Capstone Project - Turf Booking Application

Name: Nirmit Surve
Batch: IDFC-JFS-B2

Table of Contents

Aim: Create a turf booking application with api and backend using react native and performance testing using Jmeter

Scope: Nowadays when we try to book a playground for some sports we need to call the owner of the playground and check with them if a slot is available. This is kind of a hassle for the customer. We need an interactive app for the customer to book a slot for turf without going through the process of contacting the turf owner.To solve the above issue I have designed and implemented a mobile app for booking a turf session. It consists of:

a) User App: This app is designed for the customer to book a slot efficiently and without any hassle. It has features like:
   1) Login
   2) Signup
   3) Homepage with about the turf i.e its amenities etc
   4) Map Location for easy navigation
   5) Contact Page
   6) View Bookings Page
   7) Booking feature with payment
   8) Rating the completed bookings

b) Admin App: This app is designed for the turf owner. It has features as follows:
   1) Admin Login
   2) Add a admin(only possible after signing in an existing admin)
   3) View all bookings for a particular date
   4) View customer queries and respond them with an email

Technologies Used:

a) **Development**:

1) React Native: React Native (also known as RN) is a popular JavaScript-based mobile app framework that allows you to build natively-rendered mobile apps for iOS and Android. The framework lets you create an application for various platforms by using the same codebase.
   React Native was first released by Facebook as an open-source project in 2015. In just a couple of years, it became one of the top solutions used for mobile development. React Native development is used to power some of the world's leading mobile apps, including Instagram, Facebook, and Skype. We discuss these and other examples of React Native-powered apps further in this post.
   There are several reasons behind React Native's global success.
   Firstly, by using React Native, companies can create code just once and use it to power both their iOS and Android apps. This translates to huge time and resource savings.
   Secondly, React Native was built based on React – a JavaScript library, which was already hugely popular when the mobile framework was released. We discuss the differences between React and React Native in detail further in this section.
   Thirdly, the framework empowered frontend developers, who could previously only work with web-based technologies, to create robust, production-ready apps for mobile platforms.

b) **Performance Testing:**

1) Jmeter: JMeter is a software that can perform load test, performance-oriented business (functional) test, regression test, etc., on different protocols or technologies.
   Stefano Mazzocchi of the Apache Software Foundation was the original developer of JMeter. He wrote it primarily to test the performance of Apache JServ (now called Apache Tomcat project). Apache later redesigned JMeter to enhance the GUI and to add functional testing capabilities.
   JMeter is a Java desktop application with a graphical interface that uses the Swing graphical API. It can therefore run on any environment / workstation that accepts a Java virtual machine, for example − Windows, Linux, Mac, etc.
   The protocols supported by JMeter are −
   - Web − HTTP, HTTPS sites 'web 1.0' web 2.0 (ajax, flex and flex-ws-amf)

- Web Services − SOAP / XML-RPC
- Database via JDBC drivers
- Directory − LDAP
- Messaging Oriented service via JMS
- Service − POP3, IMAP, SMTP
- FTP Service

c) API:

1) REST API: Representational State Transfer (REST) is an architectural style that defines a set of constraints to be used for creating web services. REST API is a way of accessing web services in a simple and flexible way without having any processing.
   REST technology is generally preferred to the more robust Simple Object Access Protocol (SOAP) technology because REST uses less bandwidth, simple and flexible making it more suitable for internet usage. It's used to fetch or give some information from a web service. All communication done via REST API uses only HTTP request.
   Working: A request is sent from client to server in the form of a web URL as HTTP GET or POST or PUT or DELETE request. After that, a response comes back from the server in the form of a resource which can be anything like HTML, XML, Image, or JSON. But now JSON is the most popular format being used in Web Services.

d) Database:

1) MongoDB: MongoDB is an open-source document-oriented database that is designed to store a large scale of data and also allows you to work with that data very efficiently. It is categorized under the NoSQL (Not only SQL) database because the storage and retrieval of data in the MongoDB are not in the form of tables.
   The MongoDB database is developed and managed by MongoDB.Inc under SSPL(Server Side Public License) and initially released in February 2009. It also provides official driver support for all the popular languages like C, C++, C#, and .Net, Go, Java, Node.js, Perl, PHP, Python, Motor, Ruby, Scala, Swift, Mongoid. So, that you can create an application using any of these languages. Nowadays there are so many companies that used MongoDB like Facebook, Nokia, eBay, Adobe, Google, etc. to store their large amount of data.

