanks to input from users. From a security standpoint, the app uses HTTPS, TLS 1.2+ encryption for data privacy and authenticates using short-lived, signed JWTs. The backend is set up to comply with KVKK and GDPR rules, for example, by allowing users to control sharing, making their data anonymous, and letting them request that their information be deleted.

10.2 Deployment and Integration Architecture Diagram



11. Conclusion

The ALSApp project has aimed to explore how digital technologies can help smaller farms deal with things like limited internet access, problems with data recording, and not having enough connection to support systems, all made possible by a simple and easy to use system. Throughout the development process, the team tried to build a system that works well offline, has sensors to help with tasks, keeps track of government support, and sends early alerts, all so it could actually help smaller farms who face internet problems, data collection issues, and need better connections with government help.

Rather than claiming anything for sure, this project has tried to show that using digital tools for farming can work well, even for smaller farms dealing with issues like bad internet. By adding things like mobile apps farmers can use offline, sensors, ways to track support from the government, and some warnings, the ALSApp wants to be a helpful tool for both farmers and people who make the rules for farming. By adding things like being able to use the app when there is no internet, connecting with farm machines, tracking what government support is available, and giving early warnings, ALSApp hopes to be helpful for both farmers and people making rules.

According to engineering requirements, the main architectural and implementation goals are now complete. The team tested the application in a real pilot and moved it to the prototype stage. Feedback is still being used to enhance the solution, mainly focusing on making the system simpler, better running, and flexible.Even though the ALSApp initiative is not fully resolved, it provides a good starting point for further work. The work gave the developers useful lessons on how to design, build, and use such an app while dealing with real-world challenges. Furthermore, the next phase requires field testing, input from stakeholders, and help from experts to further refine the solution and analyze its effects.

To sum up, ALSApp aims to blend engineering concepts with efforts that benefit society. The project seeks to help students stay engaged with important topics in sustainable, food, and rural development by contributing, however little, to these discussions.

12. Appendix A: UI Mockups or Screenshots

12.1 UI Screenshots

For an easier understanding, screenshots will be used to explain the user interface in this section. The pictures highlight the most important screens, pointing out how users interact with everything in the app.Readers can easily learn about the application’s design and organization by moving through the visuals one step at a time.

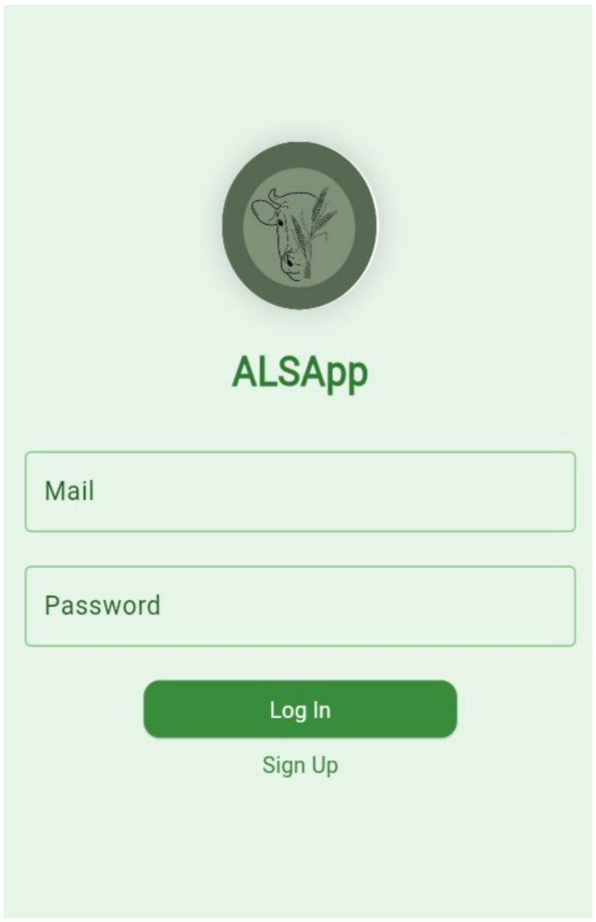


Figure 1: *The screen displays the ALSApp logo and name above two fields for entering your mail and password. Just under those fields is a green “Log In” button and a “Sign Up” link.*

