**Online Charging System**

**Project**

**PIXCELL**

**Copyright of i2i Systems Turkey 2024**

The copyright in this work is vested in i2i Systems Turkey and the information contained herein is confidential.   
This work (either in whole or in part) must not be modified, reproduced, disclosed or disseminated to others or used for purposes other than that for which it is supplied, without the prior written permission of i2i Systems Turkey. If this work or any part hereof is furnished to a third party by virtue of a contract with that party, use of this work by such party shall be governed by the express contractual terms between the i2i Systems Turkey which is a party to that contract and the said party.

1 ENTRANCE 3

1.2. Used Technologies3

1.3. Project Stakeholders…………………………………………………………………………………….4

2 FRONTEND 7

2.2. Android App Frontend8

2.3. IOS App Frontend………………………………………………………………………………..…….11

2.4. Web App Frontend13

2.5. SMS App Frontend…………………………………………………………………………………….15

3 BACKEND 18

4 DATABASE20

5 TEST 21

6 DIAGRAMS 22

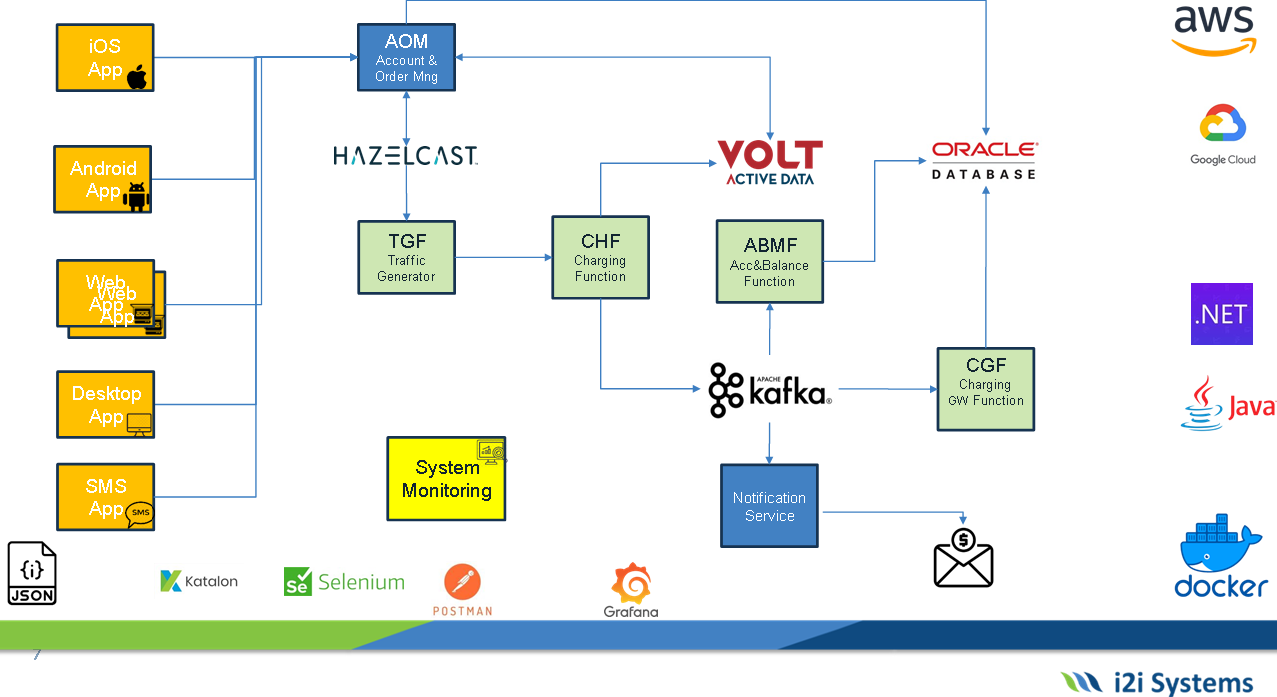
7 USE CASE 25

8 APPENDIX28

ABBREVIATIONS…………………………………………………………...………29

1.Entrance

In this project, a prototype of the Online Charging System will be created. The name of the prototype is designated as "PIXICELL". OCS is a specialized communication function that allows a service provider to charge users for services in real-time. OCS manages the subscriber's account balance, rating, charging transaction control, and correlation. With OCS, a telecom operator ensures the enforcement of credit limits and the authorization of resources on a per-transaction basis.

To better understand the project, the Project High Level Diagram is provided below.

* The project consists of backend, frontend and database parts.

**1.2.** **Used Technologies**

**Frontend:** Java,

**Backend:** HazelCast, Kafka

**Database:** VoltDB, Oracle SQL

**Test:** Selenium

* Installations were made on the latest versions. It is built on Docker cloud.

**1.3.** **Project Stakeholders**

|  |  |  |
| --- | --- | --- |
| PROJE YÖNETİCİSİ KİMDİR, PROJE YÖNETİCİSİ NASIL OLUNUR? | **Project Manager** | Semih Bilal ONAR |
| Service Designers Vs Business Analysts | **Analyst/Desinger** | Beyza BARMAN |
| Oracle Database Architecture Explained | by Nii Tetteh Adjirackor | Medium | **Oracle DB** | Emre ÇOLTU |
| VoltDB - Revision #3 - Database of Databases | **Volt DB** | Berkay ŞAHİN |
| Brand Assets | Hazelcast | **Hazelcast** | Barış ZIRHLI |
| Apache Kafka | **Kafka** | Oytun YELDAN |
| Kayseri Web Tasarım | **Web App** | Bilge GÖKKAYA |
| iOS App Store'da öne çıkmanızı sağlayacak 4 ipucu - Mobil Uygulama Yap,  Yaptır ve Para Kazan | Mobiroller | **IOS App** | Engincan GÜNDÜZ |
| Android Nedir? - Technopat | **Android App** | Ayşe Elif ÇELİK |
| Google'ın SMS 2.0 Üzerine Olan Hamleleri | **SMS App** | Tan Emre KAR |
|  | **CHF/OCS** | Barbaros UZUNLAR |
|  | **ABMF/FS** | Ömer Faruk KIZILDAMAR |
|  | **TGF** | Muhammed Taha BAYRAK |
| Push Notification Best Practices: 7 Questions Designers Should Ask | Toptal® | **NF** | Emrullah DONSAK |
| Software Testing: Complete Guide 2019 |Professionalqa.com | **TEST (1)** | Barış ZIRHLI |
| Software Testing Methods - Atlas Computer Systems Ltd | **TEST (2)** | Mustafa KARTAL |
| What are the criteria for choosing a software test tool | Informatica e  Ingegneria Online | **TEST (3)** | Safiye ŞAHİN |

**2. Frontend**

The project will primarily include the following pages:

• Login Page

• Register Page

• Forgot Password Page

• Remaining Usage Page (Home Page)

Login Page:

• On the Login page, registered users log in to the system with their phone number and chosen password.

Register Page:

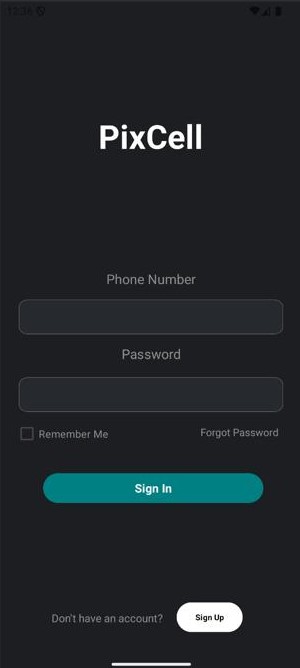
• On the Register page, users who have not yet registered in the system sign up with their first name, last name, phone number, email, and a password they choose. After registration, users are directed to the package selection page. Once the package is selected, they are taken to the homepage.The packages include internet, minutes, and SMS details.

