CS492 Senior Design Project - Spring 2025

Detailed Design Report

YOLLA-T2418

Celal Salih Türkmen - 22102498

Sadra Mohammadzadehazarabadi - 22101018

Eslim Rana Emiroğlu - 22102692

Elif Şen - 22102144

Halil Tataroğlu - 22102198

Supervisor: Can Alkan

1. Introduction

1.1 Purpose of the System

Yolla is a comprehensive mobile and web-based cargo marketplace designed to revolutionize the logistics industry in Türkiye. It aims to provide a seamless experience for individuals and SMEs seeking efficient, cost-effective, and transparent cargo solutions. Yolla enables users to compare multiple service providers based on delivery time, pricing, and additional features, ensuring informed decision-making. By integrating advanced technologies such as real-time tracking, AI-powered chat support, and automated package dimension measurement, Yolla enhances user convenience and operational efficiency. Furthermore, it supports sustainability by offering eco-friendly delivery rankings and empowering local businesses to compete in the logistics market.

1.2 Design Goals

Yolla is designed with the following goals in mind:

- **Usability:** The platform ensures an intuitive interface with a seamless user experience across web and mobile applications.
- **Performance:** Yolla is optimized for fast response times, ensuring users can access cargo services with minimal latency.
- **Reliability:** The system guarantees a 99.9% uptime with robust error-handling and backup mechanisms.
- **Marketability:** With a unique blend of comprehensive provider comparisons, real-time tracking, and AI-driven customer support, Yolla aims to capture a significant market share.
- **Extendibility:** The architecture supports the integration of additional features, including partnerships with new logistics providers and AI-driven optimizations.
- **Security:** Yolla implements industry-standard encryption, multi-factor authentication, and compliance with KVKK and GDPR regulations.
- **Scalability:** The system supports a growing user base by employing horizontal scalability and dynamic resource allocation.
- **Maintainability:** Code modularity and continuous integration practices ensure long-term maintainability.
- **Flexibility:** Users can personalize their experience, including notification preferences, payment methods, and cargo service selection criteria.
- **Modularity:** The platform follows a microservices architecture, allowing independent feature enhancements and improvements.
- **Aesthetics:** A clean, responsive design enhances the overall user experience, reinforcing brand identity and accessibility.

1.3 Definitions, Acronyms, and Abbreviations

- **AHP:** Analytic Hierarchy Process
- **API:** Application Programming Interface
- **CLV:** Customer Lifetime Value
- **CPU:** Central Processing Unit
- **FAISS:** Facebook AI Similarity Search
- **GDPR:** General Data Protection Regulation
- **HTTPS:** Hypertext Transfer Protocol Secure
- **ILP:** Integer Linear Programming
- **JWT:** JSON Web Token
- **KVKK:** Kişisel Verilerin Korunması Kanunu (Turkish equivalent of GDPR)
- **LLM:** Large Language Model
- **NGO:** Non-Governmental Organization
- **SaaS:** Software as a Service
- **SME:** Small and Medium-sized Enterprises

1.4 Overview

Yolla is a digital cargo marketplace that facilitates seamless logistics solutions by connecting users with multiple cargo service providers. The system allows users to search, compare, and book cargo services efficiently through web and mobile applications. With real-time tracking, AI-powered chatbot assistance, and an intuitive interface, Yolla enhances the logistics experience for both individuals and businesses. It supports various payment options, automated package measurement, and flexible service choices. Additionally, Yolla promotes sustainability by ranking service providers based on their environmental impact and offering users carbon footprint insights. The platform ensures secure transactions, reliable service uptime, and a scalable infrastructure to accommodate growing user demand.

2. Current Software Architecture

2.1. Competitors, Alternative and Current Solutions

Several applications provide partial solutions to the identified problem, despite certain limitations such as overlooking individual shipments, lacking various comparison options, the unavailability of some services, and software-related issues etc. The following section presents a list of these companies and an analysis of their systems.

2.1.1. Geliver

It is the most prominent active competitor, it offers a comprehensive platform integrating multiple cargo companies such as Aras Kargo, HepsiJet, and Yurtiçi Kargo. It stands out for features like real-time package tracking, e-commerce integrations, and end-to-end management of logistics processes. Geliver's service offerings and extensive partner network position it as the primary competitor for Yolla, presenting a significant challenge in market differentiation [1].

2.1.2. Kargonomi

Another functional cargo startup that is partnering with five major companies: Aras, HepsiJet, PTT, Sürat, and Sendeo. The platform relies on phone-based bookings and advertises economic pricing starting from 29.90 TL. It employs an AI-supported system to recommend the most cost-effective cargo options, making it a notable competitor in terms of pricing transparency and simplicity [2].

2.1.3. Grak Grak Kargo

Startup aimed at uniting senders and couriers on a single platform. It is no longer operational, likely due to low team caliber [3].

2.1.4. Kolibu

A company focused on designing to track multiple cargo services from a single interface, has ceased operations, as evidenced by its inactive LinkedIn profile and absence of updates [4].

2.1.5. KargoBul

The competitive landscape for cargo aggregation platforms indicates a market interest. However the failures of some startups highlight the challenges in sustaining operations in this sector, such as limited scalability, insufficient differentiation. Project must focus more on the failure of the startups to enhance its work. Also active competitors like Geliver and Kargonomi have succeeded in the market by integrating real-time tracking and cost-effective solutions, but significant gaps remain in areas like comprehensive provider comparison, advanced user experience, and affordability for SMEs and freelance e-commerce users which can be utilized by the project Yolla [5].

3. Proposed software architecture

3.1 Overview

Yolla is a mobile and web-based cargo marketplace designed to simplify the process of finding, comparing, and booking cargo services in Türkiye. Users can easily search for cargo options based on location, size, weight, and delivery time while comparing prices and services from

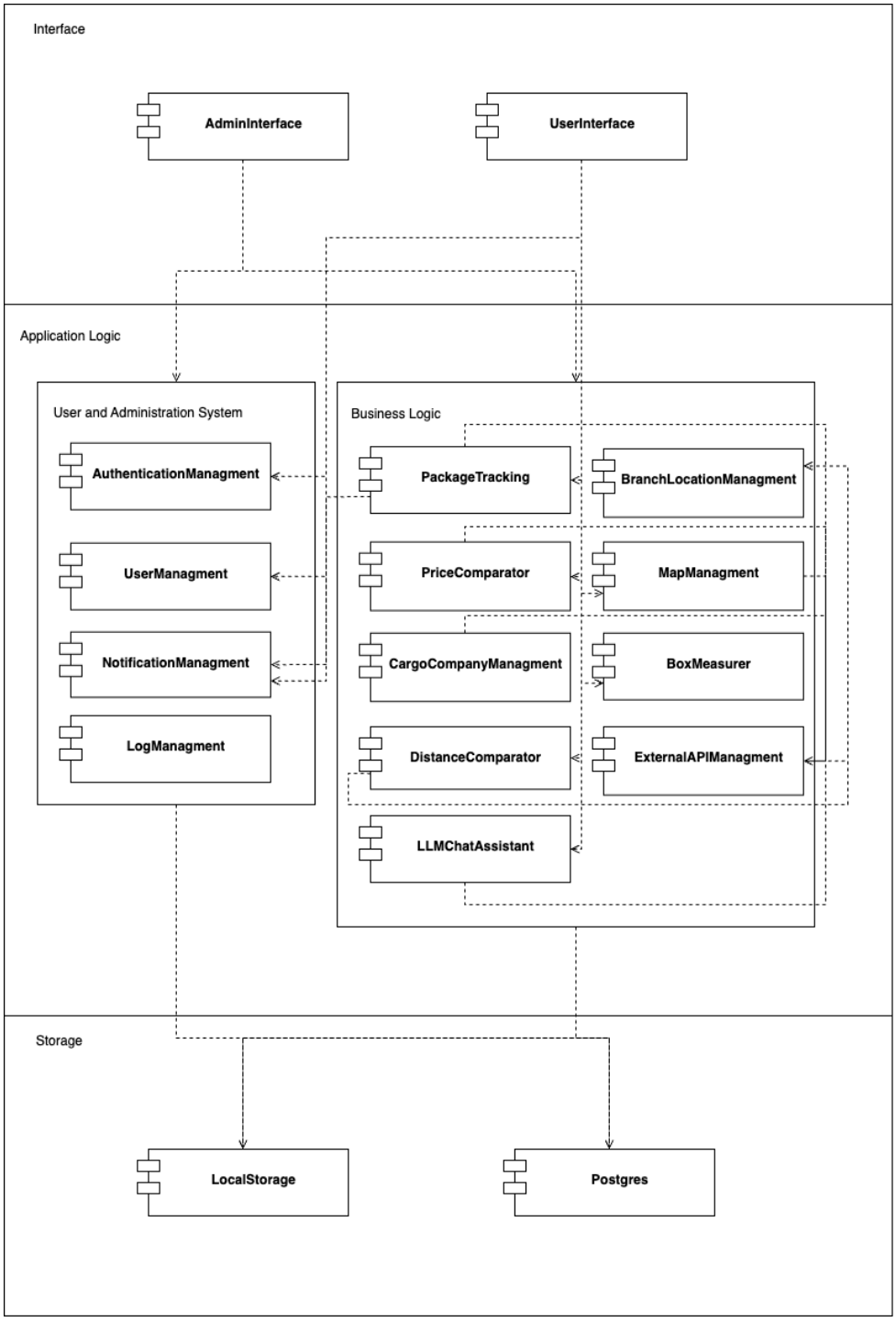
multiple providers. The platform provides an optimized logistics experience through real-time package tracking, notifications, and AI-driven assistance, ensuring a seamless and user-friendly experience for both individuals and SMEs.