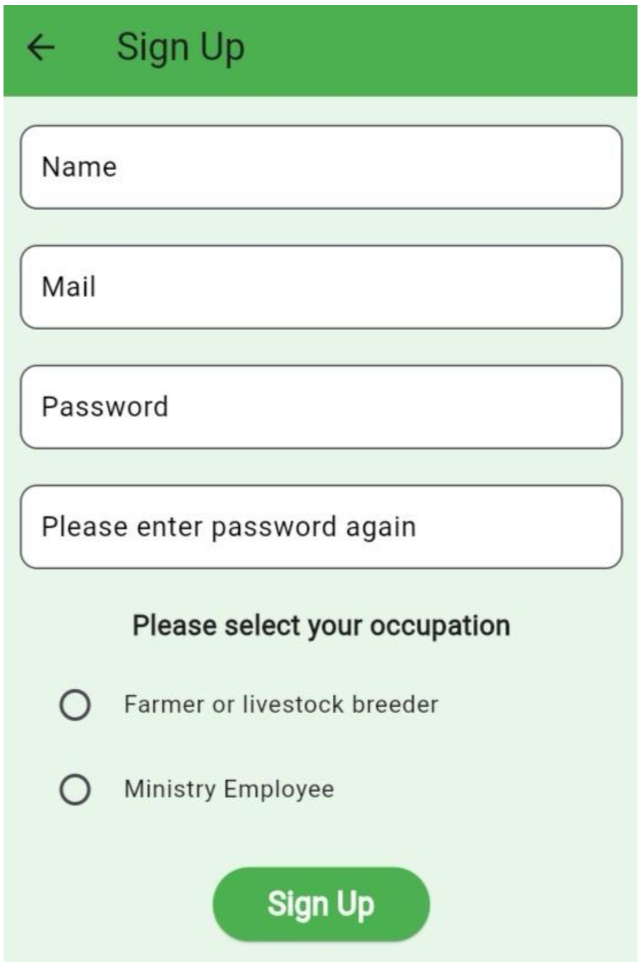
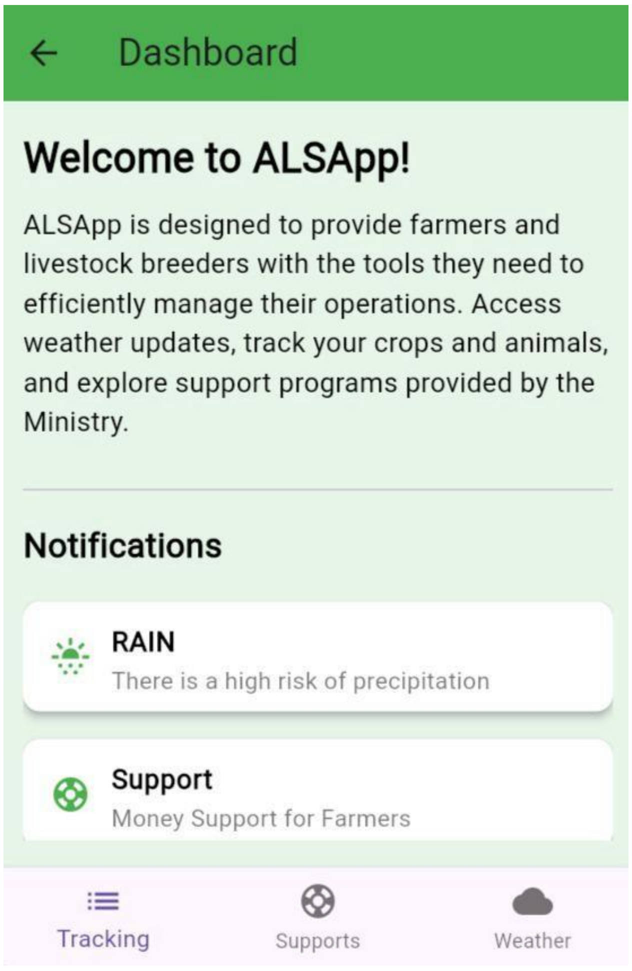

Sign Up
Name
Mail
Password
Please enter password again
Please select your occupation
Farmer or livestock breeder
Ministry Employee
Sign Up

Figure 2: *At the top of the screen, you find the names of the fields "Name, Mail, Password and Confirm Password" in order for people to enter their information. The last part of the form has two radio buttons for users to select their job (Farmer or Ministry Employee) and a bold green "Sign Up" button allows them to create their account.*



Dashboard

Welcome to ALSApp!

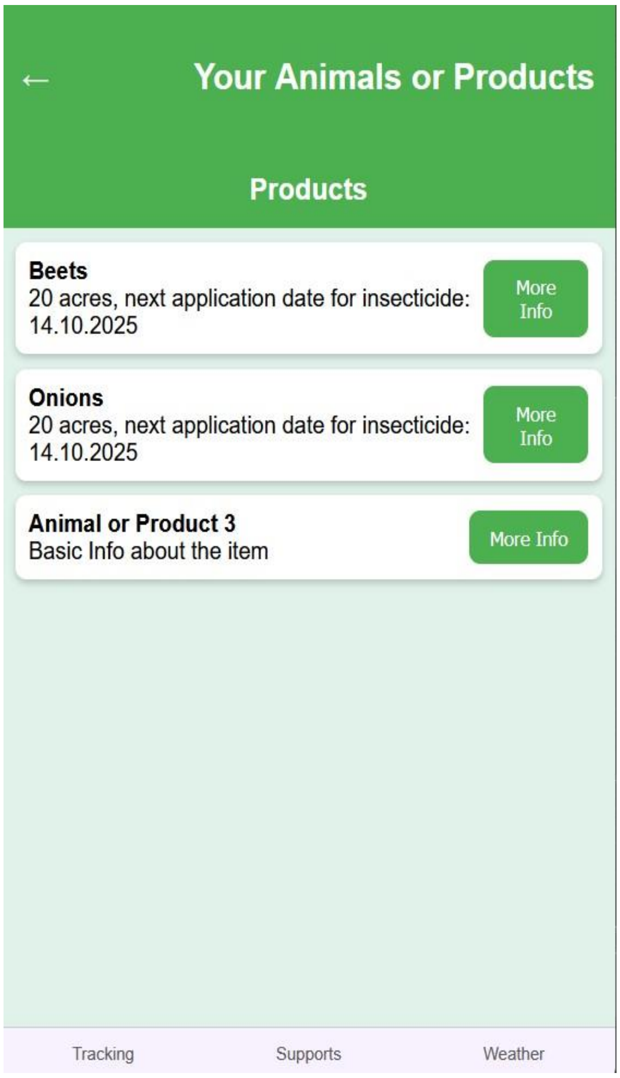
ALSApp is designed to provide farmers and livestock breeders with the tools they need to efficiently manage their operations. Access weather updates, track your crops and animals, and explore support programs provided by the Ministry.