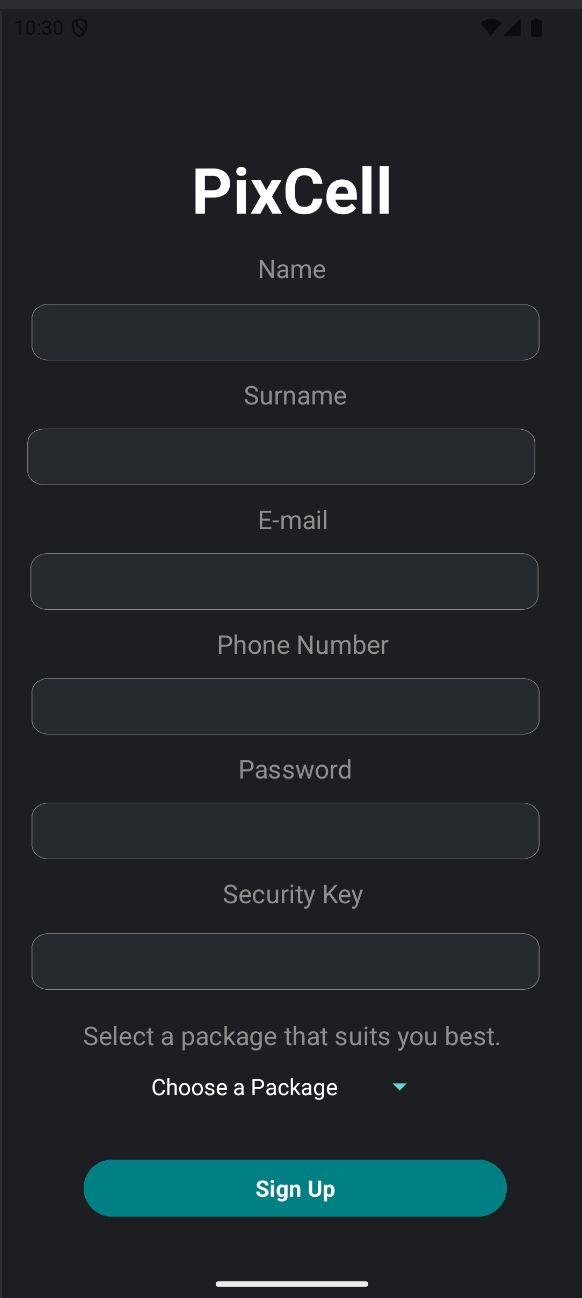
Forgot Password Page:

• On the Forgot Password page, users who cannot remember their passwords enter their phone numbers in the relevant field and create a new password via an SMS sent to their phone number.

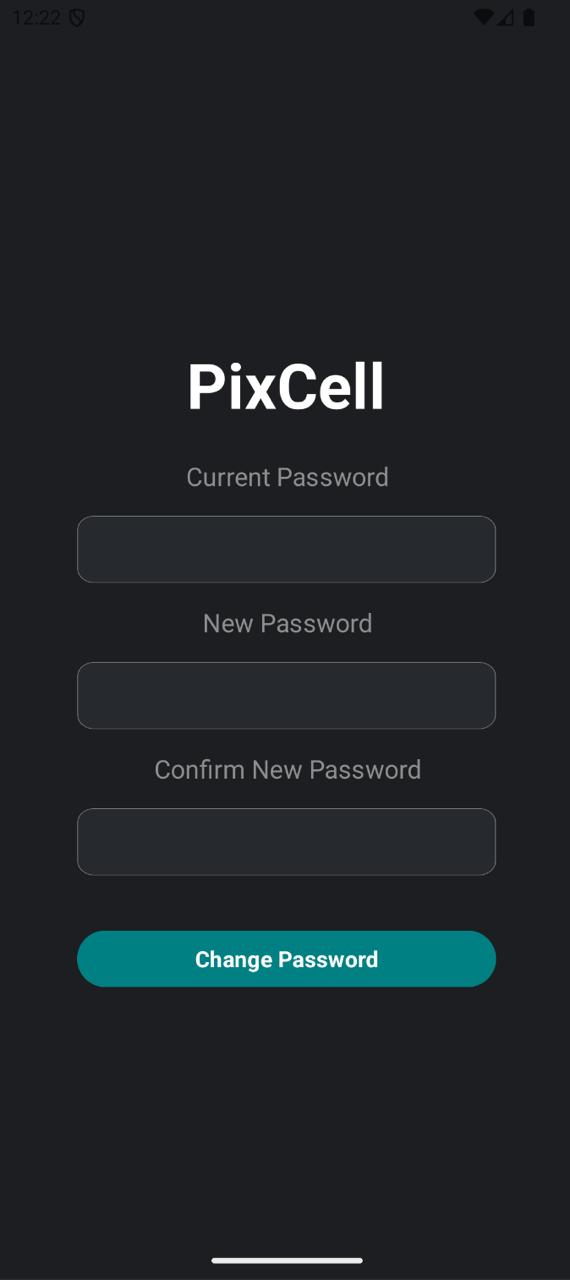
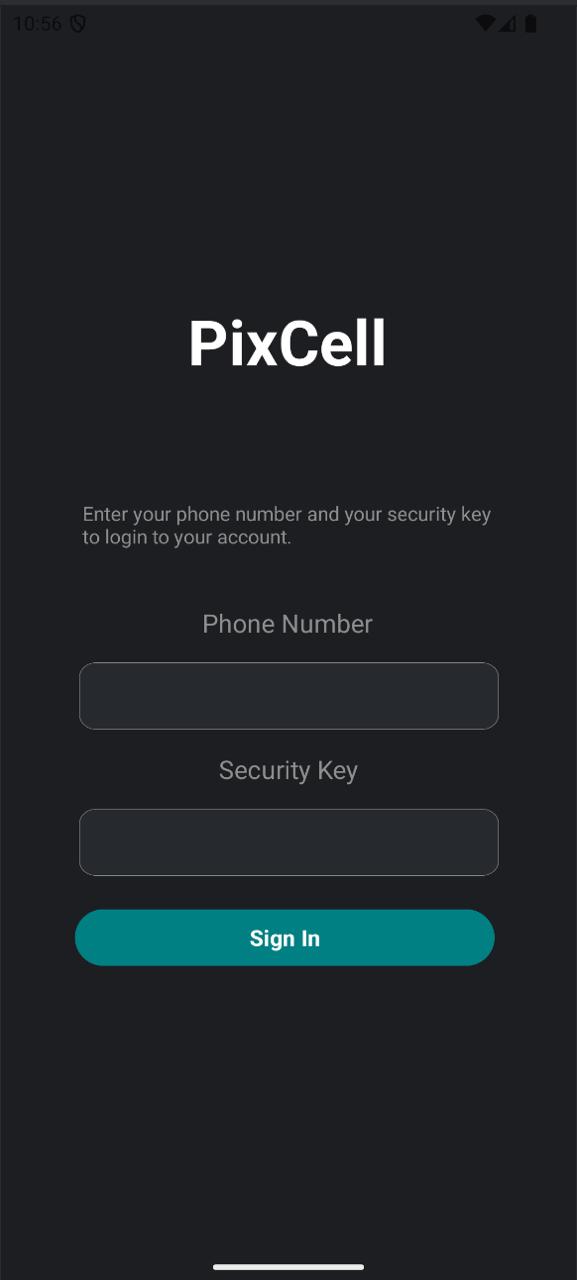
Remaining Usage Page (Home Page):

• On the Remaining Usage page, users can view the minutes, SMS, and internet details of the packages they have purchased and see how much they have spent during the month. In addition to the remaining usage rights, billing information is also displayed on this page.