- The MongoDB database contains collections just like the MYSQL database contains tables. You are allowed to create multiple databases and multiple collections.

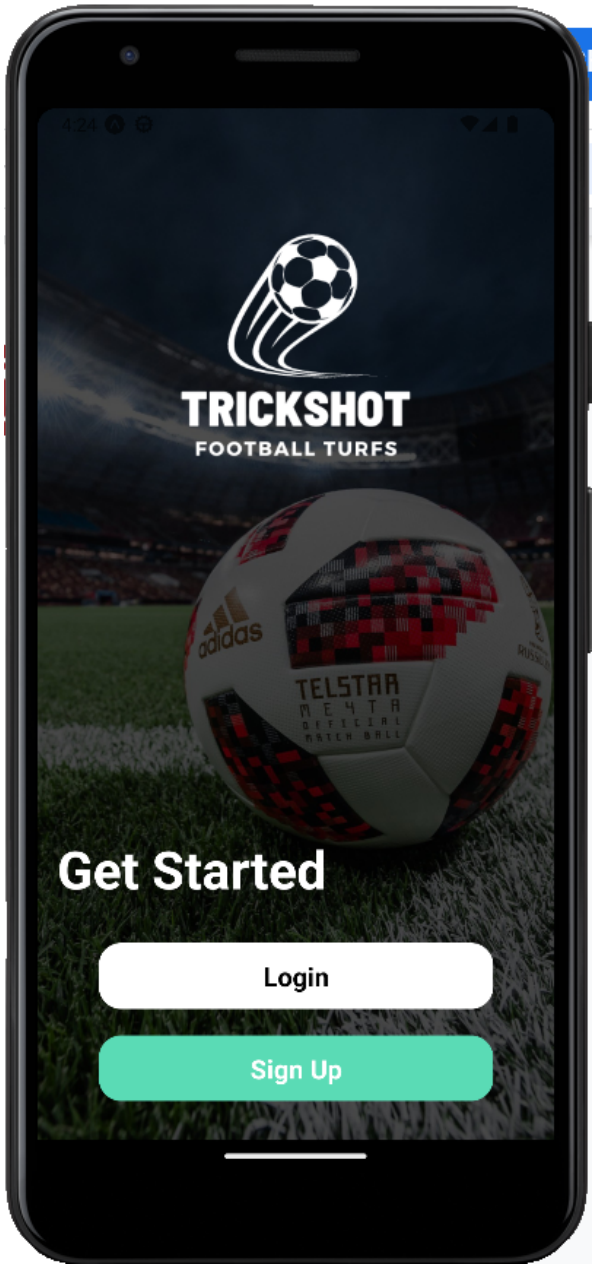
- Now inside of the collection we have documents. These documents contain the data we want to store in the MongoDB database and a single collection can contain multiple documents and you are schema-less means it is not necessary that one document is similar to another.
- The documents are created using the fields. Fields are key-value pairs in the documents, it is just like columns in the relation database. The value of the fields can be of any BSON data types like double, string, boolean, etc.
- The data stored in the MongoDB is in the format of BSON documents. Here, BSON stands for Binary representation of JSON documents. Or in other words, in the backend, the MongoDB server converts the JSON data into a binary form that is known as BSON and this BSON is stored and queried more efficiently.
- In MongoDB documents, you are allowed to store nested data. This nesting of data allows you to create complex relations between data and store them in the same document which makes the working and fetching of data extremely efficient as compared to SQL. In SQL, you need to write complex joins to get the data from table 1 and table 2. The maximum size of the BSON document is 16MB.