Notifications

RAIN
There is a high risk of precipitation

Support
Money Support for Farmers

Figure 3: *A welcome and basic overview of what ALSApp offers to farmers and livestock breeders is shown once the dashboard loads. You'll find both rain risk alerts and programs for support, along with a bottom bar to select between the Tracking, Supports and Weather options.*

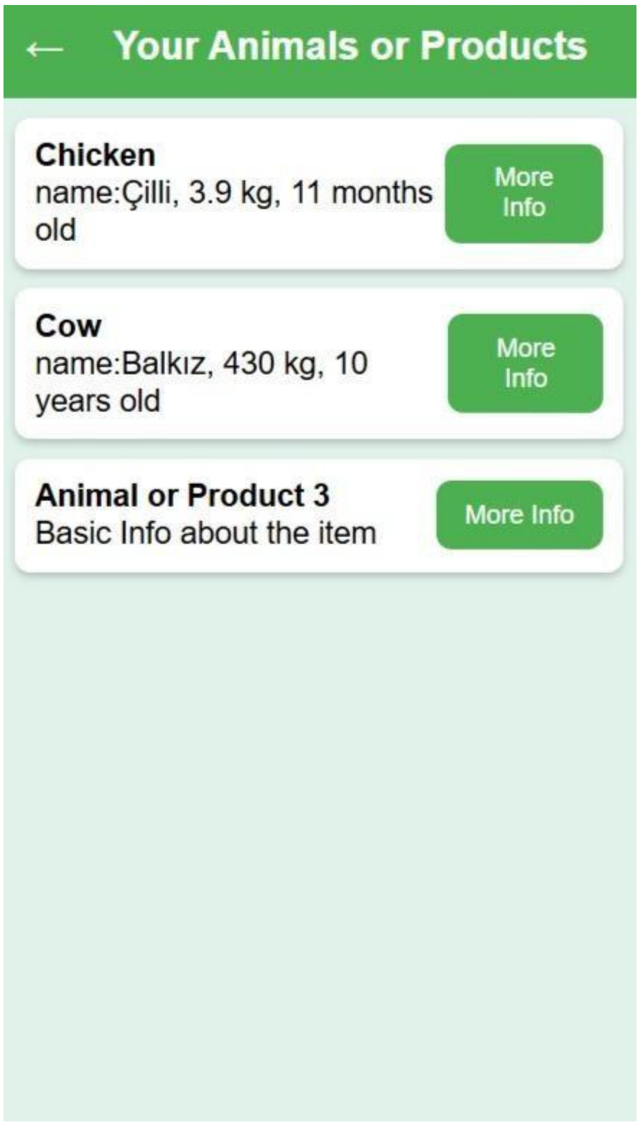


Beets
20 acres, next application date for insecticide: 14.10.2025

Onions
20 acres, next application date for insecticide: 14.10.2025

Animal or Product 3
Basic Info about the item

Figure 4: *Each item in the "Your Animals or Products" section shows basic details (e.g., how much land and the next recommended insecticide spray) with a green "More Info" button beside. Up in the green AppBar, you'll find the back arrow and the title and below that, a navigation bar helps you go from Tracking to Supports to Weather.*

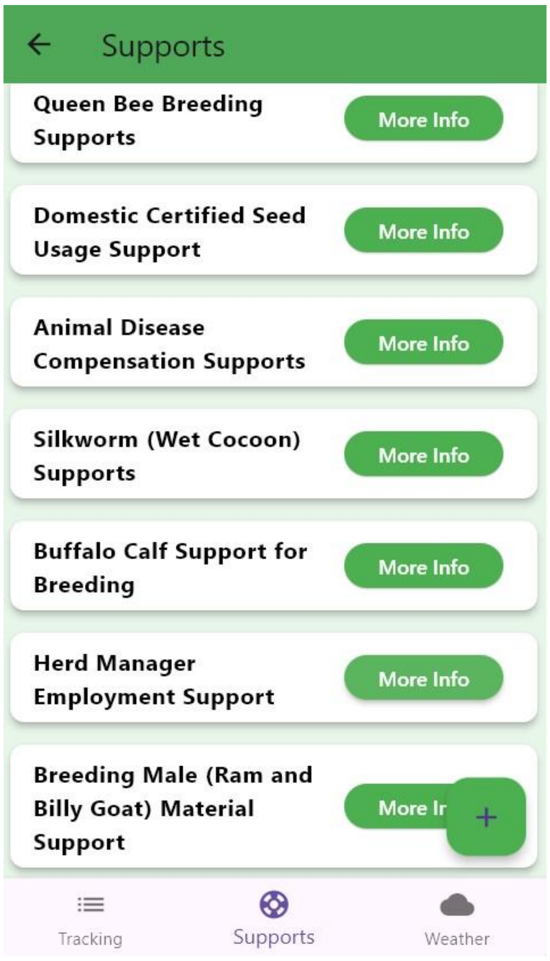


Chicken
name:Çilli, 3.9 kg, 11 months old

Cow
name:Balkız, 430 kg, 10 years old

Animal or Product 3
Basic Info about the item

Figure 5: *You can see all your animals or products on the screen, with each item's name, weight and age. A "More Info" button is found on every card and you can press the AppBar's back button to go back to the previous place.*