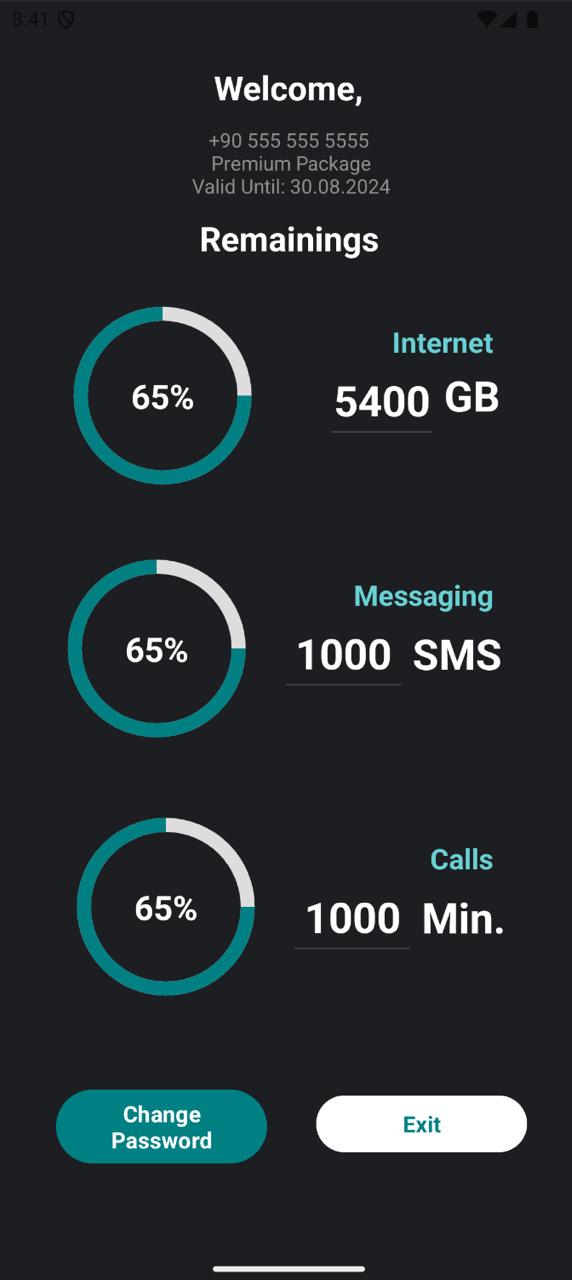
**2.2 Android App Frontend**



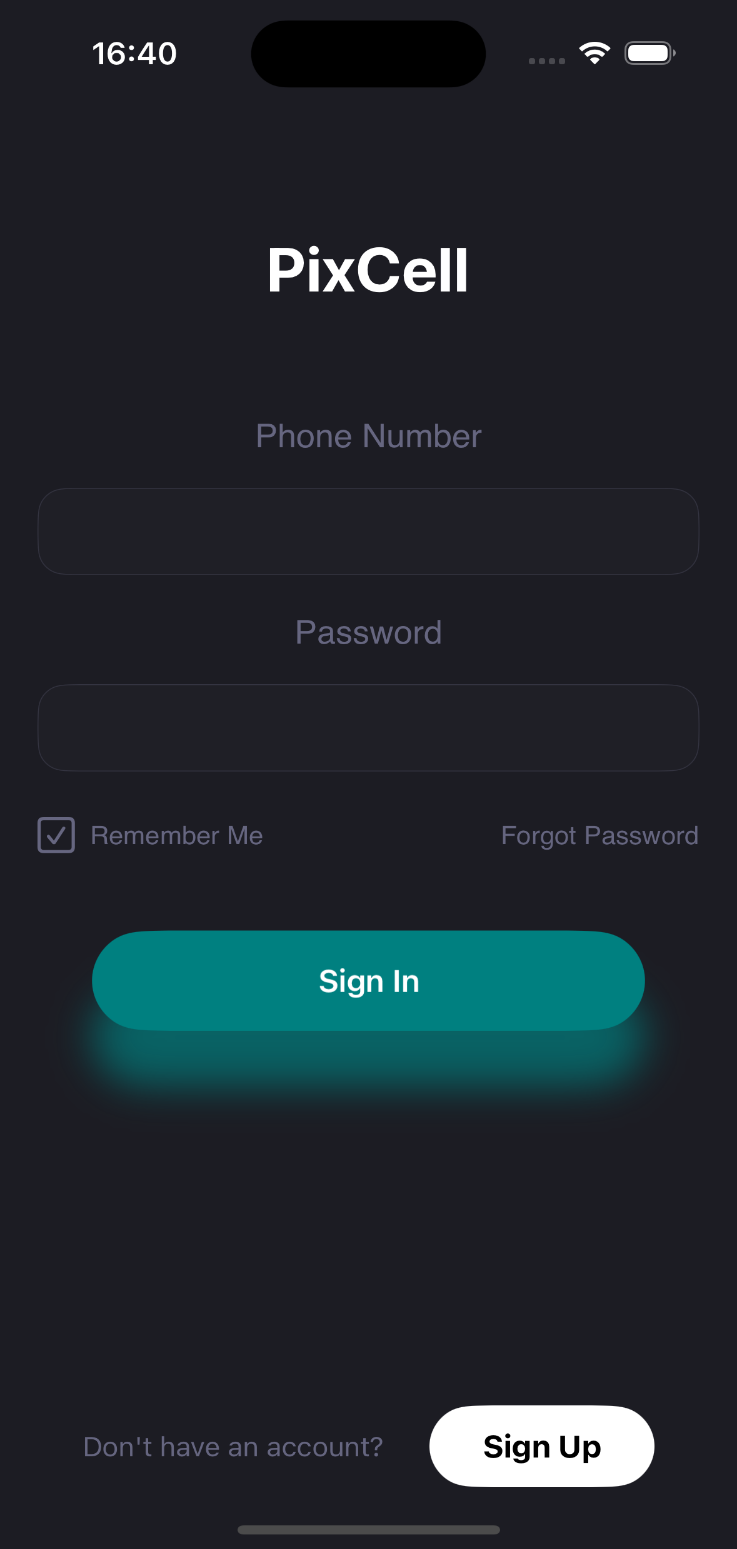
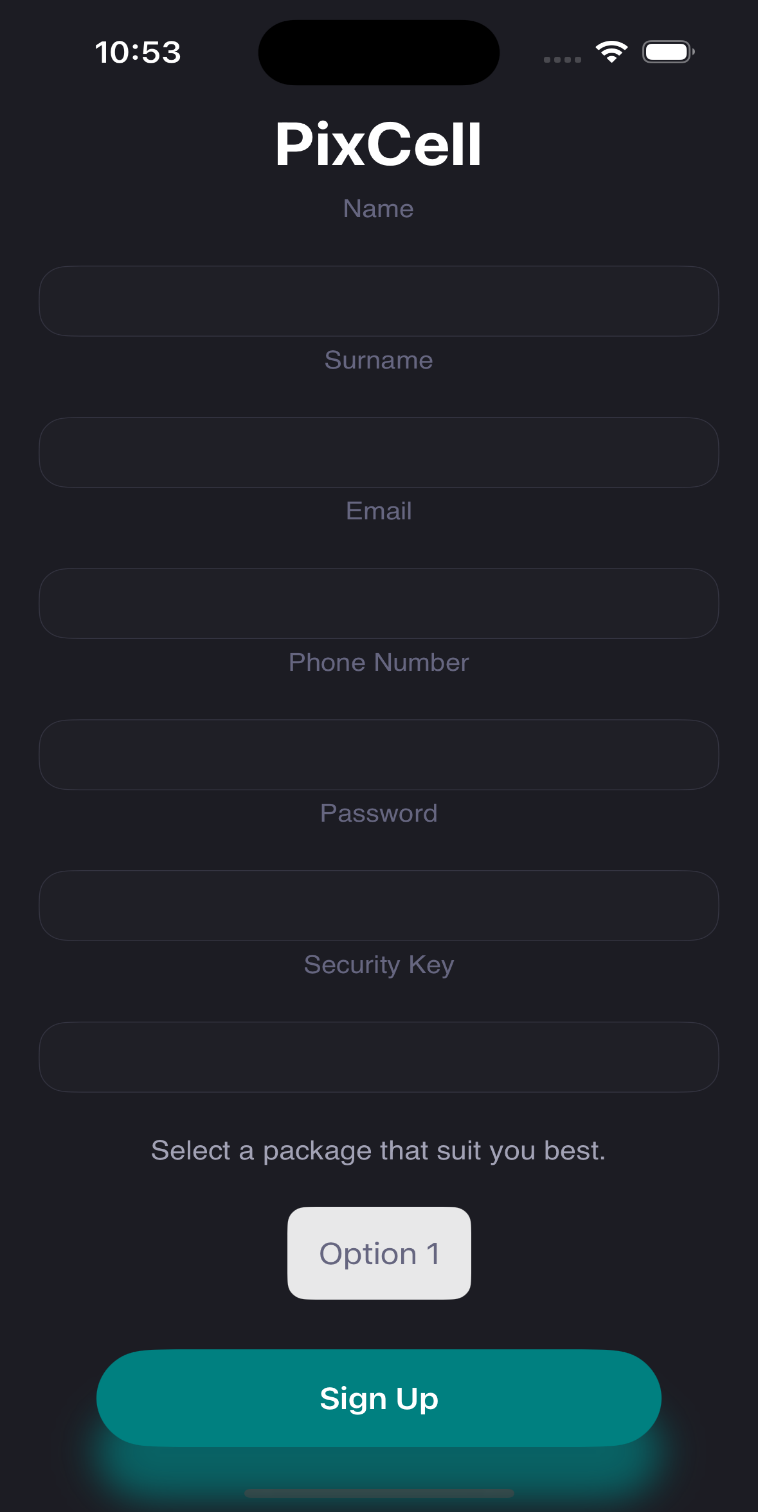
**Picture 1: Android Login Page** **Picture 2: Android Register Page**