Yolla's system is structured around a multi-layered architecture that integrates a scalable backend with a modern frontend and persistent data storage. The frontend is developed using React for the web application and React Native for the mobile version, ensuring a responsive and intuitive user experience. The backend, built with FastAPI, efficiently handles user authentication, package tracking, real-time price comparisons, and integrations with third-party cargo providers via APIs. PostgreSQL serves as the primary database, managing system logs, transaction records, and user data while ensuring data consistency through ACID compliance.

To maintain smooth performance and scalability, Yolla employs cloud infrastructure for hosting, leveraging load balancing and auto-scaling to handle varying traffic loads dynamically. Redis caching optimizes frequent data access, while Docker containerization ensures modular deployment of microservices. Security is enforced through JWT authentication, role-based access control (RBAC), and AES-256 encryption for sensitive data. Additionally, Nginx is used for load balancing, while PgBouncer manages database connection pooling to reduce latency.

By leveraging a modular and scalable architecture, Yolla ensures high availability, reliability, and performance across all system components. The system is designed to handle thousands of concurrent users while maintaining a 99.9% uptime, secure transactions, and a seamless logistics experience.

3.2 Subsystem decomposition



The Interface Layer, Application Logic Layer, and Storage Layer make up Yolla's properly organized three-layer subsystem breakdown, and each has a vital role to play in providing a smooth cargo marketplace experience. The two central elements of the Interface Layer are the User Interface, by which users are able to rapidly search, compare, book, and track shipments,

and the Admin Interface, by which system administrators use to oversee user activity, track cargo activities, and track service providers. User and Administration System and Business Logic are the two Application Logic Layer subsystems where these interfaces interact. User management, authentication, and system log maintenance are in the scope of the User and Administration System. The key components are Notification Management, which offers real-time notifications, User Management, which manages user accounts, Authentication Management, which manages user login and security, and Log Management, which maintains system activity records. On the other hand, the subsystem responsible for processing and optimizing freight operations is referred to as Business Logic. It features an LLM Chat Assistant for delivering AI-powered customer support, Package Tracking to track shipments in real-time, Price Comparator to compare prices across multiple shipping companies, Cargo Company Management for the integration of third-party cargo companies, and Distance Comparator for determining the most effective routes for delivery. In addition, while Map Management improves navigation capabilities, the Branch Location Management module assists customers in locating nearby cargo branches. The Box Measurer determines the package sizes using augmented reality and image processing, while External API Management integrates third-party payment and logistics providers seamlessly. These integrated components of communication occur within the Storage Layer that consists of Postgres as the core database and Local Storage to be used for temporary cache. Postgres holds the secure storage of operating logs, transaction records, cargo details, and user information. As a result of this hierarchical decomposition, Yolla is able to offer data security and system stability with end-to-end communication between users, companies, and logistics providers, resulting in an efficient, scalable, and user-friendly digital cargo marketplace.

3.3 Hardware/software mapping

Yolla's hardware/software mapping is to have its software elements hosted on cloud infrastructure with the provision for scalability and data management efficiency. The central data management system will be the PostgreSQL database to store system logs, user information, cargo, and transactions. Business logic will be handled by the FastAPI backend and will engage third-party APIs for package tracking, real-time price comparison, and AI-driven chatbot services. A mobile-friendly and responsive web user interface will be provided by the frontend, developed using React. AWS, Google Cloud, or Azure will host the whole system to provide high availability, scalability, and reliability. Traffic will be handled efficiently by cloud load balancers, and auto-scaling features will be utilized to deal with unpredictable loads dynamically. The store layer will combine caching software (e.g., Redis) for performance optimization with permanent cloud storage for database management. Additionally, to facilitate the simple deployment and management of microservices, containerization tools like Docker can be utilized alongside Github's workflow features.

3.4 Persistent data management

Yolla's reliable data handling provides for mission-critical system information, like user accounts, cargo transactions, and business workflows, to remain available and recoverable over the long term—through even system crashes or shut downs. PostgreSQL is the central relational database management system (RDBMS) of the system and ensures consistency and reliability through the application of ACID (Atomicity, Consistency, Isolation, Durability) principles. In order to ensure integrity, the database is designed to be able to accept a combination of data types in the form of system logs, cargo data, transactional information, and authentication data. These categories of data are then mapped to relational tables with foreign keys to define complex relationships.

Yolla utilizes several techniques for persistence to maximize data resilience and durability. To allow recovery when there is a failure, Write-Ahead Logging (WAL) guarantees that database modifications are first written to a log before they are committed. Moreover, master-slave replication is used to keep several copies of the database to allow for improved read performance and failover support, and automatic backup runs at regular intervals to give disaster recovery facilities. Yolla uses connection pooling using PgBouncer to minimize latency and improve database queries, Redis caching for frequently accessed data, and partitioning for large tables to effectively manage growing data sizes. Nginx is also used to tackle load-balancing-related issues.

Yolla's cloud-managed strategy for permanent data management is the utilization of cloud-managed PostgreSQL services, such as Amazon RDS or Google Cloud SQL, which provide automatic scaling, maintenance, and security because it's on a cloud provider. To guarantee optimal database performance, large files—such as invoices and shipping receipts—are stored independently through the utilization of cloud object storage (e.g., AWS S3 or Google Cloud Storage). Data encryption (AES-256 when stored, SSL/TLS during transmission), role-based access control (RBAC), and audit logging for visibility and monitoring are all essential components of the system's security and compliance controls. Yolla implements effective, scalable, and secure persistent data management using the combination of PostgreSQL, caching mechanisms, replication, cloud backup, and security features. This provides uninterrupted cargo tracking, company continuity, and adherence to industry standards like GDPR and KVKK.

3.5 Access control and security

Yolla's security management and access control are implemented to provide user information, transactions, and system functions with strong protection. JSON Web Tokens (JWT), enabling secure stateless authentication for everyone, are utilized to manage authentication. The system issues a JWT token, bcrypt hashes the user's password, and checks their credentials when logging in. Secure session management is ensured by the encrypted user data in this token, including the user role and time to live. All subsequent requests include the token, and the backend checks for its validity before providing access. Yolla supports permissions based on

user roles by using role-based access control (RBAC) to provide increased security. Admin accounts handle all system configurations, including user management, transaction monitoring, and security log management. Shippers, or repeat customers, can book shipments, compare prices, search for cargo services, and track shipments. Unregistered guest visitors can view only limited things and are allowed to browse and compare services; they cannot complete transactions. They can compare the price of different services only.

Security management of Yolla involves several tiers of security. Brute force and rainbow table attacks are eliminated by using "bcrypt" hashing with salting to protect the passwords. The admin accounts too are made subject to multi-factor authentication (MFA) to provide additional protection. JWT tokens need refresh tokens for extended periods of sessions and are set to have limited period expiration limits. Cross-site scripting (XSS) attacks are prevented by placing these tokens into HTTP-only cookies. AES-256 is used to encrypt all sensitive user data, including payment information, to provide security during transit and at rest. All client-server communication is also secured using TLS (Transport Layer Security), which defends against man-in-the-middle (MITM) attacks.

Input validation to reduce SQL injection and other cyberattacks, CORS (Cross-Origin Resource Sharing) restrictions to limit unwanted API requests, and rate limiting to avoid DDoS (Distributed Denial-of-Service) attacks are some of the API security measures. Role-based access control (RBAC), which asserts that various services run with the lowest privileges necessary, is used to implement database security for database users. For compliance and monitoring reasons, all changes to administrative activities, user data, and transactions are captured by audit logs. Automated periodic backups guarantee that the system is recovered quickly from unintentional loss of data or cyberattacks.