Supports

Queen Bee Breeding Supports

Domestic Certified Seed Usage Support

Animal Disease Compensation Supports

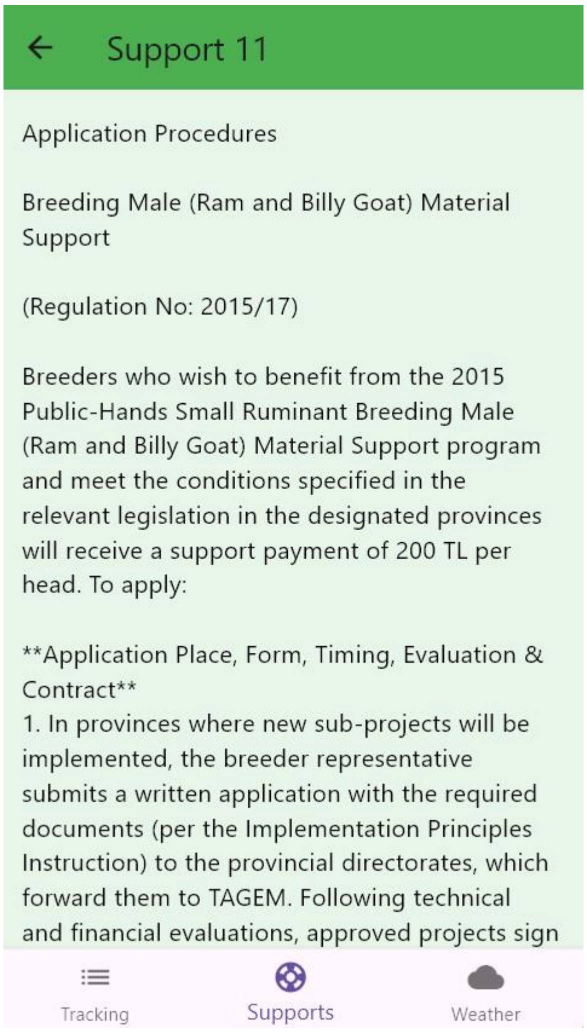
Silkworm (Wet Cocoon) Supports

Buffalo Calf Support for Breeding

Herd Manager Employment Support

Breeding Male (Ram and Billy Goat) Material Support

Figure 6: *It can be observed that each program listed as a card on the Supports page and clicking the green "More Info" button lets you find out extra details. A floating "+" button in the bottom corner lets users flag supports, while the bottom navigation bar (Tracking, Supports, Weather) highlights the current tab.*


Support 11

Application Procedures

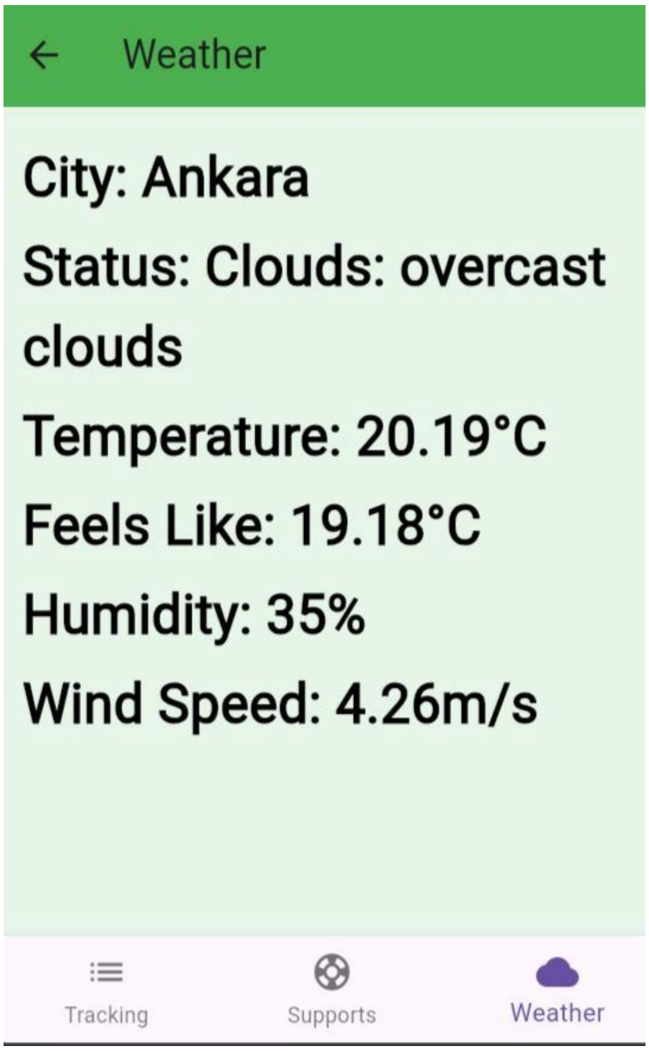
Breeding Male (Ram and Billy Goat) Material Support

(Regulation No: 2015/17)

Breeders who wish to benefit from the 2015 Public-Hands Small Ruminant Breeding Male (Ram and Billy Goat) Material Support program and meet the conditions specified in the relevant legislation in the designated provinces will receive a support payment of 200 TL per head. To apply:

Application Place, Form, Timing, Evaluation & Contract
1. In provinces where new sub-projects will be implemented, the breeder representative submits a written application with the required documents (per the Implementation Principles Instruction) to the provincial directorates, which forward them to TAGEM. Following technical and financial evaluations, approved projects sign

Figure 7: *The detail page has a green label for Support 11 with a back button and underneath, you'll find the full text of the English application procedures for Breeding Male (Ram and Billy Goat).*