Project Link: https://github.com/nirmit46/TurfBookingApp

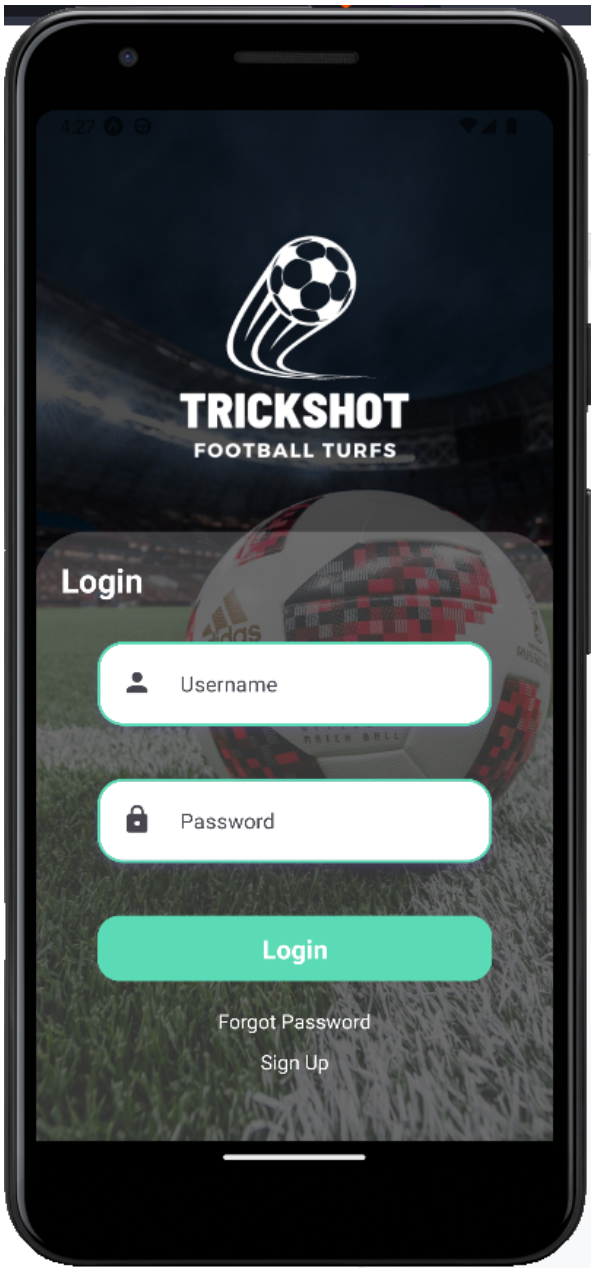
Output with Steps:

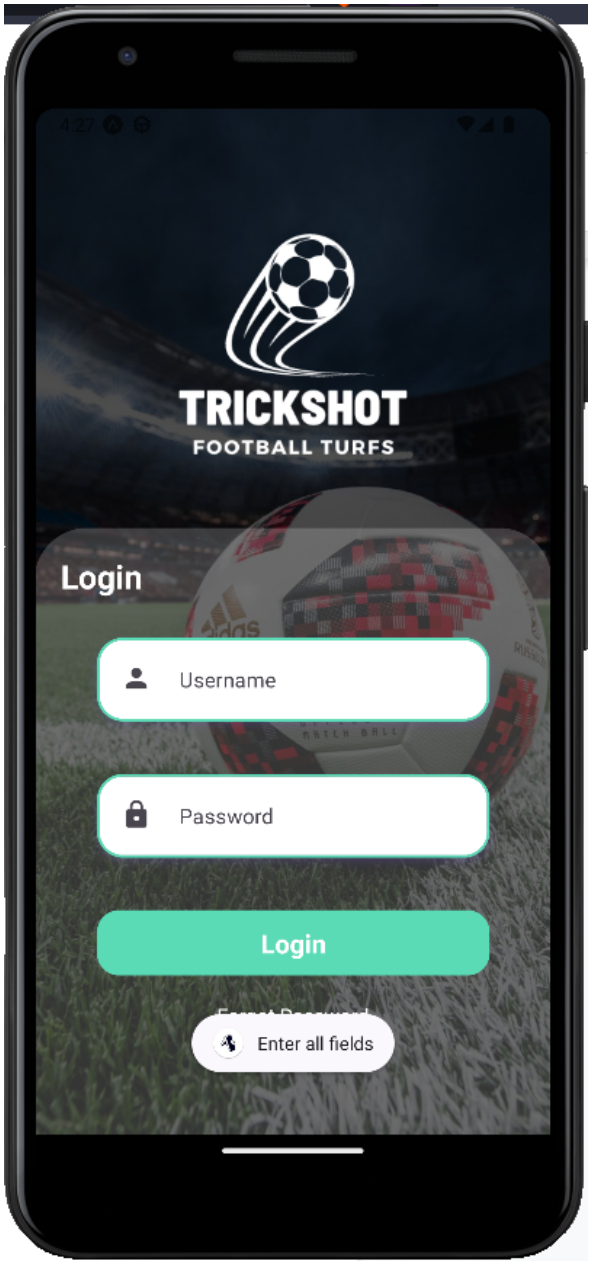
- **User App**

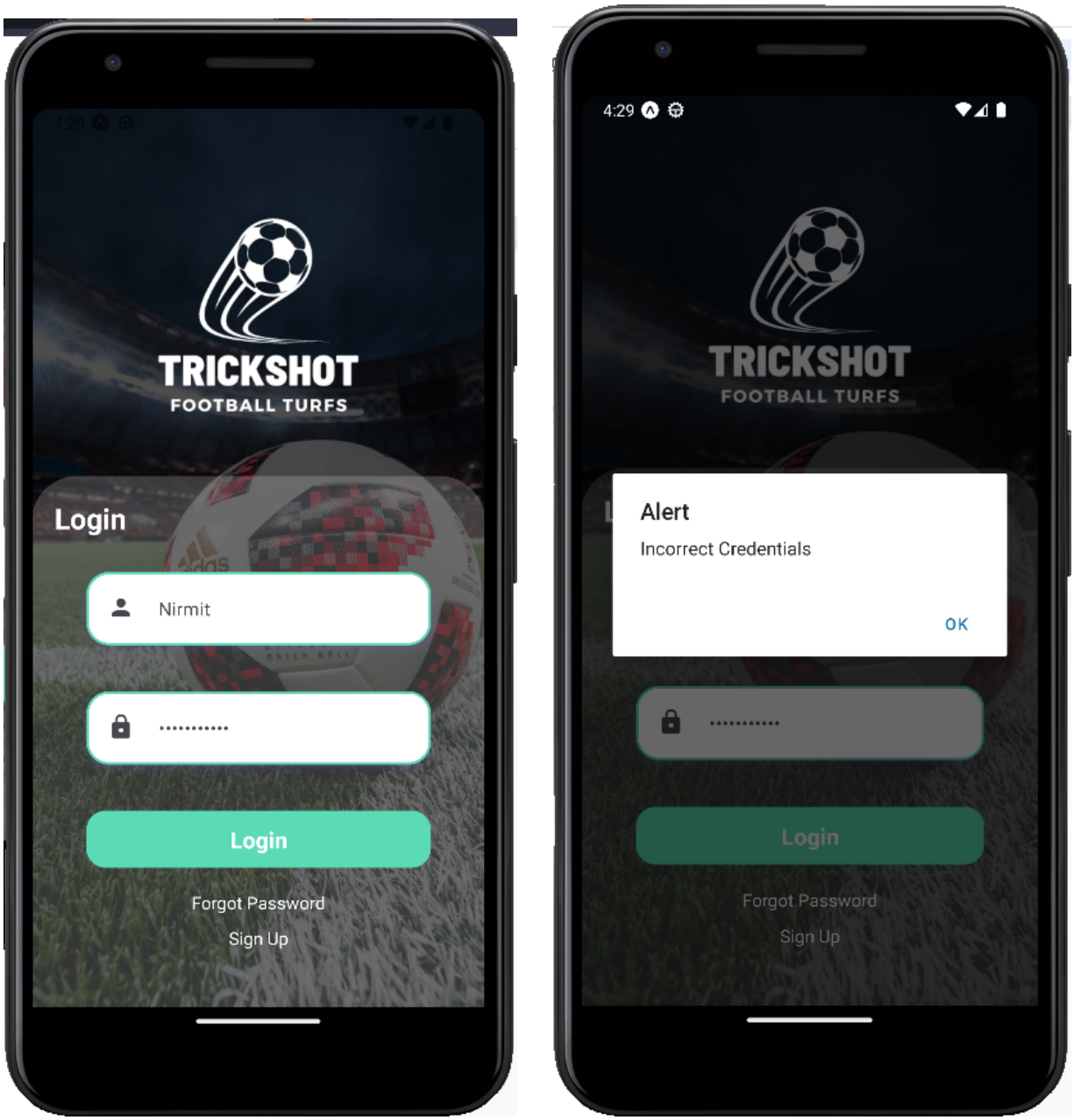
1) This is the landing page for the application. It has two options: Login and Signup. Clicking on these options will redirect you to the respective pages