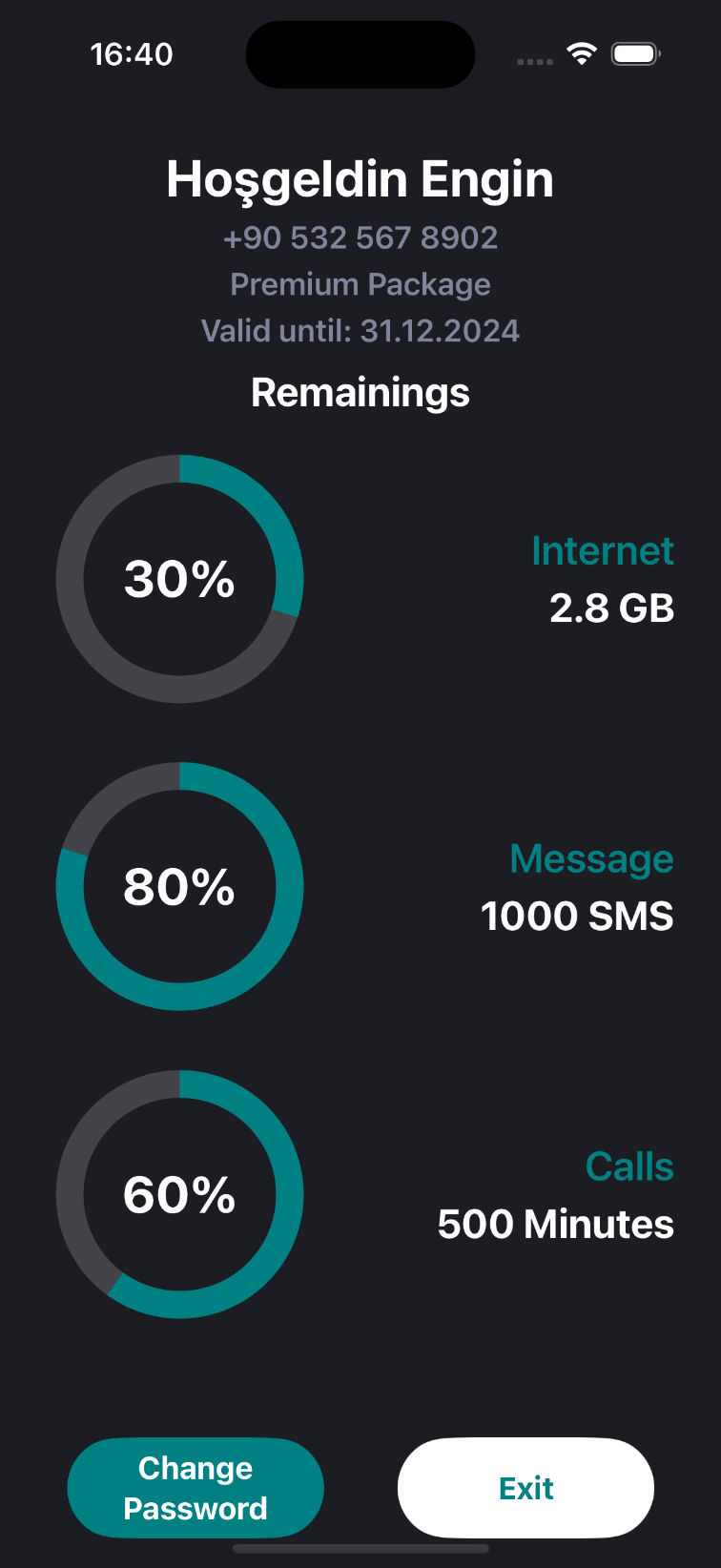
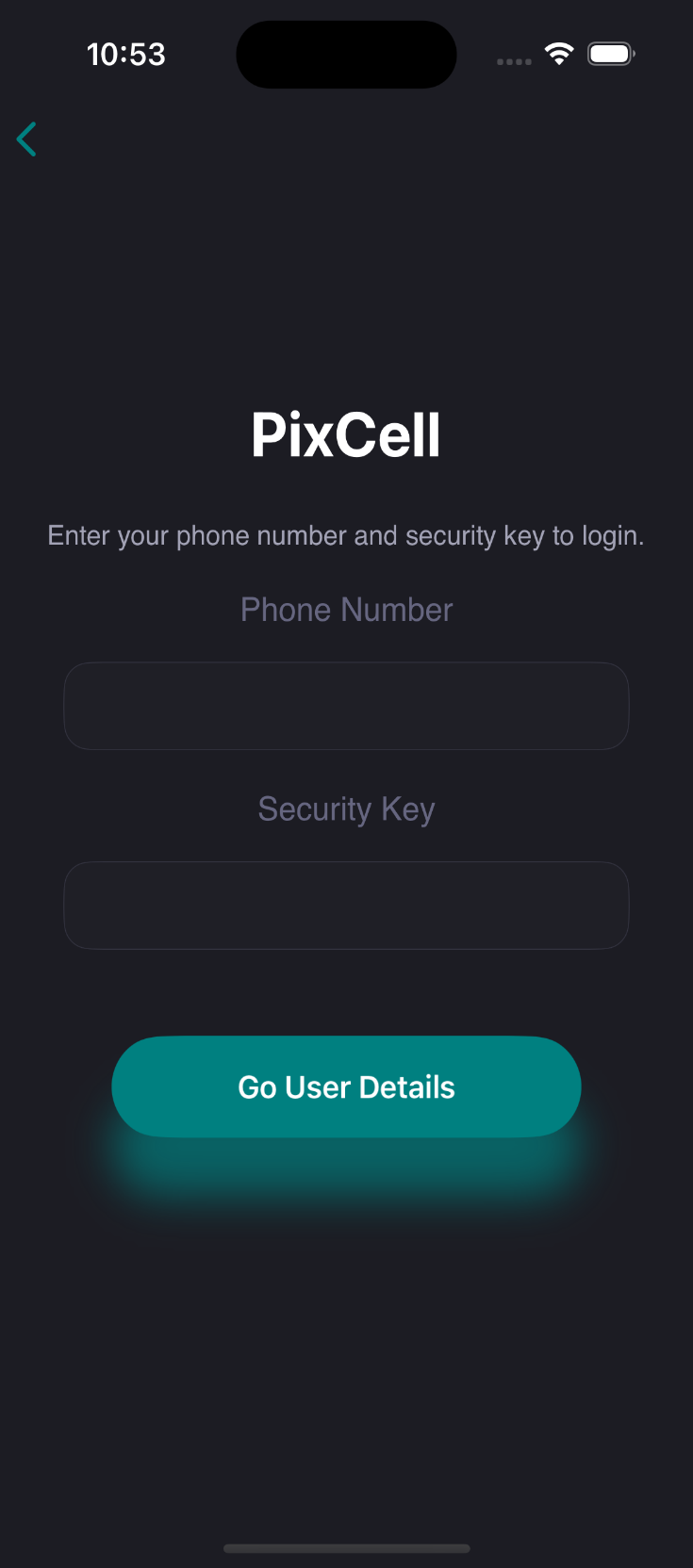
**Picture 3: Android Forgot Password Page Picture 4: Android Password Reset Page**