City: Ankara

Status: Clouds: overcast clouds

Temperature: 20.19°C

Feels Like: 19.18°C

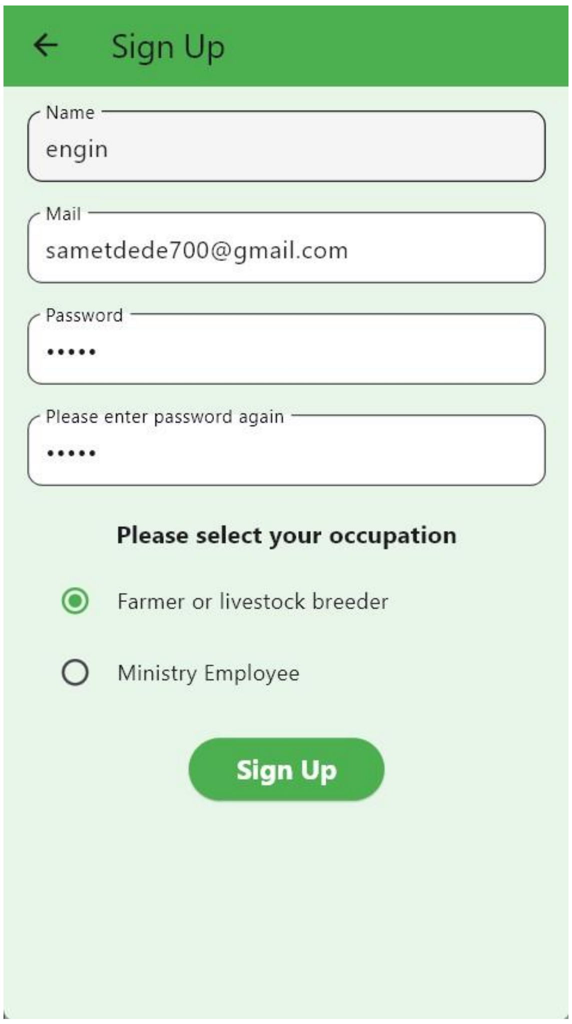
Humidity: 35%

Wind Speed: 4.26m/s

Figure 8: *Here you'll find the weather in Ankara clearly displayed with cloud cover, the actual temperature, "feels like" temp, humidity and wind amount. Swipe from the left to see your most recent searches and call up the Weather page through the bottom nav bar.*

12.2 Data Input Examples

In this part the data inputs were explained in detail using the UI screenshots below for better understanding:



Sign Up

Name: engin

Mail: sametdede700@gmail.com

Password: •••••

Please enter password again: •••••

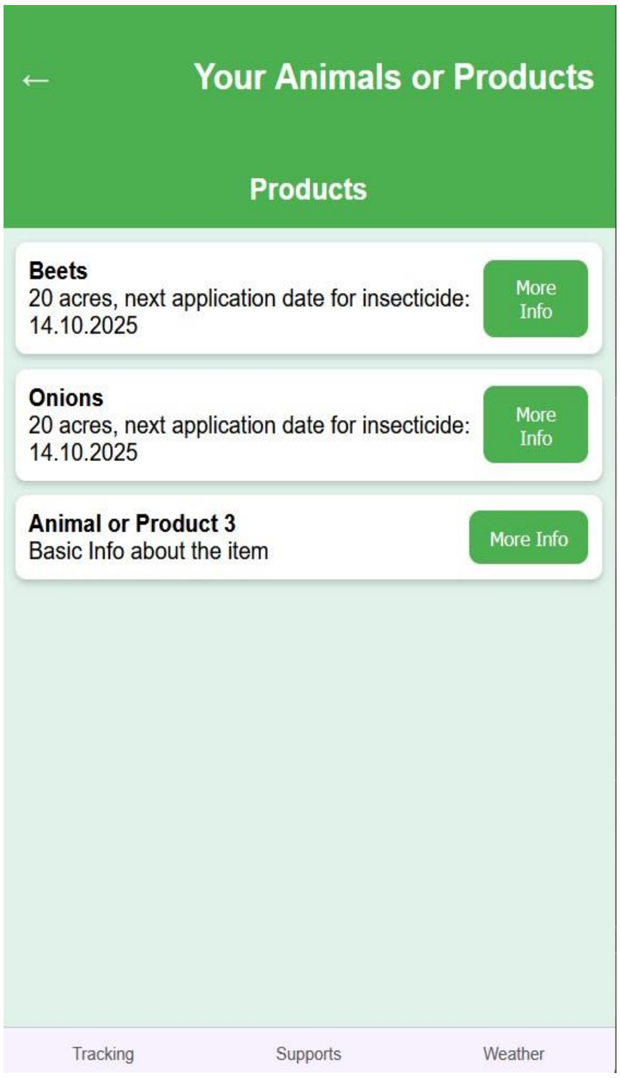
Please select your occupation

[✓ Farmer or livestock breeder]

Ministry Employee

Sign Up

Figure 9: *Here in this user interface user is asked to save their datas such as their name, e-mail etc. to sign up succesfully*

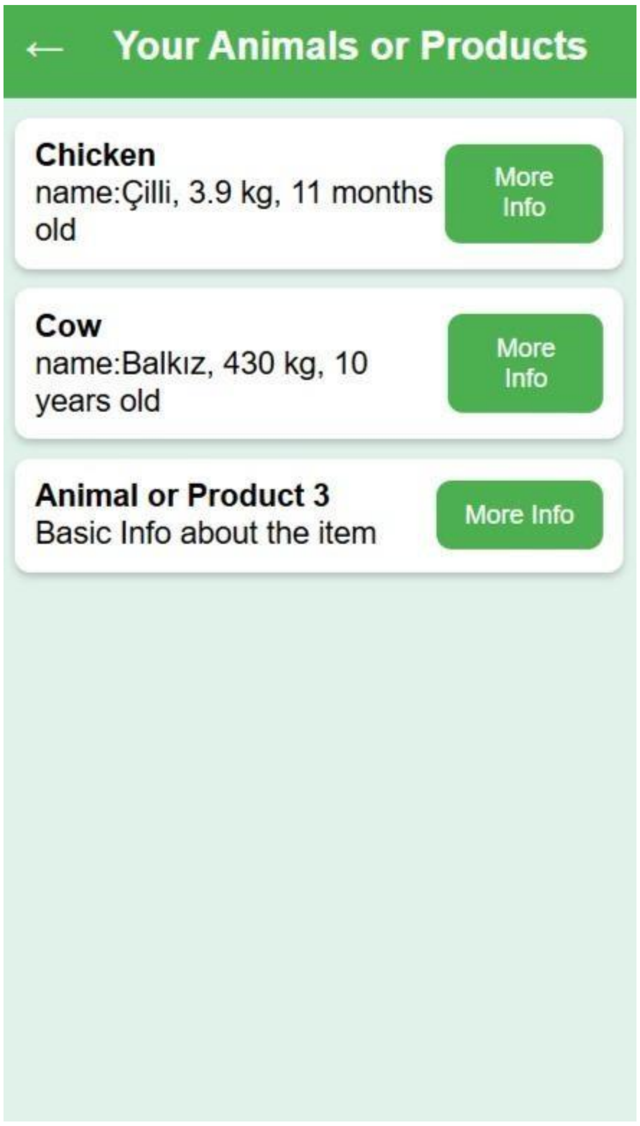


Beets
20 acres, next application date for insecticide: 14.10.2025

Onions
20 acres, next application date for insecticide: 14.10.2025

Animal or Product 3
Basic Info about the item

Figure 10: *This user interface is utilized for saving the products data for continuing the tracking. User will click on the more info button to see more informations about their products and add informations for their products.*