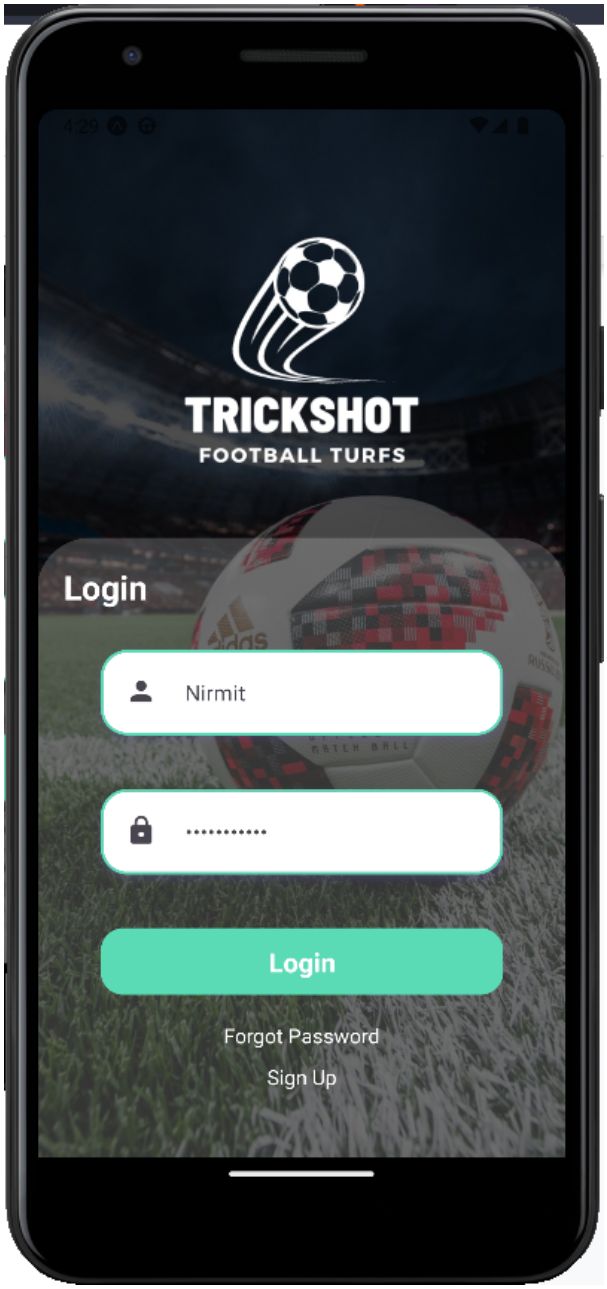


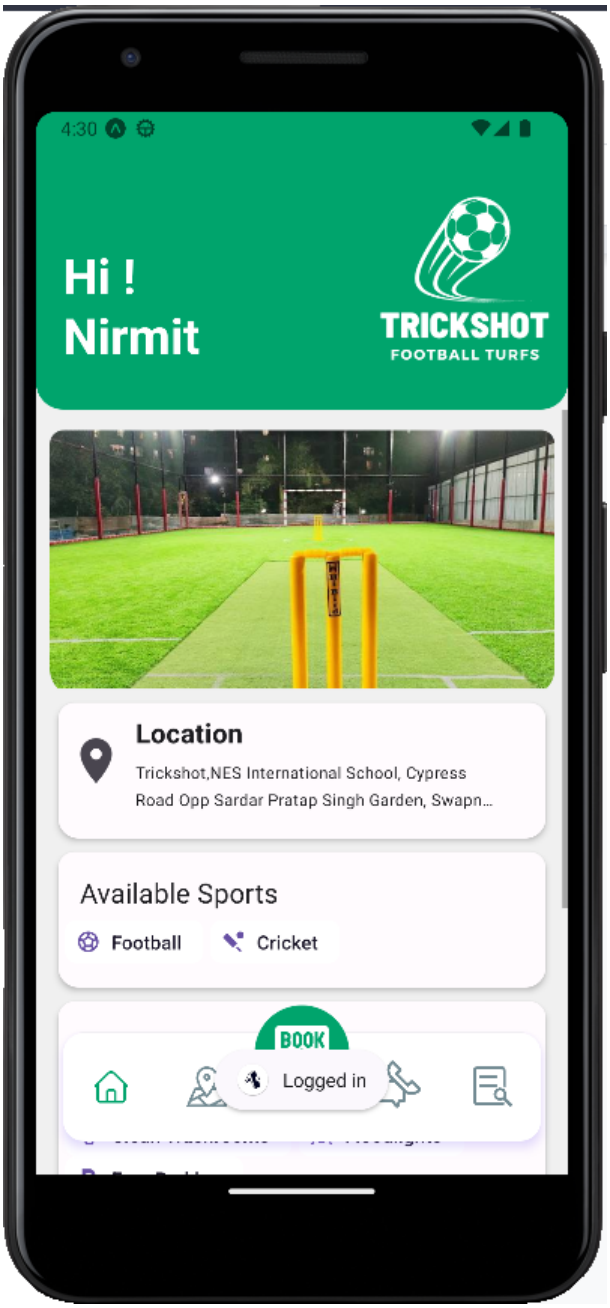
2) This is the login page. Here the user can login and go on to the homepage.
Validation used: Empty fields and incorrect username and password