Whitelisted IPs are the only ones who are provided access to the admin panel, and two-factor authentication (2FA) is required to log in. Administrators can track failed login attempts, suspicious activity, and other security incidents from the dedicated security panel. The system also includes intrusion detection systems (IDS), which track unusual activity like unauthorized API requests or excessive login attempts. Yolla conducts routine penetration testing to find and remediate vulnerabilities and has an account lockout following a series of failed login attempts to restrict probable security breaches. All existing JWT tokens are automatically revoked, admin credentials are changed, and compromised users are informed in case of a security violation.

Yolla keeps its user platform safe and robust with JWT authentication, RBAC authorization, encryption, API security, and strong password policies. The platform ensures trustworthiness and reliability in its cargo marketplace operations by continuously adapting to neutralize new cyber threats and adhering to GDPR and KVKK data protection regulations.

4. Subsystem services

4.1 Client Services (Frontend)

The client services are responsible for rendering the Yolla web and mobile applications. These services interact with the backend via APIs and manage UI logic, state management, and user experience.

4.1.1 User Interface Services

- ➔ Authentication UI Service
  - ◆ Provides login, signup, and password recovery UI
  - ◆ Integrates with OAuth (Google, Apple, Microsoft)
  - ◆ Handles JWT-based session management
  - ◆ Uses Firebase for Multi-Factor Authentication (MFA)
- ➔ Cargo Search UI Service
  - ◆ Allows users to input package details (size, weight, destination)
  - ◆ Fetches and displays cargo options from backend services
  - ◆ Implements filtering (price, delivery time, provider rating)
- ➔ Package Tracking UI Service
  - ◆ Displays real-time shipment updates
  - ◆ Renders real-time tracking updates
- ➔ Box Measurement UI Service
  - ◆ Uses Augmented Reality (AR) for package size estimation
  - ◆ Utilizes mobile camera-based image processing

4.1.2 Notification & Communication UI Services

- ➔ Push Notification UI Service
  - ◆ Manages Firebase Cloud Messaging (FCM) notifications
  - ◆ Renders notifications
- ➔ Chatbot UI Service
  - ◆ Integrates chatbot interface into the support section

4.2 Backend Services in Yolla

The backend of Yolla follows a microservices-based architecture built using FastAPI and PostgreSQL, ensuring scalability, efficiency, and maintainability. The backend services handle authentication, cargo processing, notifications, and external API integrations.

4.2.1 Authentication & User Management Services

- ➔ Authentication Service:
  - ◆ Handles user login, registration, and session management.
  - ◆ Uses JWT tokens for secure authentication.
  - ◆ Implements OAuth 2.0 for third-party logins (Google, Apple, Microsoft).
- ➔ Authorization Service:
  - ◆ Implements Role-Based Access Control (RBAC) for admin, business, and individual users.
- ➔ User Profile Service:
  - ◆ Manages user data, order history, and saved preferences.

4.2.2 Cargo Processing Services

- ➔ Cargo Search Service:
  - ◆ Fetches available cargo services based on user input (location, weight, delivery time).
- ➔ Cargo Comparison Service:
  - ◆ Ranks services based on price, speed, provider ratings, and sustainability.
- ➔ Package Tracking Service:
  - ◆ Provides real-time tracking through logistics provider APIs.
- ➔ Box Measurement Service:
  - ◆ Uses computer vision & augmented reality (AR) to estimate package dimensions.

4.2.3 Notification & Communication Services

- ➔ Email Notification Service:
  - ◆ Sends system-generated emails (registration, order updates, promotional offers).
- ➔ Push Notification Service:
  - ◆ Uses Firebase Cloud Messaging (FCM) for real-time package status updates.
- ➔ SMS Notification Service:
  - ◆ Sends OTPs and package updates via Twilio or a local SMS provider.

4.2.4 AI & Optimization Services

- ➔ AI Chatbot Service:
  - ◆ Uses LangChain + FAISS + Gemini API for real-time customer support.
- ➔ Pricing Optimization Service:
  - ◆ Uses Linear Programming (ILP + Fuzzy AHP) to rank the most cost-effective cargo options dynamically.

4.2.5 External API Integration Services

- ➔ Cargo Provider API Service:

        - Connects with major logistics providers (PTT, Aras Kargo, HepsiJet, Sürat Kargo).
- ➔ Geolocation & Routing Service:
    - Uses Google Maps API for distance calculations and route optimization.
- ➔ Carbon Footprint Estimation Service:
    - Computes emissions based on delivery routes to encourage eco-friendly choices.

5. Test Cases

5.1 Functional Test Cases

The following test cases are designed to ensure that the core functionalities of the yolla.tech web and mobile applications are working as expected. These tests cover key user interactions, such as logging in, searching for cargo options, and placing an order. Each test case specifies the steps involved, expected outcomes, and the priority or severity of the test to ensure that any critical issues are identified and addressed promptly. These functional tests aim to verify both individual features and integration between different components of the system.

Test ID	**TC-FUNC-001**
Category	Functional, Integration
Objective	Verify that a registered user can log in successfully using email and password.
Procedure	1. Open the yolla.tech website. 2. Navigate to the login page. 3. Enter a valid registered email and correct password. 4. Click the "Login" button. 5. Observe the app's response.
Expected Results	After step 4, the user should be redirected to the main dashboard with a welcome message displaying their name (e.g., "Selam Halil, Nereye yollayalım?").
Priority/Severity	Critical
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-002**

Category	Functional, Integration
Objective	Verify that an unregistered user can search for cargo options without logging in.
Procedure	1. Open the yolla.tech web app. 2. From the landing page, select the "Search Cargo" option without logging in. 3. Enter a source address (e.g., "Bilkent Üniversitesi, Ankara") and a destination address (e.g., "Koç Üniversitesi, İstanbul"). 4. Select package type as "Box" and input dimensions (e.g., Desi: 0.4). 5. Click "Search" and observe the results.
Expected Results	After step 5, a list of cargo options should be displayed with details like price, delivery time, and provider, sorted by relevance (e.g., "En yeşil", "En hızlı").
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-003**

Category	Functional, Integration
Objective	Verify that a registered user can place a cargo order and receive a QR code or instructions for supported cargo companies.
Procedure	1. Log in to the yolla.tech app as a registered user. 2. Navigate to the "Send Cargo" page. 3. Enter source and destination addresses, select "Box" as the package type, and input dimensions. 4. Apply a promotional code if available, then select a cargo option (e.g., "En hızlı"). 5. Confirm the order and proceed to payment via a gateway (iyzico Payment Gateway API) if payment via yolla.tech is applicable. 6. Complete the payment and observe the response.
Expected Results	After step 6, the user should receive a confirmation message with a QR code or order number and the order should appear in the "Kargolarım" section with status "Yolladın!".
Priority/Severity	Critical
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-004**
Category	Functional, Integration
Objective	Verify that a non-user can track a package using a tracking code.
Procedure	1. Open yolla.tech website without login. 2. Navigate to the "Kargo Takip Et" section. 3. Enter the tracking code manually in the "Kargo Takip Et" page. 4. Observe the tracking status.
Expected Results	After step 3 or 4, the tracking details should display the current status (e.g., "Adana Transfer Merkezi'nde") and estimated delivery date (e.g., "Tahmini teslimat: 2 Aralık").
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-005**

Category	Functional, Integration
Objective	Verify that a user receives notifications for cargo status updates.
Procedure	1. Log in to yolla.tech as a registered user. 2. Place a new cargo order (as in TC-FUNC-003). 3. Wait for the cargo status to update (e.g., simulate a status change to "Shipped" via admin interface). 4. Check the "Bildirimleriniz" section in the app. 5. Verify if a notification is received via email (if enabled).
Expected Results	After step 4, a notification should appear in the app (e.g., "Sadra sana bir koli yolladı"). If email notifications are enabled, a message with the tracking code should be received.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-006**
Category	Functional, Integration

Objective	Verify that a user can measure a box using the phone's camera.
Procedure	1. Log in to the yolla.tech app. 2. Navigate to the "Send Cargo" page and select "Box" as the package type. 3. Click the "Ölçmek istersen tıkla" option to use the camera-based Box Measurer. 4. Point the camera at a box and follow on-screen instructions to capture dimensions. 5. Submit the measurement and observe the recorded dimensions.
Expected Results	After step 5, the app should display the box dimensions (e.g., "Desi: 0.4") and allow the user to proceed with the order.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-007**
Category	Functional, Integration

Objective	Verify that a user can interact with the LLM Chat Assistant for support.
Procedure	1. Open the yolla.tech website. 2. Navigate to the "Destek" section. 3. Type a query in the chat (e.g., "How do I track my cargo?"). 4. Submit the query and observe the chatbot's response. 5. Ask a follow-up question (e.g., "What if I lost my tracking code?"). 6. Verify the response and session continuity.
Expected Results	After step 4, the chatbot should provide a relevant answer (e.g., "You can track your cargo in the 'Kargolarım' section using your tracking code."). After step 6, the chatbot should maintain context and respond appropriately.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-008**
Category	Functional, Integration

Objective	Verify that a user can filter cargo options by price and delivery time.
Procedure	1. Open the yolla.tech web app and navigate to the "Search Cargo" page. 2. Enter source and destination addresses. 3. Set a price range filter (e.g., 50-100 TL). 4. Set a delivery time filter (e.g., "Express"). 5. Click "Search" and observe the results.
Expected Results	After step 5, the results should only show cargo options within the specified price range and delivery time (e.g., "En hızlı" options with prices between 50-100 TL).
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-009**
Category	Functional, Integration
Objective	Verify that an admin can manage user accounts via the Admin Interface.