Chicken
name:Çilli, 3.9 kg, 11 months old

Cow
name:Balkız, 430 kg, 10 years old

Animal or Product 3
Basic Info about the item

Figure 10: *This user interface is utilized for saving the animals data for continuing the tracking. User will click on the more info button to see more informations about their animals and add informations for their animals.*

12.3 Notification Triggers & Alerts

In this part the Notification Triggers & Alerts were explained in detail using the UI screenshots below for better understanding:

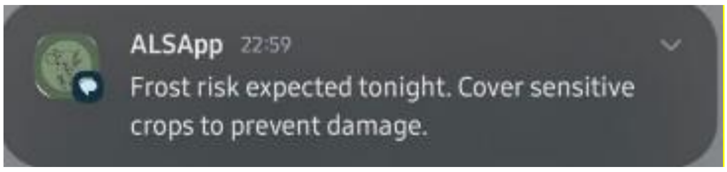
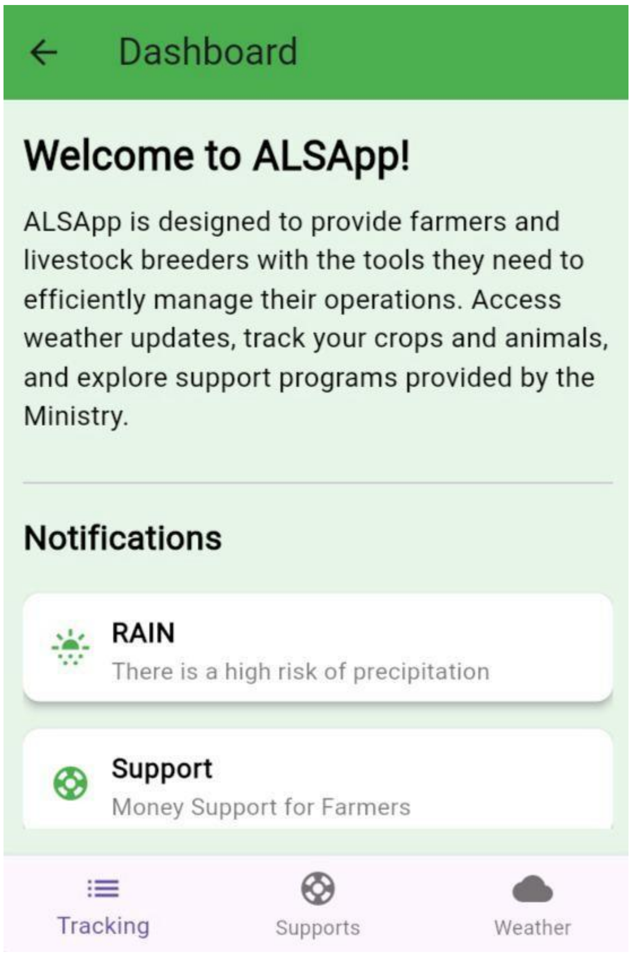

ALSApp 22:59
Frost risk expected tonight. Cover sensitive crops to prevent damage.

Figure 11: *This is a push notification from ALSApp showing the app icon, name, and timestamp (22:59) at the top. The message warns of a frost risk tonight and advises covering sensitive crops to prevent damage. The farmer will be notified by such notifications above*


Dashboard

Welcome to ALSApp!

ALSApp is designed to provide farmers and livestock breeders with the tools they need to efficiently manage their operations. Access weather updates, track your crops and animals, and explore support programs provided by the Ministry.

Notifications

RAIN
There is a high risk of precipitation

Support
Money Support for Farmers

Figure 12: Here in this dashboard user interface the notifications that will be sent will also appear to track the important notifications.

13. Pseudocodes

Pseudocodes of some of the crucial parts of the application are as follows.

13.1 Node.js Pseudocodes

13.1.1 Dash.js pseudocode

It Handles secure access to the dashboard page. It's triggered when the user clicks the "openDashboardButton".

```
When the page is fully loaded:

- If "openDashboardButton" is clicked:

    → Prepare authentication data:

- id

- password

    → Send POST request to /dash

    → If the response is successful:

→ Display it in the "res" element

    → If there's an error:

        → Display the error message in the "res" element
```

13.1.2 Index.js pseudocode

It tests the backend connection or confirming server availability.

```
When the page has fully loaded:

  → Prepare test data with value "client"

  → Send a POST request to /testpost
```

```
    → If the request succeeds ,display the response in the "res" element
```

13.1.3 Login.js pseudocode

It handles user login with email and password credentials.

```
When the page is fully loaded:
    - When "loginButton" is clicked:
        → Prepare login credentials (email, password)
        → Send POST request to /login endpoint
→ If the response is successful:
            → Show response in the "res" element
```

13.1.4 Product.js pseudocode

```
When the page is fully loaded:
    - When "getProductsButton" is clicked:
        → Prepare user credentials (id, password)
→ Send POST request to /products endpoint
→ If response is successful or an error:
            → Display the response in the "res" element
```

13.1.5 signUp.js pseudocode

It handles new user account creation.

```
When the page is fully loaded:
```

```
- When "signupButton" is clicked:
    → Prepare signup data (email, name, password, role)
    → Send POST request to /signup endpoint
    → If response is successful or an error:
        → Display the response in the "res" element
```