**Picture 5: Android Home Page**

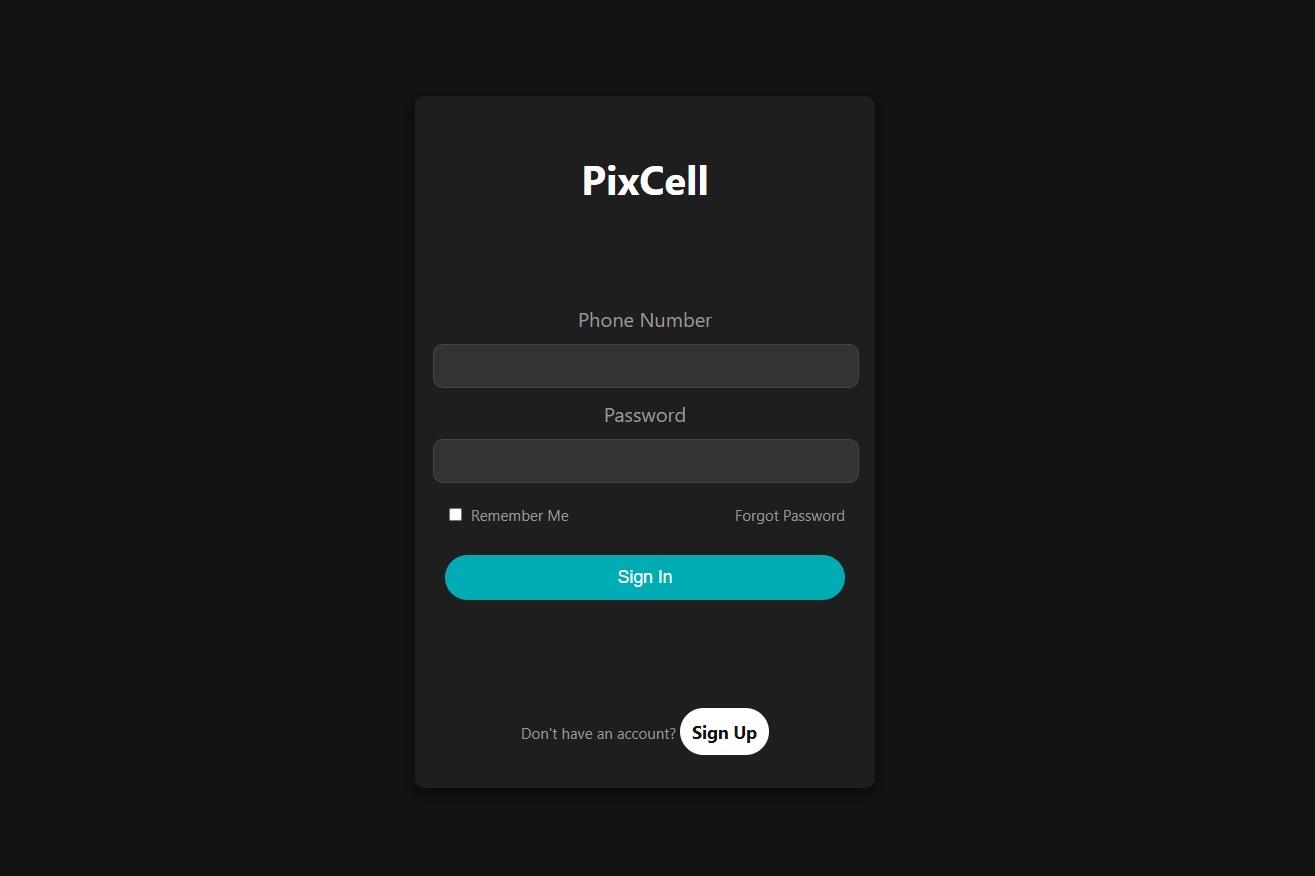
**2.3 IOS App Frontend**

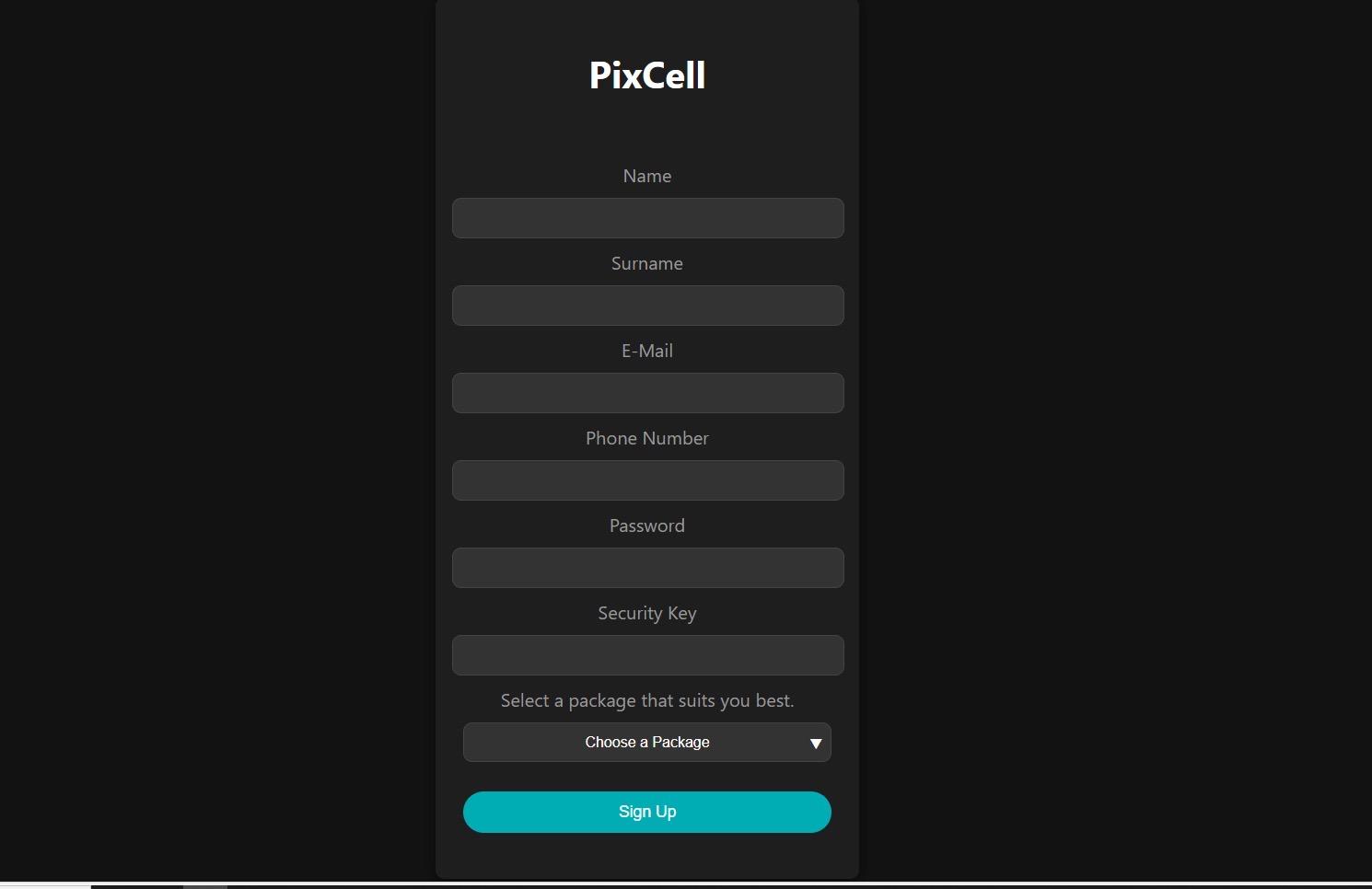
**Picture 6: IOS Login Page Picture 7: IOS Sign Up Page**



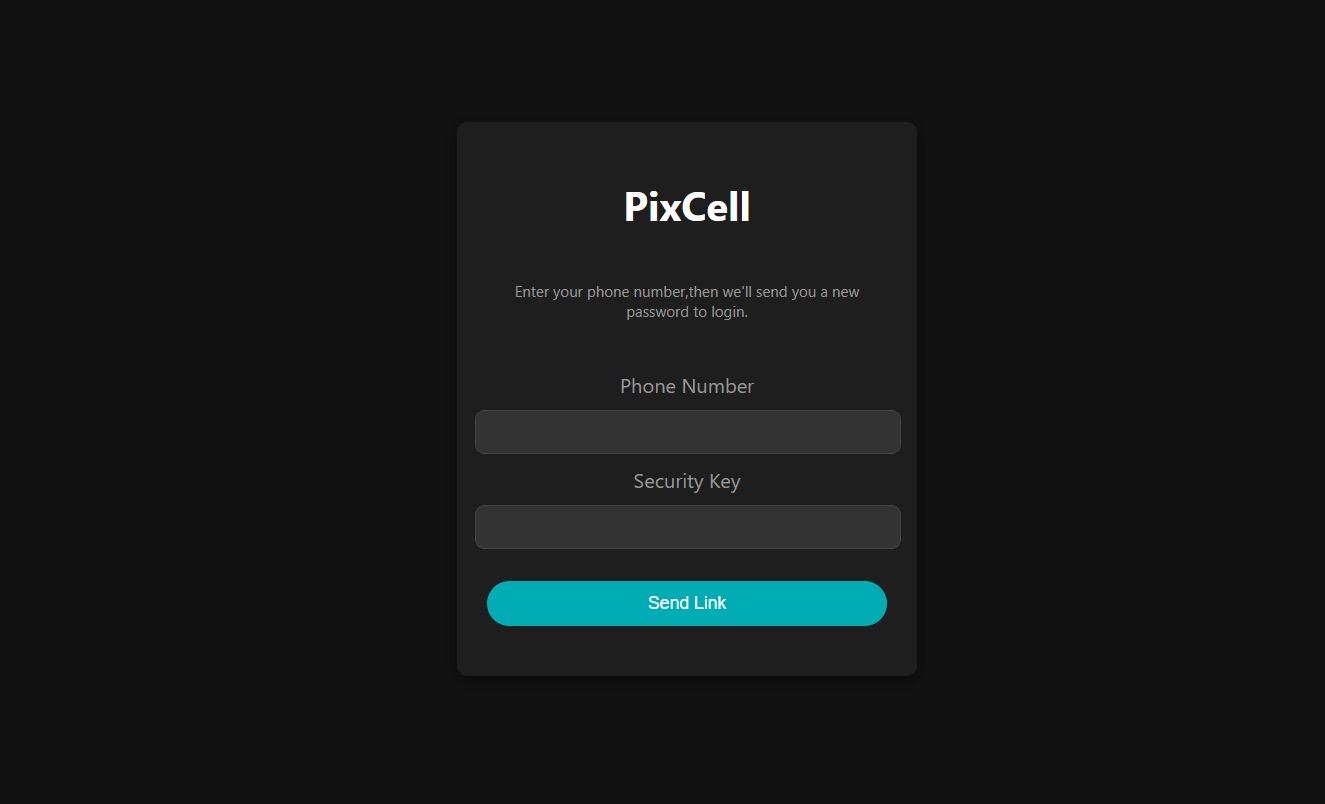
**Picture 8: IOS Forgot Password Page Picture 9: IOS Home Page**