Procedure	1. Log in to the yolla.tech web app as an admin user. 2. Navigate to the "User Management" section in the Admin Interface. 3. Select a user account to edit. 4. Save the changes. 5. Log out and log in as the edited user to verify the changes.
Expected Results	After step 4, the user's email should be updated in the system. After step 5, the user should be able to log in with the updated email.
Priority/Severity	Critical
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-010**
Category	Functional, Integration
Objective	Verify that a business user can track multiple cargo shipments.

Procedure	1. Log in to the yolla.tech app as a business user. 2. Navigate to the "Business Cargo Management" section. 3. View the list of sent cargos (e.g., 5 cargos with different statuses). 4. Select one cargo to view detailed tracking information. 5. Verify the status updates for all cargos.
Expected Results	After step 3, a list of cargos should be displayed with their statuses (e.g., "Delivered", "In Transit"). After step 4, detailed tracking info should match the cargo status.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-011**
Category	Functional, Integration
Objective	Verify that a user can add insurance to a cargo order.

Procedure	1. Log in to the yolla.tech app. 2. Start a new cargo order (as in TC-FUNC-003). 3. At the order summary page, select the "Add Insurance" option. 4. Choose an insurance provider and confirm. 5. Complete the payment and place the order. 6. Verify the order details in the "Kargolarım" section.
Expected Results	After step 5, the order should be placed successfully. After step 6, the order details should include the selected insurance provider and additional cost.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-012**
Category	Functional, Integration
Objective	Verify that a user can rate and review a cargo service after delivery.

Procedure	1. Log in to the yolla.tech app. 2. Navigate to the "Kargolarım" section and select a delivered cargo (e.g., "Teslim edildi!"). Click the "Rate & Review" option. 4. Provide a rating (e.g., 4 stars) and a review comment (e.g., "Fast delivery, good service"). 5. Submit the review and check if it’s reflected in the cargo provider’s profile.
Expected Results	After step 5, the review should be submitted, and the cargo provider’s profile should reflect the new rating and comment.
Priority/Severity	Minor
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-013**
Category	Functional, Integration
Objective	Verify that the system displays promotions and advertisements on the main page.

Procedure	1. Open the yolla.tech app without logging in. 2. Navigate to the main page. 3. Click on the "Kampanyalar" section. 4. Click on a promotion (e.g., a discount offer). 5. Verify if the promotion details are displayed and can be applied to a new order.
Expected Results	After step 3, promotions should be visible (e.g., discount offers). After step 5, the promotion should be applied to a new order if the user proceeds.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-014**
Category	Functional, Integration
Objective	Verify that donation to TEMA for carbon emissions works.

Procedure	1. Log in to the yolla.tech website. 2. Book cargo between two locations. 3. In the payment step of the bo, redirect to the TEMA's website for donation in account of the carbon emissions.
Expected Results	After step 3, the carbon emission part in the should be displayed as zero in the "Kargolarım" page (e.g., "Bu teslimatın karbon salınımı SIFIR!"). Also donated amount should be added to the x in "TEMA'ya x ₺ bağışladın" part.
Priority/Severity	Minor
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-015**
Category	Functional, Integration
Objective	Verify that a user can save and reuse a favorite delivery address.
Procedure	1. Log in to the yolla.tech app. 2. Navigate to the "Hesabım" section and select "Hesap Ayarları".

	3. Add a new favorite address (e.g., "Bilkent Üniversitesi, Ankara"). 4. Start a new cargo order and check if the saved address appears as an option. 5. Select the saved address and proceed with the order.
Expected Results	After step 4, the saved address should appear in the address dropdown. After step 5, the order should use the selected address correctly.
Priority/Severity	Minor
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-016**
Category	Functional, Integration
Objective	Verify that a user can update their password successfully via the "Şifreni Güncelle" page.
Procedure	1. Log in to the yolla.tech website. 2. Navigate to the "Hesabım" section and select "Şifreni Güncelle". 3. Enter a new password and confirm it in the respective fields.

	4. Click the "Güncelle" button. 5. Log out and attempt to log in with the new password.
Expected Results	After step 4, a success message should appear (e.g., "Şifreniz güncellendi"). After step 5, the user should log in successfully with the new password.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-017**
Category	Functional, Integration
Objective	Verify that a user can invite friends and earn coupons via the "Davet Et" feature.
Procedure	1. Log in to the yolla.tech website. 2. Navigate to the "Davet Et" section. 3. Copy the referral link or enter a friend’s phone number. 4. Have a friend sign up using the link or number and place an order.

	5. Check the "Davet Ettiklerin" section for updates and the user's coupon balance.
Expected Results	After step 5, the user should see the invited friend listed (e.g., with a check mark) and their coupon balance increased by 50 TL (e.g., "Toplam Kazancın: 100 TL").
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-018**
Category	Functional, Integration
Objective	Verify that a user can update their profile information (e.g., phone number, email) and the changes are reflected across the platform.
Procedure	1. Log in to the yolla.tech web app. 2. Navigate to "Profil ve Tercihler" and view the current profile details (e.g., phone: +90 210 420 03 03, email: user@example.com). 3. Update the phone number to a new value (e.g., +90 555 987 65 43) and the email to a new address (e.g., newuser@example.com).

	4. Save the changes and verify a confirmation message (e.g., "Profil güncellendi"). 5. Place a new order and check if the updated contact details are used in the order details. 6. Log out, log back in, and confirm the updated details are still present in the profile.
Expected Results	After step 4, the profile should reflect the new phone number and email. After step 5, the order details should use the updated contact information (e.g., +90 555 987 65 43). After step 6, the profile should still show the updated details.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-019**
Category	Functional, Integration
Objective	Verify that a user can toggle notification settings (e.g., email, push) in the "Profil ve Tercihler" section.

Procedure	1. Log in to the yolla.tech website. 2. Navigate to "Profil ve Tercihler" and find "Bildirim Ayarları". 3. Toggle off the "E-posta" option for "Kargo Hareketleri". 4. Place a new order and check for email notifications. 5. Toggle it back on and verify the change.
Expected Results	After step 4, no email should be received for the order update. After step 5, an email should be received for the next update.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-020**
Category	Functional, Integration
Objective	Verify that a user can request account deletion via the "Hesap Kapatma Talebi" option.
Procedure	1. Log in to the yolla.tech web app. 2. Navigate to "Profil ve Tercihler" and select "Hesap Kapatma Talebi".

	3. Confirm the request. 4. Check for a confirmation email or notification. 5. Attempt to log in after 24 hours.
Expected Results	After step 4, a confirmation message or email should be received. After step 5, login should fail with an "Account deleted" message.
Priority/Severity	Critical
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-021**
Category	Functional, Integration
Objective	Verify that a user can link a social media account (e.g., Google) to their yolla.tech profile.
Procedure	1. Log in to the yolla.tech app. 2. Navigate to "Profil ve Tercihler" and select "Bağlı Hesaplar". 3. Click the Google icon to link an account. 4. Authenticate with Google credentials.

	5. Verify the linked account in the profile.
Expected Results	After step 4, the Google account should be linked successfully. After step 5, the Google logo should appear under "Bağlı Hesaplar".
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-022**
Category	Functional, Integration
Objective	Verify that a user can view and redeem coupons earned from the referral program.
Procedure	1. Log in to the yolla.tech account. 2. Navigate to "Davet Et" and check the "Toplam Kazancın" (e.g., 100 TL). 3. Start a new order and apply the coupon at checkout. 4. Verify the discount is applied to the total. 5. Complete the order and check the updated coupon balance.