13.1.6 Weather.js pseudocode

It fetches weather data based on the user's current location using geolocation.

```
When user clicks "Get Weather" button:
    → Use browser geolocation to get latitude and longitude
    → If location is retrieved successfully:
        → Send POST request to backend endpoint /getWeather with latitude and longitude
        → When server responds:
            → Display weather info in the UI
→ If location access fails:
        → Show error message in the UI
```

13.2 Flutter Pseudocodes

13.2.1 DashboardScreen pseudocode

It handles secure access to the dashboard page. It's triggered when the user taps the "openDashboardButton."

```
When the screen is fully loaded:
    • If openDashboardButton is tapped:
        → Prepare authentication data:
```

```
- id
- password
→ Send POST request to /dash with auth data
→ If the response is successful: → Display
response body in the res element
→ If there's an error:
→ Display error message in the res element
```

13.2.2 RegisterScreen pseudocode It handles new user account registration.
When the screen is fully loaded:

```
• If registerButton is tapped:
    → Prepare registration data:
    - name
    - email
    - password
    - confirmPassword
    → Send POST request to /register with registration data
    → If the response is successful:
    → Display "Registration successful" in res element
    → If there's an error:
    → Display error message in res element
```

13.2.3 AddProductScreen pseudocode It handles adding a new product.
When the screen is fully loaded:

```
• If addProductButton is tapped:
    → Prepare product data:
    - title
    - description
```

```
        - price
        - image (optional)
        → Send POST request to /product/add with product data
        → If the response is successful: →
        Display "Product added" in res element
        → If there's an error:
        → Display error message in res element
```

13.2.4 ProductListScreen pseudocode

It displays a list of products and lets the user pick one.

```
When the screen is fully loaded: →
Send GET request to /products
→ If the response is successful: →
Display product list in res element
→ If there's an error:
→ Display error message in res element

    • If a product item is tapped:
        → Navigate to ProductDetailScreen, passing productId
```

13.2.5 ProductDetailScreen pseudocode

It shows details for a selected product.

```
When the screen is fully loaded (with productId parameter):
→ Send GET request to /product/{productId}
→ If the response is successful: →
Display product details in res element
→ If there's an error:
→ Display error message in res element
```

13.2.6 ProfileScreen pseudocode

It manages viewing and updating the user's profile.

```
When the screen is fully loaded:
→ Send GET request to /profile (with user credentials)
→ If the response is successful: →
Display profile data in res element
→ If there's an error:
→ Display error message in res element

    • If updateProfileButton is tapped:
        → Prepare updated profile data
        → Send POST request to /profile/update with updated data
        → If the response is successful: → Display
        "Profile updated" in res element
        → If there's an error:
        → Display error message in res element
```

13.2.7 SupportScreen pseudocode

It allows the user to submit a support request.

```
When the screen is fully loaded:

    • If sendSupportButton is tapped:
        → Prepare support data:
        - userId
        - message
        → Send POST request to /support with support data
        → If the response is successful: →
        Display "Request sent" in res element
        → If there's an error:
        → Display error message in res element
```

13.2.8 WeatherScreen pseudocode

It fetches weather information based on the user's location. When the screen is fully loaded:

```
• If getWeatherButton is tapped:
  → Request device geolocation (latitude, longitude)
  → If location retrieval succeeds:
  → Send POST request to /getWeather with latitude & longitude
  → If the response is successful: →
  Display weather info in res element
  → If there's an error:
  → Display error message in res element
  → If location access fails:
  → Display "Location access failed" in res element
```

13.2.9 ApiService pseudocode

It centralizes all HTTP communication with the backend.
When the service is initialized:

```
• Store baseUrl

When postRequest(endpoint, data) is called:
→ Construct url = baseUrl + endpoint →
Send POST request to url with data
→ If response status is success:
→ Return response body →
Else:
→ Return or throw error message

When getRequest(endpoint) is called:
→ Construct url = baseUrl + endpoint
→ Send GET request to url
→ If response status is success:
```

→ Return response body

→ Else:

→ Return or throw error message

13.2.10 SupportService pseudocode

It wraps support-related API calls using ApiService.

When the service is initialized:

- Inject or create apiService

When sendSupportRequest(userId, message) is called:

→ Prepare payload:

- userId

- message

→ Call apiService.postRequest('/support', payload)

→ If call succeeds: →

Return success response →

Else:

→ Return or throw error

13.2.11 WeatherService pseudocode

It encapsulates weather retrieval logic, combining geolocation and API calls.

When fetchWeather() is called:

→ Request device geolocation (latitude, longitude) →

If location retrieval succeeds:

→ Prepare payload:

- lat = latitude

```
- lon = longitude
→ Call apiService.postRequest('/getWeather', payload)
→ If call succeeds: →
Return weather data →
Else:
→ Return or throw error
→ If location access fails:
→ Return "Location access failed" error
```

13.2.12 BottomNavigation pseudocode

It manages tab-based navigation across the main screens.
When the widget is initialized:

- Define pages list (e.g., DashboardScreen, ProductListScreen, ProfileScreen, SupportScreen, WeatherScreen)
- Define corresponding navItems (icons and labels)
- Initialize selectedIndex = 0

```
If a bottom nav item at index is tapped:
→ Update selectedIndex = index
→ Trigger UI rebuild

When building the UI:
→ Scaffold:
- body = pages[selectedIndex]
- bottomNavigationBar = BottomNavigationBar( items = navItems,
  currentIndex = selectedIndex, onTap = (index) → handle tap as above
)
```

14. Glossary

ALSApp: Agriculture and Livestock Support Application; a digital platform designed to assist small-scale farmers by providing tools for record keeping, early warning alerts, government support tracking, and sensor integration.

API (Application Programming Interface): A set of protocols and definitions that enable software systems to communicate with each other. ALSApp uses REST APIs for frontendbackend interaction and third-party integrations.

CI/CD (Continuous Integration / Continuous Deployment): A development practice that automates testing, building, and deployment of code to ensure reliable and fast delivery of software updates. ALSApp uses GitHub Actions for CI/CD.