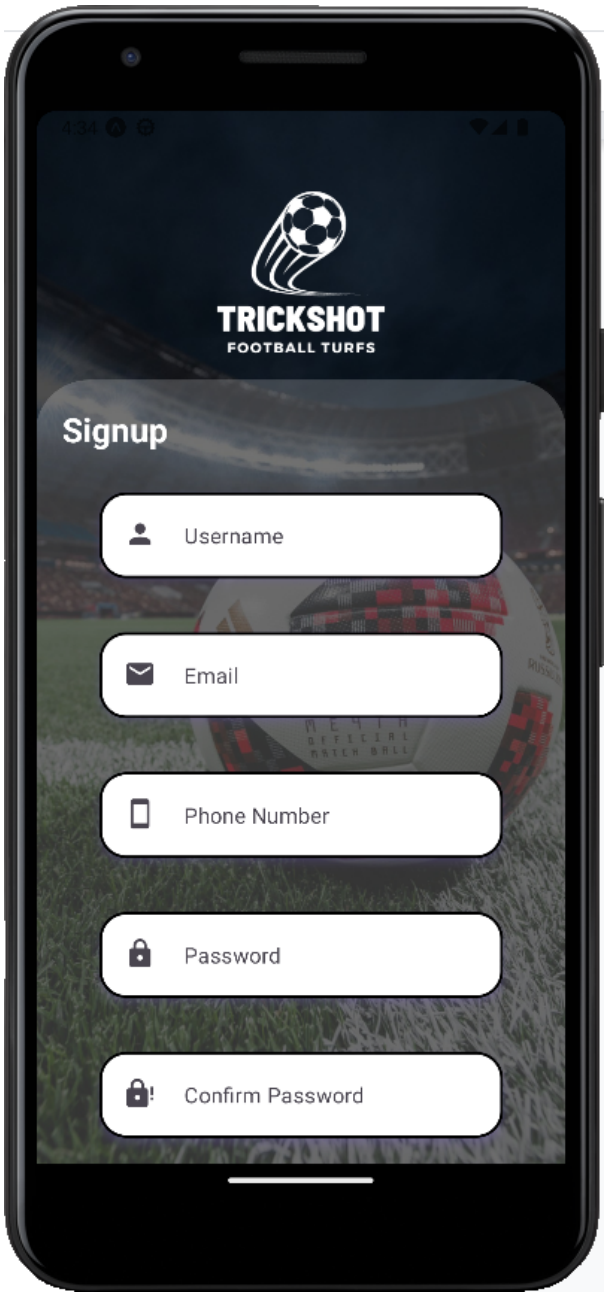


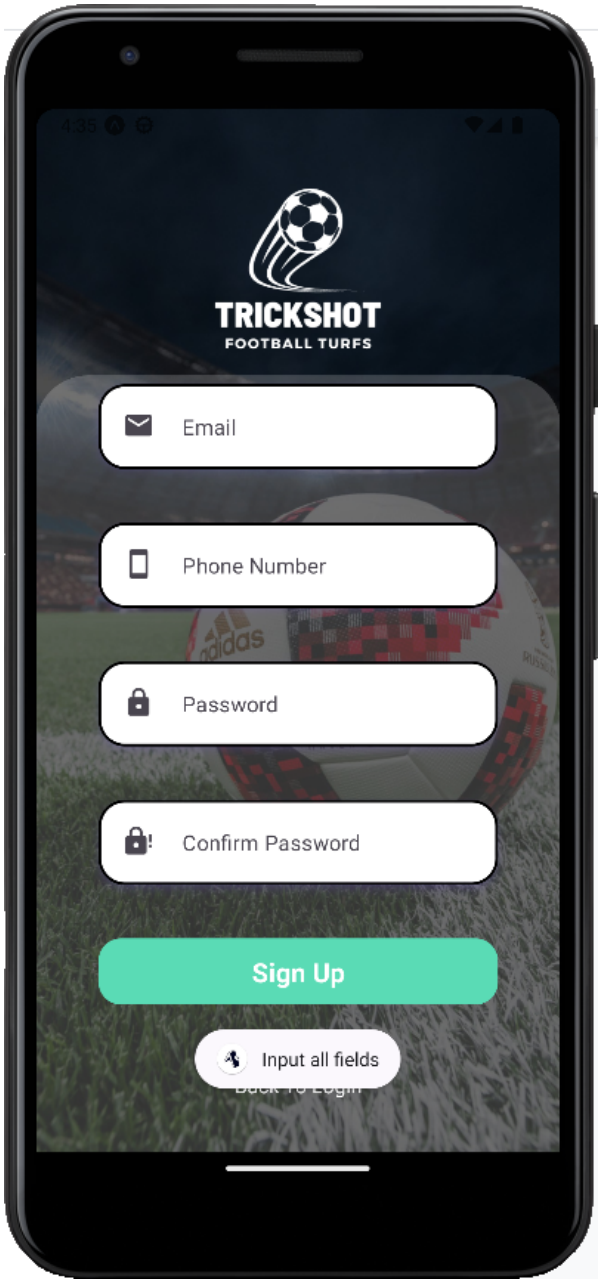

TRICKSHOT
FOOTBALL TURFS
Login
Nirmit
Login
Forgot Password
Sign Up
4:29
TRICKSHOT
FOOTBALL TURFS
Alert
Incorrect Credentials
OK
Login
Forgot Password
Sign Up