Expected Results	After step 4, the order total should decrease by the coupon amount (e.g., 100 TL off). After step 5, the coupon balance should reflect the used amount.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-023**
Category	Functional, Integration
Objective	Verify that a user can add a recipient’s contact details during order creation and the details are used for delivery notifications.
Procedure	1. Log in to the yolla.tech app. 2. Navigate to "Send Cargo" and start a new order (e.g., source: Ankara, destination: İzmir, package: 10x8x10 cm, 2 kg). 3. In the recipient details section, enter a contact name (e.g., Mehmet Kaya) and an email. 4. Complete the order and note the tracking code (e.g., L62RST). 5. Monitor the package status until it reaches "Out for delivery," and verify the recipient receives an email notification.

	6. Confirm delivery in "Kargolarım" and check the delivery note for the recipient’s details.
Expected Results	After step 4, the order should include the recipient’s details in "Kargolarım." After step 5, the recipient should receive an email (e.g., "Your package L62RST is out for delivery"). After step 6, the delivery note should confirm "Delivered to Mehmet Kaya”.
Priority/Severity	Minor
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-024**
Category	Functional, Integration
Objective	Verify that a user can search for a package using a tracking code in the "Kargolarım" section.
Procedure	1. Log in to the yolla.tech website. 2. Navigate to "Kargolarım" and enter a tracking code in the search bar. 3. Click the search icon and observe the results. 4. Verify the package status and details. 5. Repeat with an invalid tracking code.

Expected Results	After step 3, the package status (e.g., "Adana Transfer Merkezi'nde") should display. After step 5, an error message (e.g., "Invalid tracking code") should appear.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-025**
Category	Functional, Integration
Objective	Verify that a user can view historical order data in the "Kargolarım" section.
Procedure	1. Log in to the yolla.tech web app. 2. Navigate to "Kargolarım" and scroll to view past orders (e.g., from December 2024). 3. Select an old order and view details like cost and status.
Expected Results	After step 3, all historical data should display correctly.

Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-026**
Category	Functional, Integration
Objective	Verify that a user can share a tracking link via WhatsApp from the "Varması Beklenenler" section.
Procedure	1. Log in to the yolla.tech website. 2. Navigate to "Bildirimler" and select a package under "Varması Beklenenler". 3. Click the share icon and select WhatsApp. 4. Send the tracking link to a contact and verify receipt. 5. Check if the contact can view the tracking status.
Expected Results	After step 4, the contact should receive the tracking link. After step 5, the contact should see the package status (e.g., "Bilkent PTT").
Priority/Severity	Minor

Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-027**
Category	Functional, Integration
Objective	Verify that a user can view detailed notifications in the "Bildirimler" section.
Procedure	1. Log in to the yolla.tech website. 2. Navigate to "Bildirimler" and select a notification (e.g., "Halil tarafında gönderilen kargonuz yolda"). 3. View the full details (e.g., tracking code, status). 4. Mark the notification as read and verify it disappears from the unread list. 5. Repeat with a "Puanlama için dokun" notification.
Expected Results	After step 3, detailed information should display. After step 4, the notification should move to the read list. After step 5, a rating prompt should appear.
Priority/Severity	Major

Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-028**
Category	Functional, Integration
Objective	Verify that a user can cancel a pending order before it is processed.
Procedure	1. Log in to the yolla.tech app. 2. Place a new order but do not approve it yet. 3. Navigate to "Siparişlerim" and select the pending order. 4. Click "İptal Et" and confirm cancellation. 5. Verify the order no longer appears in "Siparişlerim".
Expected Results	After step 4, a cancellation confirmation should appear. After step 5, the order should be removed from the list.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)

Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-029**
Category	Functional, Integration
Objective	Verify that a user can recover their account using the "Şifreni Unuttun mu?" feature with multiple verification methods.
Procedure	1. Log out of the yolla.tech app. 2. On the login screen, select "Şifreni Unuttun mu?" and enter the registered email (e.g., user@example.com). 3. Choose to receive a verification code via email and enter the code sent. 4. Set a new password and attempt to log in with it. 5. Repeat the process but select phone verification (+90 210 420 03 03) instead, entering the SMS code received. 6. Verify successful login and check that account data remains intact.
Expected Results	After step 3, a verification code should be emailed. After step 4, login should succeed with the new password. After step 5, an SMS code should be received, and login should succeed again. After step 6, all account data should be preserved.
Priority/Severity	Critical

Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-030**
Category	Functional, Integration
Objective	Verify that a user can select different package types (e.g., Kutu, Zarf) during order creation.
Procedure	1. Log in to the yolla.tech web app. 2. Navigate to "Send Cargo" and select "Zarf" as the package type. 3. Enter dimensions and weight, then search for options. 4. Switch to "Kutu" and repeat the search. 5. Compare the pricing and options for both types.
Expected Results	After step 3 and 4, the system should display relevant options for each package type, with differing costs based on type (e.g., envelope vs. box).
Priority/Severity	Major
Date Tested	(To be filled in Final Report)

Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-031**
Category	Functional, Integration
Objective	Verify that a user can track multiple packages simultaneously in the "Kargolarım" section.
Procedure	1. Log in to the yolla.tech app. 2. Navigate to "Kargolarım" and select multiple packages. 3. View the tracking details for all selected packages. 4. Switch between packages and verify status updates. 5. Use the "Bu kargoyu takip listesine ekle" option for one package.
Expected Results	After step 3 and 4, the system should display relevant options for each package type, with differing costs based on type (e.g., envelope vs. box).
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-032**
Category	Functional, Integration
Objective	Verify that a user can request a pickup from a specific address in the order process.
Procedure	1. Log in to the yolla.tech app. 2. Start a new order and select the "Adresten Al" option. 3. Enter a pickup address (e.g., "Bilkent 82. Yurt"). 4. Proceed with the order and verify the pickup fee. 5. Check the order status in "Kargolarım" after pickup is scheduled.
Expected Results	After step 4, the order total should include the pickup fee (e.g., 21 TL). After step 5, the status should reflect "Pickup scheduled".
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-033**

Category	Functional, Integration
Objective	Verify that a user can view a map and modify the delivery route during order creation.
Procedure	1. Log in to the yolla.tech web app. 2. Start a new order and enter "Ordu" as the source and "Malatya" as the destination. 3. View the map and click "Adresi Değiştir" to update the destination to "Ankara". 4. Verify the updated route and cost. 5. Approve the order.
Expected Results	After step 4, the order total should include the pickup fee (e.g., 21 TL). After step 5, the status should reflect "Pickup scheduled".
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-034**
Category	Functional, Integration

Objective	Verify that a user can request a return (iade) for a delivered package.
Procedure	1. Log in to the yolla.tech app. 2. Navigate to "Kargolarım" and select a delivered package. 3. Click "Kolay İade Et" and follow the return request process. 4. Confirm the request and check the status. 5. Verify the return tracking details.
Expected Results	After step 4, a return request confirmation should appear. After step 5, the return status should be visible (e.g., "Return in progress").
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-FUNC-035**
Category	Functional, Integration
Objective	Verify that a user can view "Biz Kimiz?" information and navigate back to the order page.

Procedure	1. Log in to the Yolla Tech web app. 2. Start a new order and click "Biz Kimiz?" on the map page. 3. Read the information and click "Durak Takip Et" or another option. 4. Verify the return to the order creation page. 5. Complete the order process.
Expected Results	After step 3, the "Biz Kimiz?" content should display. After step 4, the user should return to the order page seamlessly.
Priority/Severity	Minor
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

5.2 Non-Functional Test Cases

Non-functional test cases evaluate the performance, usability, security, and other system attributes that are crucial for the overall user experience and system stability. These tests ensure that the yolla.tech platform performs well under different conditions and meets various non-functional requirements such as load capacity, response time, and security protocols. The following test cases address these aspects to ensure the platform operates efficiently, securely, and consistently, even during peak usage or under other stress conditions.