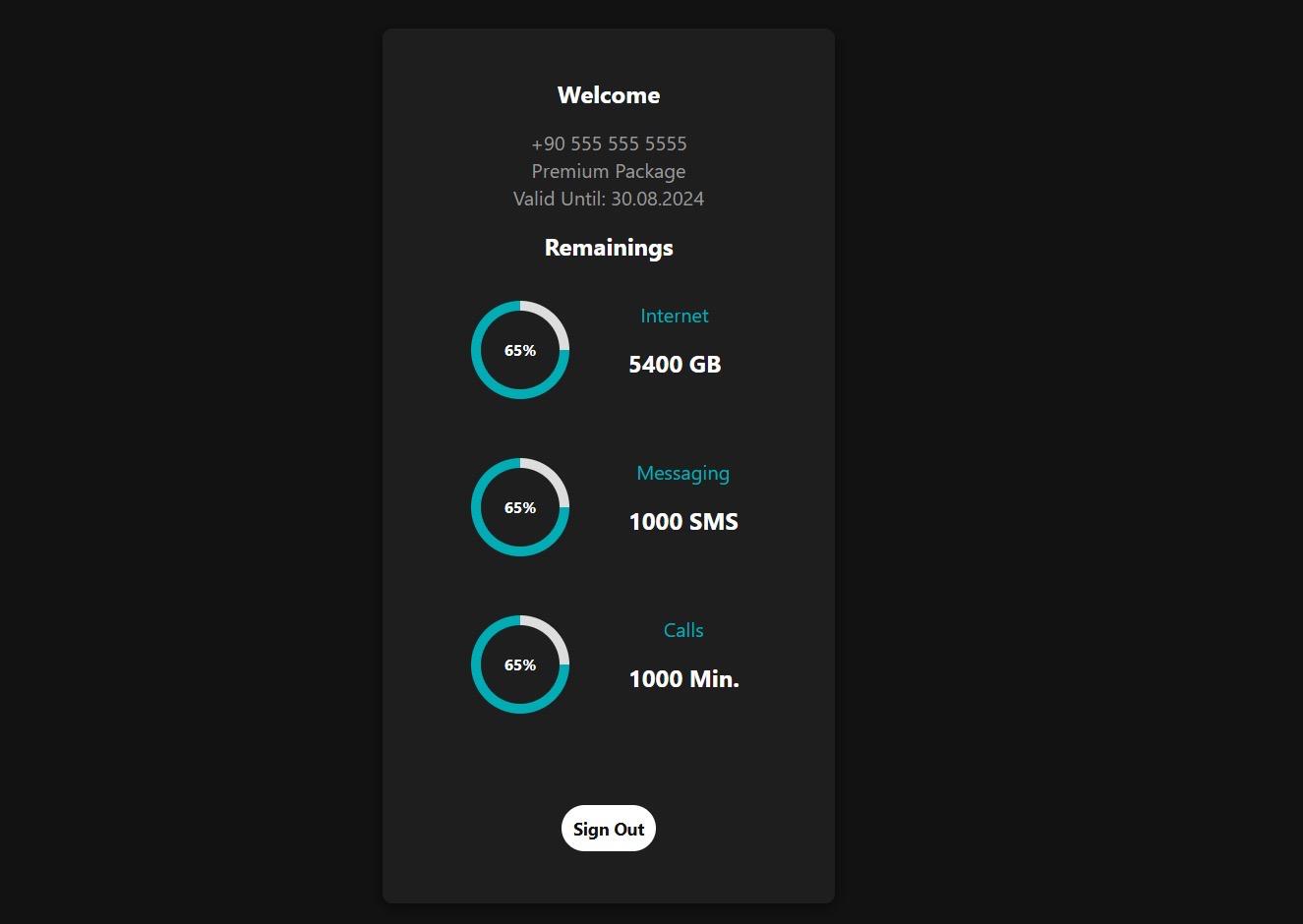
**2.4 Web App Fronted**



**Picture 10: WEB Login Page**

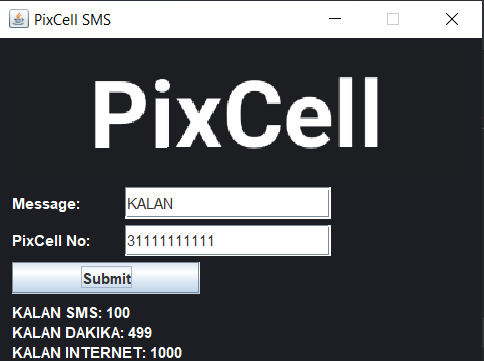
**Picture 11: WEB Sign Up Page**

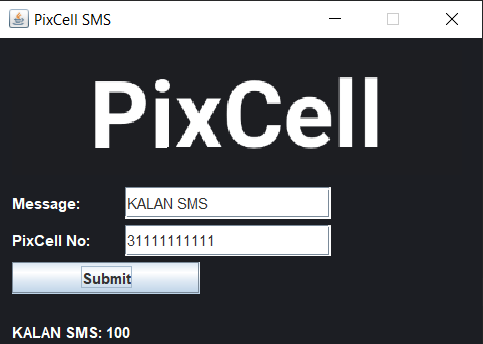


**Picture 12: WEB Forgot Password Page**

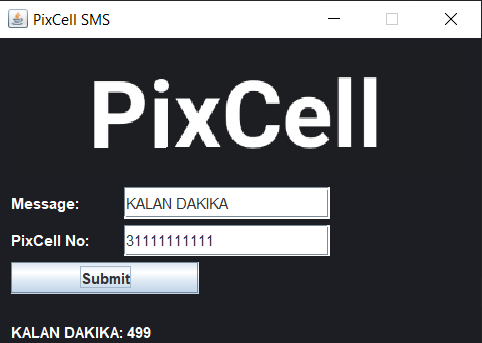
**Picture 13: WEB Home Page**

**2.5 SMS App Frontend**

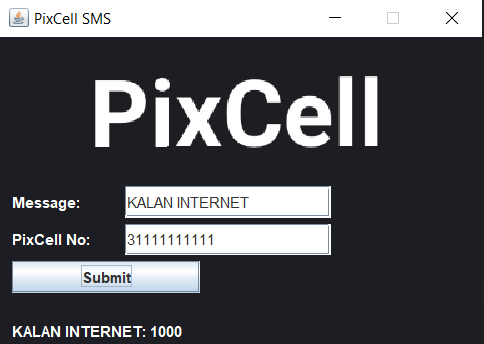


**Picture 14: SMS “KALAN” Page**

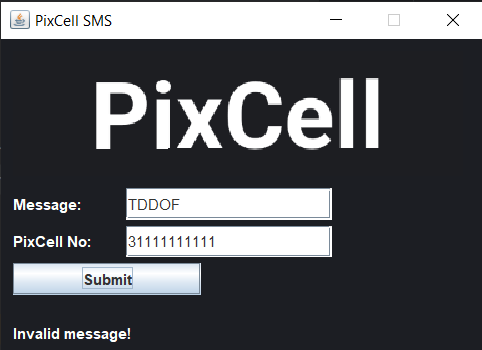
**Picture 15: SMS “KALAN SMS” Page**



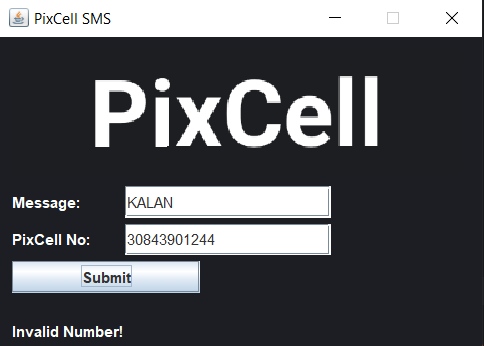
**Picture 16: SMS “KALAN DAKIKA” Page**



**Picture 17: SMS “KALAN INTERNET” Page**



**Picture 18: SMS Wrong Message Page**



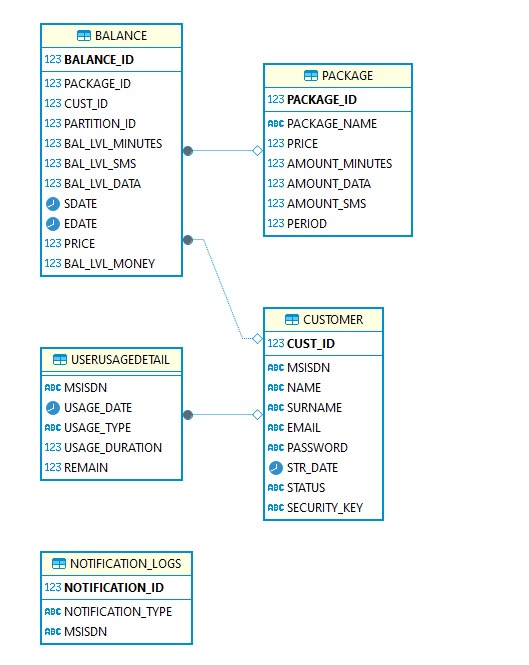
**Picture 19: SMS Wrong Phone Number Page**