Docker: A containerization platform that packages applications and their dependencies into standardized units for consistent deployment. ALSApp's backend services are deployed using Docker.

ECS (Elastic Container Service): A container orchestration service offered by Amazon Web Services (AWS), used to manage and scale Docker-based microservices in ALSApp.

Flutter: An open-source UI framework developed by Google, used to build ALSApp's crossplatform mobile application for both Android and iOS.

Firebase Cloud Messaging (FCM): A cloud-based messaging solution that allows the ALSApp backend to send push notifications (e.g., alerts and reminders) to mobile users.

GitHub Actions: A DevOps tool integrated with GitHub that automates tasks such as testing, building, and deploying ALSApp's codebase.

GPS / QR Code Pairing: A method used to identify and link sensor hardware (like Bluetooth scales or moisture probes) with user devices using scannable QR codes.

JWT (JSON Web Token): A secure, compact format for transmitting user identity and permissions between frontend and backend services in ALSApp, used for session control.

KVKK / GDPR: Turkish and European data protection regulations that define how user data must be collected, stored, and shared. ALSApp follows these standards to ensure user privacy.

LoRaWAN (Long Range Wide Area Network): A low-power communication technology used in rural areas for transmitting sensor data. ALSApp can optionally support LoRaWAN-based sensors.

Microservices: An architectural style that organizes backend functionality into small, independent services (e.g., auth, sensor handling, notification), each deployed and scaled separately.

MongoDB: A NoSQL document-based database used in ALSApp to store semi-structured sensor data, such as soil moisture and livestock weight records.

MVP (Minimum Viable Product): A simplified but functional version of an application built to validate core features and gather user feedback. ALSApp's MVP includes manual data entry, alerting, and sensor integration.

NoSQL: A non-relational database model used for flexible storage of sensor and unstructured data. ALSApp uses MongoDB as its NoSQL layer.

Offline-First: A design principle where the mobile application operates independently of internet connection, storing data locally and syncing it when a connection becomes available.

OpenWeatherMap API: A third-party service used in ALSApp to retrieve real-time weather data and forecasts for alert generation.

PostgreSQL: A robust open-source relational database used in ALSApp (via Supabase) for managing structured data such as users, forms, and logs.

REST API (Representational State Transfer API): A type of web service that uses standard HTTP methods to interact with backend resources. ALSApp's frontend communicates with its backend using RESTful APIs.

RSS Feed (Rich Site Summary): A format used to publish frequently updated content. ALSApp parses RSS feeds from government sources to display relevant agricultural support announcements.

Node.js: Runtime environment that lets you run JavaScript code on the server side, outside of the browser, enabling backend development using JavaScript.

Supabase: An open-source backend-as-a-service that provides PostgreSQL hosting, user authentication, and storage features. ALSApp uses it for structured data and account management.

TRL (Technology Readiness Level): A scale from 1 to 9 that describes the maturity of a technology. ALSApp currently operates at TRL-5, meaning it is validated in a relevant environment.

Z-score: A statistical measurement used in ALSApp to detect anomalies in sensor data. It compares current values to historical averages to identify potential early warnings.

15. References

[1] Cumhurbaşkanlığı Strateji ve Bütçe Başkanlığı, On İkinci Kalkınma Planı, 2024-2028. Strateji ve Bütçe Başkanlığı, 2023. [Online]. Available: https://www.sbb.gov.tr

[2] Devlet Su İşleri Genel Müdürlüğü, Türkiye'nin Su Verileri 2023. T.C. Tarım ve Orman Bakanlığı, 2023.

[3] Devlet Su İşleri Genel Müdürlüğü, Sulama Tesisleri Performans Değerlendirmesi (Tablo 2). T.C. Tarım ve Orman Bakanlığı, 2022.

[4] Meteoroloji Genel Müdürlüğü, Meteoroloji Veri Servisi. 2024. [Online]. Available: https://mgm.gov.tr

[5] Tarım Dünyası Dergisi, Türkiye'de Çiftçi Profili ve Dijitalleşme Raporu. Tarım Dünyası Yayıncılık, 2023.

[6] "Tarım ve Hayvancılık Destek" Web Sitesi. [Online]. Available: https://tarimhayvancilikdestek.com.tr

[7] Tarım ve Orman Bakanlığı, Tarım ve Orman Bakanlığı Stratejik Planı, 2024-2028. 2023. [Online]. Available: https://www.tarimorman.gov.tr

[8] Tarım ve Orman Bakanlığı, Hayvancılık İstatistikleri 2023: Buzağı Kayıpları ve Ölüm Oranları Değerlendirme Raporu. T.C. Tarım ve Orman Bakanlığı, 2024.

[9] Türkiye İstatistik Kurumu, Hasat ve Hasat Sonrası Kayıplar, 2022 (TÜİK Temel İstatistikler Serisi No. 28610). 2023. [Online]. Available: https://data.tuik.gov.tr

[10] OpenWeatherMap, OpenWeatherMap API. [Online]. Available: https://openweathermap.org/api

[11] E. S. Dede, S. D. Demirci, and G. Ünsal, "ALSApp: Agriculture and Livestock Support Application - Analysis Report," TED University, Department of Computer Engineering, 2024.

[12] E. S. Dede, S. D. Demirci, and G. Ünsal, "ALSApp: Agriculture and Livestock Support

Application - High Level Design Report," TED University, Department of Computer Engineering, 2024.

[13] Association for Computing Machinery (ACM), "ACM Code of Ethics and Professional Conduct," 2018. [Online]. Available: https://www.acm.org/code-of-ethics

[14] G. Gungor and M. Dogan, "A Sentiment Analysis System," Boğaziçi University Undergraduate Project, 2019. [Online]. Available:

https://www.cmpe.boun.edu.tr/~gungort/undergraduateprojects/A%20Sentiment%20Analysis%20System.pdf