Test ID	**TC-NONFUNC-001**
Category	Performance, Integration

Objective	Verify that the app loads pages within 3 seconds under normal network conditions.
Procedure	1. Open the yolla.tech app on a device with a stable Wi-Fi or 4G connection. 2. Start a timer and navigate to the "Search Cargo" page. 3. Stop the timer and record the time taken to load the page. 4. Repeat for the "Kargolarım" and "Hesabım" pages. 5. Perform the test 5 times for each page and calculate the average load time.
Expected Results	The average load time for each page should be less than 3 seconds.
Priority/Severity	Critical
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-002**
Category	Performance, Integration

Objective	Verify that the app can handle 10,000 concurrent users without performance degradation.
Procedure	1. Set up a test environment with 10,000 simulated users using a load testing tool. 2. Simulate all users searching for cargo options simultaneously. 3. Measure the response time for search results. 4. Monitor server CPU and memory usage during the test. 5. Check for any crashes or errors in the app.
Expected Results	Search results should be returned within 3 seconds for all users, with no crashes. CPU and memory usage should remain within acceptable limits (e.g., <80% utilization).
Priority/Severity	Critical
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-003**
Category	Security, Integration

Objective	Verify that all data transmissions use secure HTTPS protocols.
Procedure	1. Open the yolla.tech web app on a browser with developer tools enabled. 2. Navigate to the "Search Cargo" page and perform a search. 3. Open the browser’s network tab and inspect the requests. 4. Repeat for login and payment processes. 5. Check the protocol used for each request.
Expected Results	All requests (search, login, payment) should use HTTPS, with no HTTP requests detected.
Priority/Severity	Critical
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-004**
Category	Security, Integration
Objective	Verify that the app prevents unauthorized access to user accounts.

Procedure	1. Open the yolla.tech website. 2. Attempt to log in with a valid email but an incorrect password 5 times. 3. Observe the app's response after the 5th attempt. 4. Attempt to access a user's "Kargolarım" section without logging in by directly navigating to the URL (if web app). 5. Verify if the app blocks access.
Expected Results	After step 3, the app should lock the account temporarily. After step 5, the app should redirect to the login page, preventing access.
Priority/Severity	Critical
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-005**
Category	Stability, Integration
Objective	Verify that the app remains stable during prolonged usage.

Procedure	1. Open the yolla.tech website. 2. Log in and perform multiple actions (search cargo, track packages, place orders) continuously for 4 hours. 3. Monitor for crashes, freezes, or unexpected errors. 4. Check memory usage on the device periodically.
Expected Results	The app should not crash or freeze during the 4-hour test. Memory usage should remain stable (e.g., no memory leaks).
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-006**
Category	Reliability, Integration
Objective	Verify that the app maintains 99.9% uptime during a 24-hour test period.
Procedure	1. Deploy the yolla.tech app on the hosting environment. 2. Use a monitoring tool to check app availability every 5 minutes for 24 hours.

	3. Perform regular user actions (search, track, login) during the test. 4. Record any downtime or unavailability. 5. Calculate the uptime percentage.
Expected Results	The app should be available for at least 99.9% of the time (less than 86 seconds of downtime in 24 hours).
Priority/Severity	Critical
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-007**
Category	Compliance, Integration
Objective	Verify that the app complies with KVKK by allowing users to delete their data.
Procedure	1. Log in to the yolla.tech app as a registered user. 2. Navigate to the "Hesabım" section and select "Hesap Ayarları". 3. Choose the "Delete Account" option. 4. Confirm the deletion and observe the response.

	5. Attempt to log in with the deleted account credentials.
Expected Results	After step 4, the account should be deleted, and a confirmation message should be displayed. After step 5, login should fail with an "Account not found" message.
Priority/Severity	Critical
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-008**
Category	Usability, Integration
Objective	Verify that the app is usable for users with visual impairments using screen readers.
Procedure	1. Open the yolla.tech app on an iOS device with VoiceOver enabled. 2. Navigate to the "Search Cargo" page using VoiceOver. 3. Perform a search by entering source and destination addresses via voice input. 4. Listen to the search results read by VoiceOver. 5. Repeat on an Android device with TalkBack enabled.

Expected Results	VoiceOver/TalkBack should read all elements (buttons, labels, results) correctly, allowing the user to complete the search without issues.
Priority/Severity	Minor
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-009**
Category	Compatibility, Integration
Objective	Verify that the app works on supported mobile OS versions.
Procedure	1. Install the yolla.tech mobile app on an Android 8.0 device. 2. Perform a cargo search, place an order, and track a package. 3. Repeat the test on an iOS 12 device. 4. Check for any crashes, UI distortions, or functional issues. 5. Verify that all features work as expected on both OS versions.
Expected Results	The app should function correctly on both Android 8.0 and iOS 12, with no crashes or UI issues.

Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-010**
Category	Compatibility, Integration
Objective	Verify that the web app works on supported browsers (Chrome, Firefox, Safari, Edge).
Procedure	1. Open the yolla.tech web app on Chrome. 2. Perform a cargo search, log in, and view the profile page. 3. Repeat the test on Firefox, Safari, and Edge. 4. Check for any UI distortions, broken features, or errors. 5. Verify that all features work consistently across browsers.
Expected Results	The web app should function correctly on all supported browsers, with no UI distortions or functional issues.
Priority/Severity	Major

Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-011**
Category	Performance, Integration
Objective	Verify that search queries return results within 2 seconds.
Procedure	1. Open the yolla.tech mobile app on a device with a stable internet connection. 2. Navigate to the "Search Cargo" page. 3. Enter source and destination addresses and start a timer. 4. Click "Search" and stop the timer when results are displayed. 5. Repeat the test 10 times and calculate the average response time.
Expected Results	The average response time for search queries should be less than 2 seconds.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)

Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-012**
Category	Reliability, Integration
Objective	Verify that the app recovers within 15 minutes after a server failure.
Procedure	1. Deploy the yolla.tech app on the hosting environment. 2. Simulate a server failure by stopping the backend server. 3. Attempt to perform a cargo search and note the error. 4. Restart the server and start a timer. 5. Retry the cargo search every minute until it succeeds, then stop the timer.
Expected Results	The app should recover and allow successful cargo searches within 15 minutes after the server restart.
Priority/Severity	Critical
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-013**
Category	Usability, Integration
Objective	Verify that the app supports multiple languages (Turkish and English).
Procedure	1. Open the yolla.tech app. 2. Navigate to the "Hesabım" section and change the language to English. 3. Verify that all UI elements (labels, buttons, messages) are displayed in English. 4. Perform a cargo search and check the results in English. 5. Switch back to Turkish and repeat the verification.
Expected Results	The app should display all UI elements and results in the selected language (English or Turkish) without errors.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-014**
Category	Scalability, Integration
Objective	Verify that the app can handle a 100% increase in user base without redesign.
Procedure	1. Set up a test environment with the current user base. 2. Simulate normal usage (search, track, order) and measure performance metrics (response time, server load). 3. Increase the simulated user base to 10,000 users (100% increase). 4. Repeat the usage simulation and measure the same metrics. 5. Check for any performance degradation or crashes.
Expected Results	The app should handle 10,000 users with response times and server load similar to the 5,000-user test, with no crashes.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

Test ID	**TC-NONFUNC-015**

Category	Efficiency, Integration
Objective	Verify that the mobile app minimizes battery consumption during usage.
Procedure	1. Install the yolla.tech mobile app on an Android device with a battery monitoring app. 2. Fully charge the device and note the battery level. 3. Use the app continuously for 1 hour (search, track, chat with support). 4. Record the battery consumption after 1 hour. 5. Repeat on an iOS device and compare the results.
Expected Results	Battery consumption should be minimal (e.g., less than 10% per hour) on both devices during normal usage.
Priority/Severity	Major
Date Tested	(To be filled in Final Report)
Test Result	(To be filled in Final Report)

6. Consideration of Various Factors in Engineering Design

6.1. Constraints

6.1.1. Implementation Constraints

- Git and GitHub will be used for managing code, collaboration, and tracking changes in the project.
- Jira will be used for tracking tasks and managing project progress.
- React will be used to build the web application interface.
- React Native will be used for building the mobile application interface.
- FastAPI will be used for handling the backend of the project.
- PostgreSQL will be used as the database for managing and storing data.
- Automated testing and CI/CD will be used to have faster and more reliable development and deployment processes.
- Cloud hosting and infrastructure will be used to deploy and scale the application.
- LangChain and FAISS will be utilized to enable a Retrieval-Augmented Generation (RAG)-based chatbot service, integrated with the Gemini API.

6.1.2. Economic Constraints

Yolla will be a SaaS project with various types of costs such as maintenance, license, and marketing costs.