**3.Backend**

* **Hazelcast**: It was installed on the cloud, and GET and PUT codes were written. A base was created, and the connection was established.
* **Kafka:** After creating a cloud account, a virtual machine was set up through the cloud. Kafka, Kafdrop, and Zookeeper were installed using Docker. Kafdrop was accessed through a web browser, and three topics were created via the Kafdrop interface. Remote access was established by connecting to CHF, ABMF, CGF, and NF. Classes written in JSON format were imported, enabling access with other technologies.
* **Web App**: The data has been fetched from the API, and the code has been written.
* **IOS App:** The data has been fetched from the API, and the code has been written.
* **Android App**: The data has been fetched from the API, and the code has been written.
* **SMS App:** We retrieve data from the database to find out how many SMS messages are left by using the code 5555.
* **CHF/OCS:** First, the general OCS schema and files were created. After adding the latest versions of the dependencies deemed necessary via Maven, classes were written. After establishing the Akka connection, VoltDB and Kafka integrations were completed. The system retrieves data from the database, modifies it according to the tariff, and sends it back. The BalanceCalculations class is used for calculating charges.
* **ABMF/SF:** It retrieves the account and balance information coming from iOS, Android, and web, from Kafka and sends it to Oracle.
* **AOM/MW:** Entities have been created. Data is being retrieved from the repository. Connections have been established with Oracle Database, VoltDB, and Hazelcast. The system is controlled through iOS, Android, and web connections. It is written in the Java programming language and coded in the IntelliJ environment.
* **TGF:** It is a simulation. In real life, actual people will provide random traffic. This project has been manually written.
* **CGF**: Manual connection with Oracle has been established. It fetches the package information used by the user from Kafka and inserts it into the balance table in the Oracle database. In short, it inserts the user's usage data into the table. It has a connection with Kafka. It manipulates and changes the modified values.
* **NF:** It sends an email to the user when their internet data, SMS, or minutes decrease.

**4.Database**

* **Oracle Database:** After creating the necessary entities and attributes for the database, relationships between the entities were established. To ensure data integrity, procedures, functions, and triggers were created, facilitating the fast processing of data. Functions responsible for quick calculations were packaged and delivered to the relevant units. A connection was established between AOM and CGF. The customer's personal information and package details are stored here.



**Picture 15**

* **VoltDB:** Docker has been installed on Google Cloud, followed by the installation of VoltDB. Firewall rules were modified to ensure proper connectivity. Tables created in Oracle Database were utilized. After writing procedures in Java, the system was connected. Connections were established with AOM and CHF. VoltDB is used to monitor the remaining allowances of a customer's minute, SMS, and internet packages in real-time. It is faster than the Oracle database.

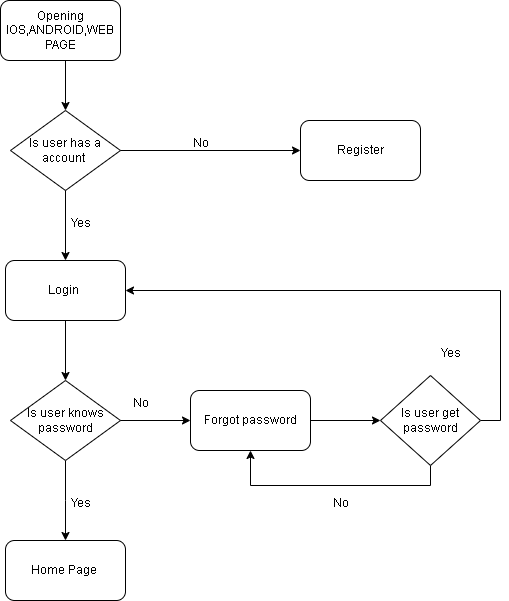
**5.Test**

The developers who coded the interfaces have conducted the tests.

|  |  |
| --- | --- |
| **Test 1** | ABMF,CGF,Notification |
| **Test2** | VoltDB, Hazelcast, Oracle DB, Kafka, AOM |
| **Test3** | Android |
| **Test4** | IOS |
| **Test5** | WEB |
| **Test6** | CHF |

6.Diagrams

* The flowchart for the user's navigation between Register, Login, Forgot Password, and Homepage is provided in Figure 1.



**Figure 1**

* Flowchart diagram showing how users register.

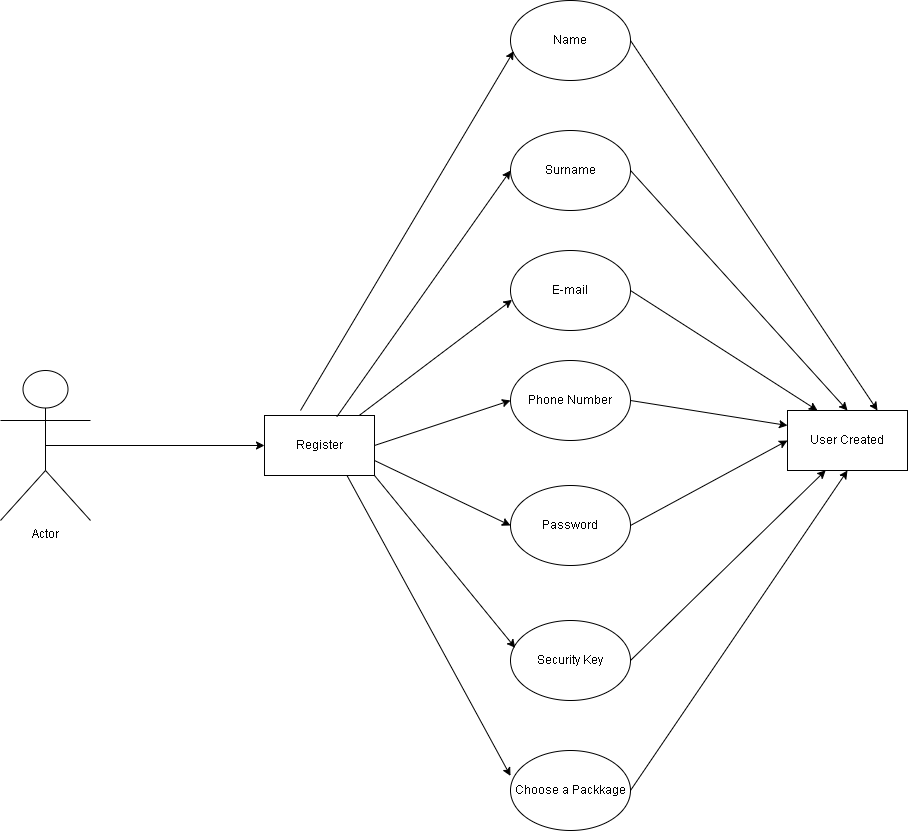


Figure 2

* Flowchart diagram showing user login and password reset scenarios.

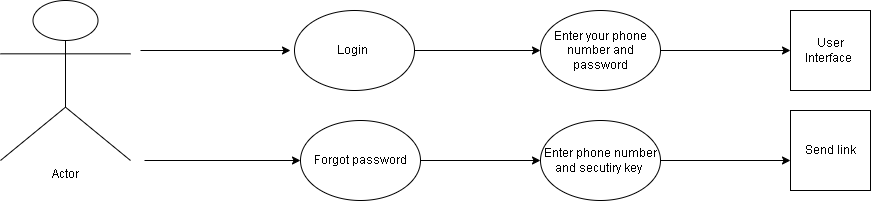
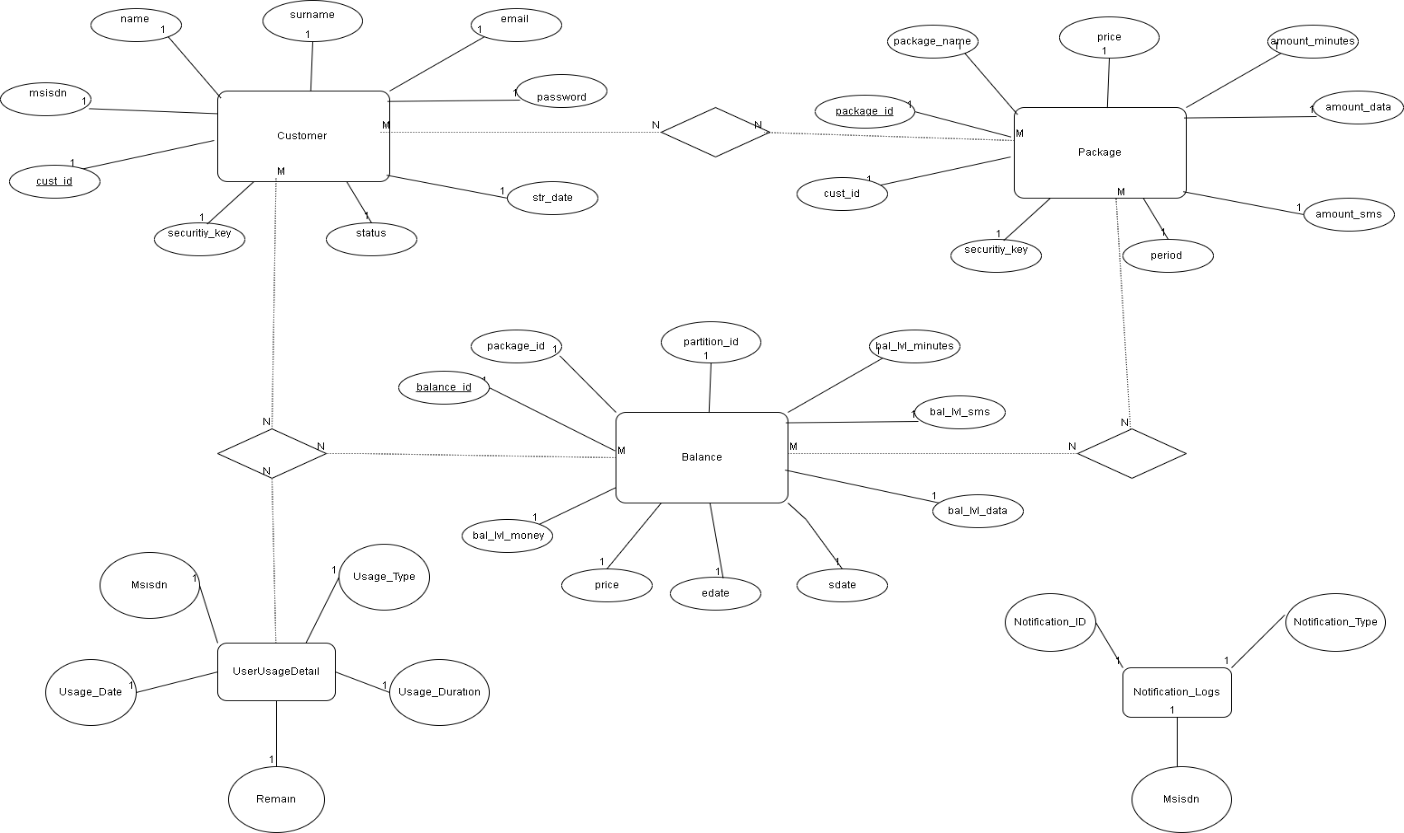


Figure 3

* It started with database design. ER Diagram is mentioned below.



**Figure 4**

7. User Case

|  |  |
| --- | --- |
| **Summary** | Successful Login |
| **Impacted System Components** | CHF, TGF, AOM, HAZELCAST, VOLTDB, KAFKA |
| **Pre-conditions** | In order for the subscriber to log into the system successfully, users must enter their information correctly. |
| **Post-conditions** | The user continues to log in to the system with the new password. |
| **Main Success Scenario** | 1-User enters their phone number and password information on the Login Screen and clicks Login button.  2-A check is made to match the user data in the database with the  entered information. If the information is correct, the user logs into the system. |
| **Alternate Flows** | Unsuccessful Login |
| **Exceptions/Errors** | If the user enters the wrong phone number and/or password, they cannot log into the system. |

Use Case 1

|  |  |
| --- | --- |
| **Summary** | Unsuccessful Login |
| **Impacted System Components** | CHF, TGW, AOM, HAZELCAST, VOLTDB, KAFKA |
| **Pre-conditions** | In order for the subscriber to log into the system successfully, users must enter their information correctly. |
| **Post-conditions** | The subscriber cannot log in to the system and receives a warning message. |
| **Main Success Scenario** | 1-User enters their phone number and password information on the Login Screen and clicks Login button.  2-A check is made to match the user data in the database with the entered information. If the information is incorrect, the user cannot log in to the system and receives a warning message. |
| **Alternate Flows** | Successful Login |
| **Exceptions/**  **Error rs** | If the information is correct, the user logs into the system. |

Use Case 2

|  |  |
| --- | --- |
| **Summary** | Forget Password |
| **Impacted System Components** | CHF, TGW, AOM, HAZELCAST, VOLTDB, KAFKA |
| **Pre-conditions** | The user must be registered in the system. |
| **Post-conditions** | The user logs into the system with the new password. |
| **Main Success Scenario** | 1-The user clicks the "Forgot Password" button on the Login Screen.  2-The user enters their e-mail address.  3-The user enters their security key  4-The security code will be sent to e-mail addresses of the user  5-The user will enter the code from the e-mail and click the change password button. |

Use Case 3

|  |  |
| --- | --- |
| **Summary** | Change Password |
| **Impacted System Components** | CHF, TGW, AOM, HAZELCAST, VOLTDB, KAFKA |
| **Pre-conditions** | The user must be entering security code. |
| **Post-conditions** | The user logs into the system with the new password. |
| **Main Success Scenario** | 1. The user will enter their new password. 2. The user will re-enter the new password. 3. The user should click on the change my password button. |

Use Case 4

|  |  |
| --- | --- |
| **Summary** | Sign Up |
| **Impacted System Components** | CHF, TGW, AOM, HAZELCAST, VOLTDB, KAFKA |
| **Pre-conditions** | The user must have an account. |
| **Post-conditions** | User successfully registers in the system |
| **Main Success Scenario** | 1. User registers to he/she system with her/his name, surname, password, phone number, e-mail and security key.      1. User must select one of the available packages. 2. The data registered by the user is saved in the database. |
| **Alternate Flows** | Unsuccessful Create Costumer |
| **Exceptions/**  **Errol rs** | 1. In case of missing or incorrect data, the subscriber will not be able to register. 2. user must select an available package. The user cannot register without selecting an existing package. |

Use Case 5

1. Appendix

## 8.2 Document Control

### References

Online Charging Sistemi Projesi.docx

## Change Control § Distribution

|  |  |
| --- | --- |
| **Owner** | Beyza BARMAN |
| **Reviewer** | Hacı Mehmet ATILGAN  Mennan TEKBİR |
| **Approved By** | Hacı Mehmet ATILGAN  Mennan TEKBİR |
| **Distribution** | Managers  Relevant Departments  Business/System Analysts  Solution/System Architects  Project Manager |
| **File Name** | PİXCELL\_SRS\_DOCUMENT.docx |

## 8.4Version History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Change Descriptions** | **Author** | **Date** |
| 1.0.0 | Initial Version | Beyza BARMAN | 15.08.2024 |
| 1.1.0 |  |  |  |

## 8.5 Approvals

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Title** | **Date** | **Signature** | **Version** |
| Hacı Mehmet ATILGAN | Solutions Architect | 15.08.2024 | 15.08.2024 | 1.0.0 |
| Mennan TEKBİR | Solutions Architect | 09.08.2024 |  | 1.0.0 |

ABBREVIATIONS

|  |  |
| --- | --- |
| OCS | Online Charging Solution |
| API | Application Programming Interface |
| ABMF | Acc&Balance Function |
| AOM | Account & Order Mng |
| CGF | Charging GW Function |
| CHF | Charging Function |
| TGF | Traffic Generator |
| NF | Notfication Service |
| BSS | Business Support System |
| DB | Database |
| GUI | Graphical User Interface |
| SAS | System/Solution Architectural Specification |
| SRS | System/Solution requirements Specification |
| CDR | Call Data Record |
| WA | Workaround |
| SMS | Short Message Service |
| MMS | Multimedia Messaging Service |
| FTP | File Transfer Protocol |
| CCR | Credit Control Request |
| CCA | Credit Control Answer |
| 3GPP | Third Generation Partnership Program |
| UDR | Usage Data Record |
| MSISDN | Mobile Station International Subscriber Directory Number |
| IMSI | International Mobile Subscriber Identity |
| WS | Web Service |
| SOI | Service Oriented Interface |
| CMS | Content Management System |
| CRM | Customer Relations Management |
| OM | Order Management |