- **Campaign costs:** Yolla aims to drive a shift in customer habits, which will require investment in promotional campaigns designed to influence and reshape these behaviors.
- **Database maintenance and server costs:** Yolla will be a web-based service, therefore there will be hosting and domain costs. Growing number of users will require storing more data, which will increase our database maintenance and storage costs. We also plan to launch mobile apps, which will include App Store and Google Play Store publication costs.
- **API costs:** APIs will be utilized to establish connections with various package delivery companies, send queries to LLMs, and retrieve estimated walking times from Google Maps. While free alternatives are available, paid APIs may be required to ensure higher quality services and reliability.
- **Legal compliance:** Additional investments in security measures and legal consultations may be necessitated to ensure compliance with local regulations and to uphold data security standards.

- **Marketing:** Funds may need to be allocated for advertising, brand building, and promotional activities to effectively reach the target audience and enhance brand visibility.

Additional costs may occur during our incremental deployment with the features we add based on the user feedback.

6.1.3. Ethical Constraints

- **Contributing to Environmental Pollution by Promoting the Cargo System**: Encouraging the use of cargo services may lead to increased environmental pollution, particularly if the promoted services rely on inefficient or high-emission vehicles.

- **Contributing to Market Monopolization**: Yolla can serve to market monopolization as large cargo enterprises may give a more competitive price opposite to the small cargo enterprises.

6.2. Standards

To ensure the Yolla project aligns with industry practices, ethical principles, and user expectations, the following standards will be established:

6.2.1. Development Standards

- **Coding Practices**: Consistent coding styles will be maintained across all files in Yolla's code repository. Meaningful names will be used for variables, functions, and classes to enhance readability and maintainability.
- **Version Control**: Git will be employed for version control, with clear commit messages ensuring transparency in development progress. The repository will be hosted on GitHub, and separate branches will be utilized for code changes. Merges to the master branch will only be performed after thorough verification to prevent potential issues.
- **Documentation**: Comments and documentation will be added for all critical functions and modules to facilitate understanding and collaboration. README files will be provided to guide the project setup process.

6.2.2. Security Standards

- **Authentication**: We will use JWT tokens for user sessions to ensure security. We will implement two-factor authentication (2FA) for enhanced security.
- **Data Encryption**: We will encrypt sensitive data in transit (using HTTPS) to prevent attacks on our data. We will ensure compliance with KVKK (The law in Türkiye to protect personal data) for data privacy.

6.2.3. User Experience Standards

- **Design Consistency**: We will ensure everyone uses consistent UI features such as fonts and colors to create our branding and to make our product visually appealing.
- **Feedback**: We will provide feedback to user actions in order to prevent events with no response from the system. These feedback may be in the form of warning messages, notifications, or toasts.
- **Speed**: We will ensure that the page load times are under 3 seconds to enhance user experience. We will reduce API calls by using cache.

6.2.4. Legal and Compliance Standards

- **Data Privacy Compliance**: We will ensure full compliance with Türkiye's KVKK and other applicable laws, as explained in Section 1.4.

Contracts: We will establish partnerships with cargo providers, ensuring mutual benefit.

7. Teamwork Details

7.1 Contributing and functioning effectively on the team

- **Celal Salih Türkmen:** Celal is responsible for continuous integration and testing of the project. He set up GitHub workflows to streamline the development process and is also working on various parts of the backend repository. Additionally, he handles deployment and the configuration of domain services.
- **Halil Tataroğlu**: Halil is responsible for marketing and manages communications with cargo companies, working on their API integrations. In addition, he develops the frontend, designing and implementing the website application. He is also responsible for local API development, ensuring integration with systems of the other cargo companies.
- **Sadra Mohammadzadehazarabadi:** Sadra is responsible for mainly the backend implementation and data communication between the frontend and backend. He is also in charge of architecting the project while maintaining a robust tech stack. He also dabbles in task management and coordination of his fellow team members.
- **Eslim Rana Emiroğlu:** Eslim is responsible for the frontend development. She manages the UI designs and implementations. She leads the frontend team. She also takes care of frontend's DevOps jobs. She contributes to the team with the system design. She is responsible from the communication with innovation expert and the advisor.
- **Elif Şen:** Elif is responsible for marketing and manages communications with cargo companies, managing company specialized campaigns. In addition, she contributes to the team as a full stack developer. She is responsible for distance location services and development of certain frontend pages.

7.2 Helping creating a collaborative and inclusive environment

We divide tasks and create subgroups to complete them. Thanks to this approach, we are able to manage tasks effectively and eliminate the need to be aware of every aspect of the project. We are using several platforms actively to create a collaborative environment. Some of them are listed below:

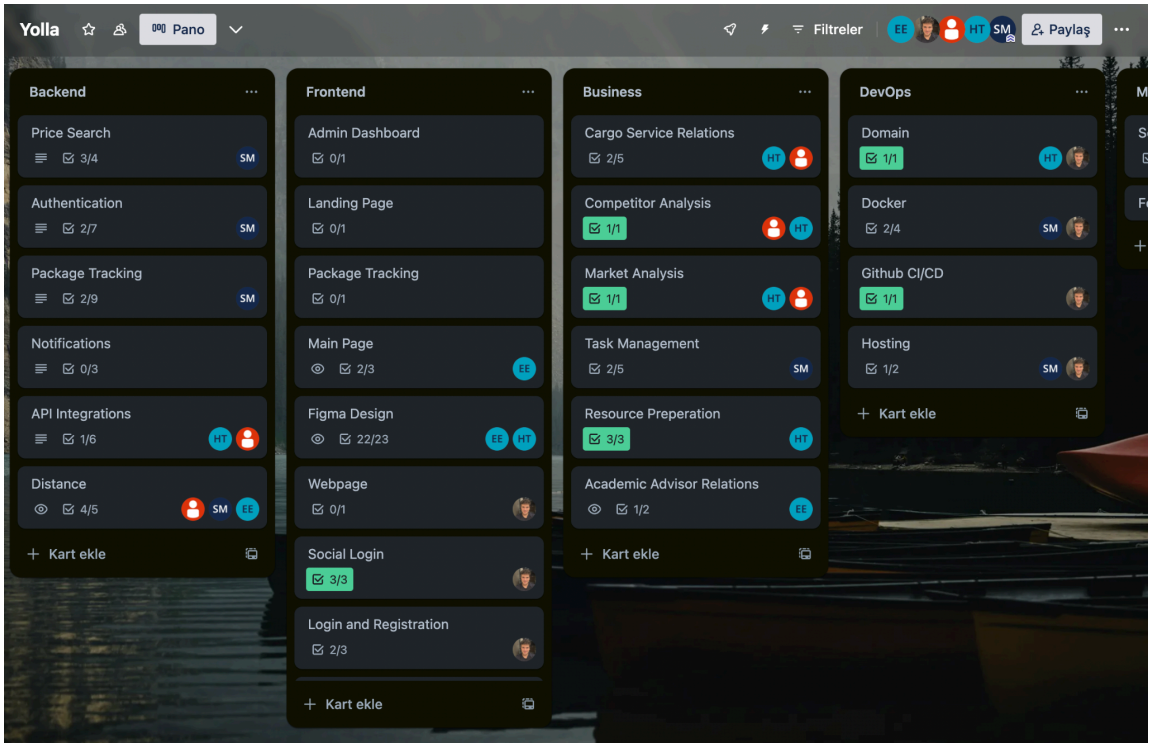


Figure 1: Trello

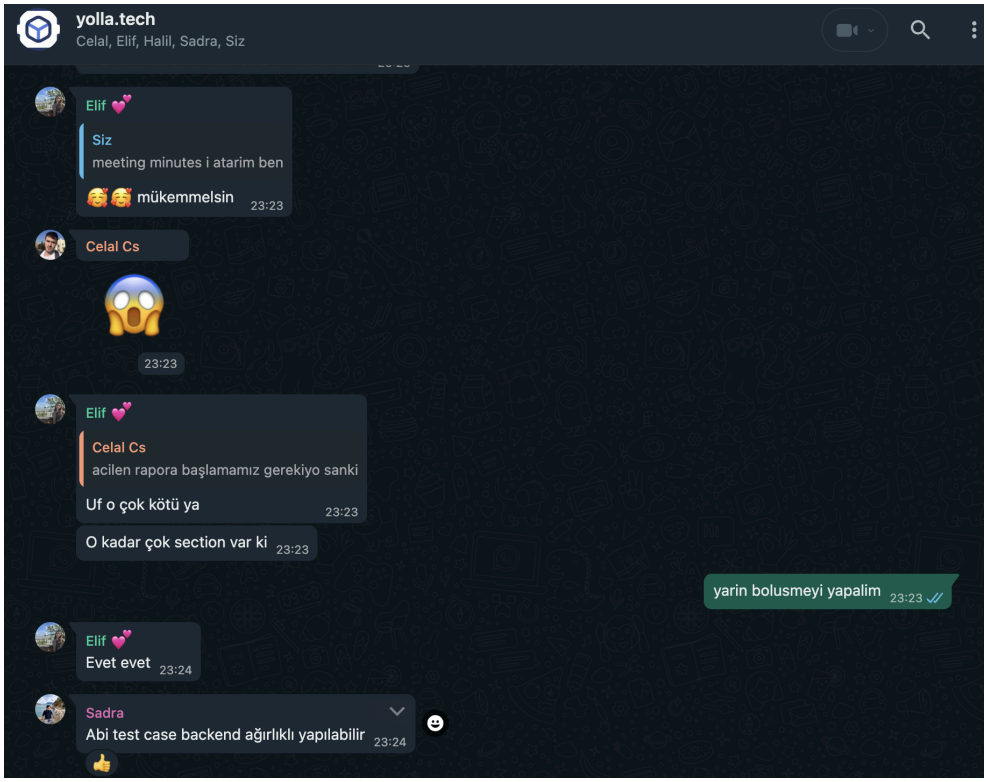


Figure 2: Whatsapp Chat

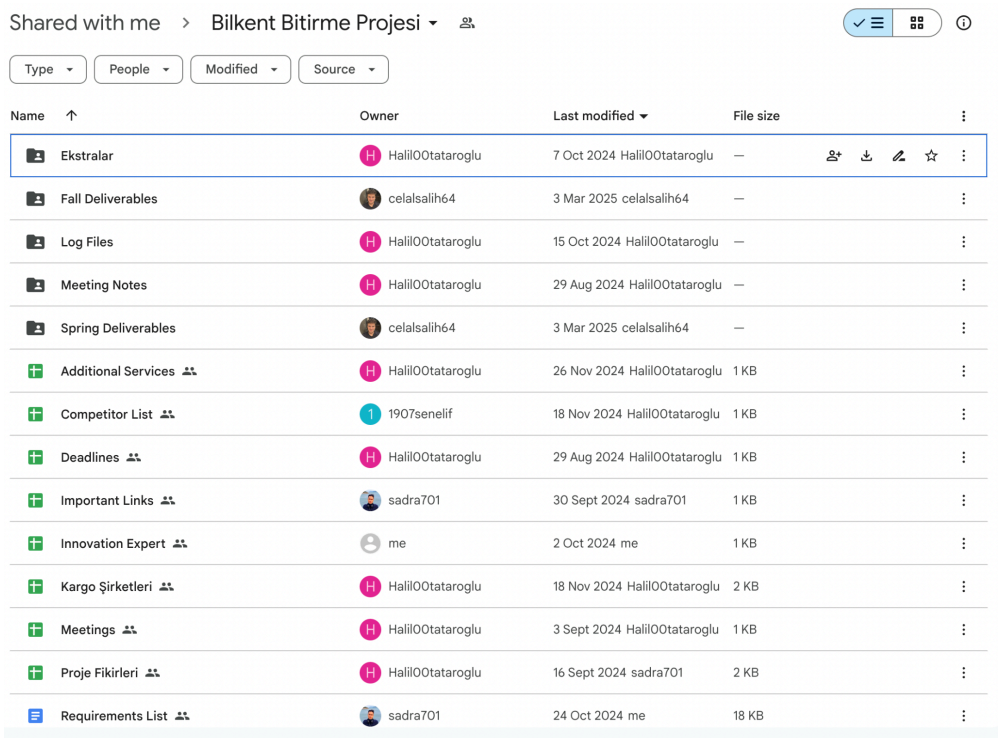


Figure 3: Google Drive

7.3 Taking lead role and sharing leadership on the team

- **Celal**: Leads the DevOps team, providing continuous integration, deployment, and testing processes. As the Scrum Master, he is responsible for sprint planning, stand-up meetings, and team coordination to maintain an efficient development workflow.

- **Halil**: Leads the business and marketing team, managing communication with cargo companies, partnership strategies, and external API integrations. He also manages the communication between development team and partner companies.

- **Sadra**: Leads the backend team, architecting the system's core infrastructure, data communication, and backend logic. He manages the tech stack, helps teammates on backend development best practices, and contributes to project management by coordinating tasks effectively.

- **Elif**: Leads the business and marketing team. She focuses on specialized campaigns, corporate collaborations, and promotional strategies. Additionally, she is responsible for the full-stack development, contributing to frontend and backend components.

- **Eslim**: Leads the frontend team, managing UI/UX design, frontend implementation, and user experience optimization. She also manages the DevOps processes for frontend deployment. Additionally, she is the main contact between the team, the innovation expert, and the advisor.

8. Glossary

AHP (Analytic Hierarchy Process): A structured decision-making framework used to prioritize alternatives based on multiple criteria.

API (Application Programming Interface): A set of rules and protocols that allow different software applications to communicate.

Authentication: The process of verifying a user's identity before granting access to a system.

Authorization: The process of determining what a user is allowed to do within the system.

Backend: The server-side components of an application responsible for processing data, managing logic, and handling API requests.

CI/CD (Continuous Integration/Continuous Deployment): A development practice that enables frequent and automated software testing and deployment.

CLV (Customer Lifetime Value): A metric that predicts the total revenue a business can earn from a single customer.

Cloud Hosting: Hosting services provided on remote servers via cloud computing platforms like AWS, Google Cloud, or Azure.

CRUD (Create, Read, Update, Delete): Basic operations that can be performed on database records.

DevOps: A set of practices that combine software development and IT operations to improve efficiency, shorten development cycles, and enhance system reliability.

Docker: A containerization technology used to package applications with their dependencies for efficient deployment and scaling.

Encryption: The process of converting data into a secure format to prevent unauthorized access.

FastAPI: A modern, high-performance web framework for building APIs with Python.

GDPR (General Data Protection Regulation): A European Union regulation governing data privacy and security.

JWT (JSON Web Token): A compact, URL-safe method for securely transmitting authentication information between parties.

KVKK (Kişisel Verilerin Korunması Kanunu): The Turkish equivalent of GDPR, regulating personal data protection and privacy.

LLM (Large Language Model): AI-powered models capable of understanding and generating human-like text, used in chatbot interactions.

Microservices Architecture: A software design pattern where an application is composed of loosely coupled, independently deployable services.

OAuth: An open standard for authentication that allows secure access without sharing user credentials.

PBAC (Policy-Based Access Control): A security model that controls access to resources based on predefined policies.

PgBouncer: A lightweight PostgreSQL connection pooler used to optimize database performance.

PostgreSQL: A powerful open-source relational database management system used for data storage.

RAG (Retrieval-Augmented Generation): An AI-based approach that enhances chatbot responses by retrieving and incorporating relevant information.

Redis: An in-memory data structure store used for caching to improve system performance.

RBAC (Role-Based Access Control): A security mechanism that restricts access based on predefined user roles.

React: A JavaScript library for building fast and interactive user interfaces for web applications.

React Native: A framework for building mobile applications using JavaScript and React.

SaaS (Software as a Service): A cloud-based software delivery model where users access applications via the internet.

Scalability: The ability of a system to handle increasing workloads and grow in response to user demand.

SME (Small and Medium-sized Enterprises): Businesses with limited scale compared to large corporations, often targeted as users in Yolla.

TLS (Transport Layer Security): A cryptographic protocol that ensures secure communication over networks.

UI (User Interface): The visual and interactive elements of an application that users interact with.

UX (User Experience): The overall experience users have while interacting with a system, focusing on ease of use and efficiency.

9. References

[1] “Geliver - Akıllı Kargo Pazaryeri,” *Geliver.io*, 2024. https://geliver.io/ (accessed Nov. 20, 2024).

[2] “Kargonomi | Kargo Göndermenin En Ekonomik Hali,” *Kargonomi*, Nov. 20, 2024. https://www.kargonomi.com.tr/ (accessed Nov. 20, 2024).

[3] “LinkedIn Login, Sign in | LinkedIn,” *LinkedIn*, 2024. https://www.linkedin.com/company/grakgrak-cargo/about/ (accessed Nov. 20, 2024).

[4] “LinkedIn Login, Sign in | LinkedIn,” *LinkedIn*, 2024. https://www.linkedin.com/company/kolibu/about/ (accessed Nov. 20, 2024).

[5] T. He, W. Ho, C. Lee Ka Man, and X. Xu, “A fuzzy AHP based integer linear programming model for the multi‐criteria transshipment problem,” *The International Journal of Logistics Management*, vol. 23, no. 1, pp. 159–179, May 2012, doi:https://doi.org/10.1108/09574091211226975.