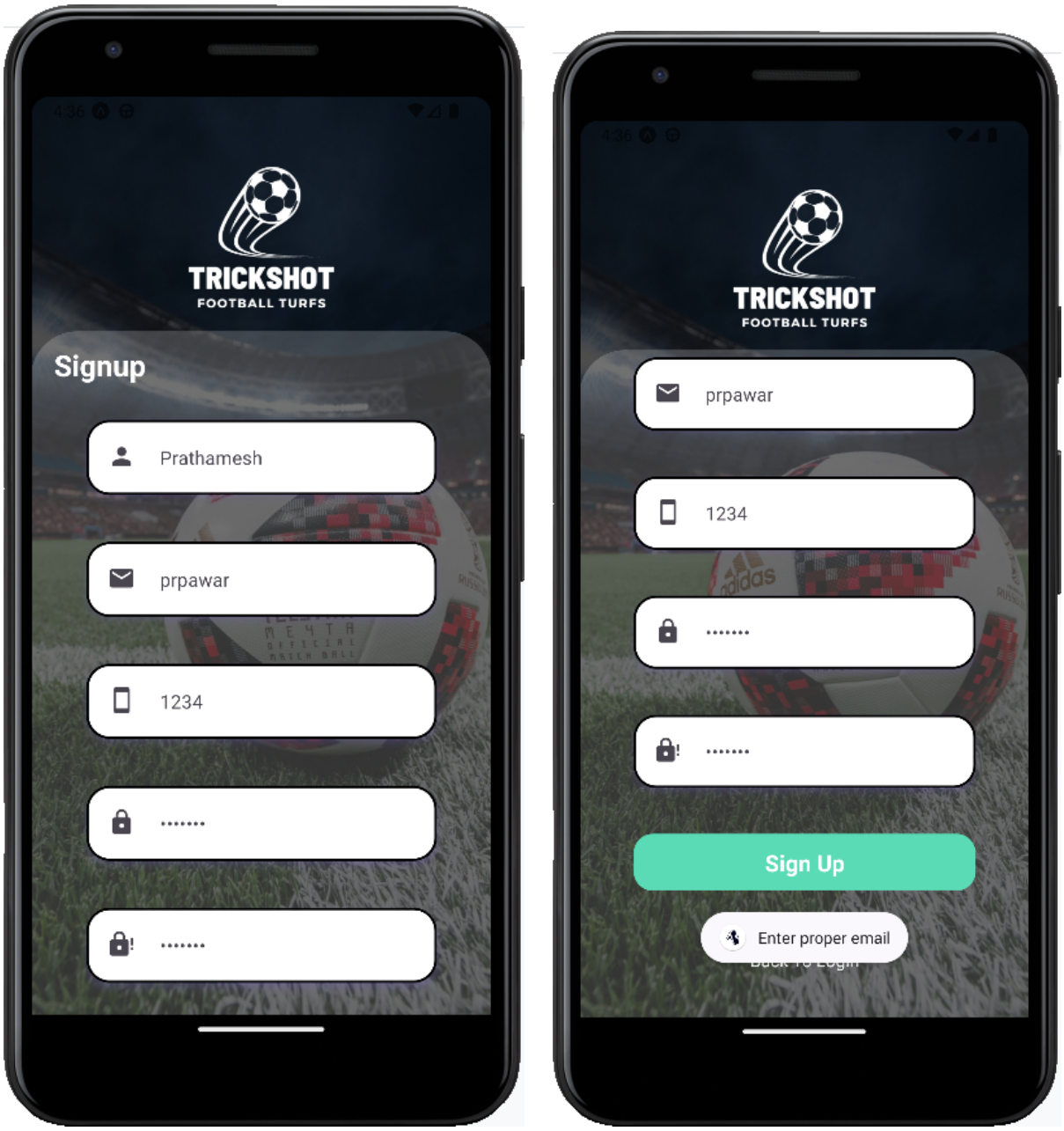

TRICKSHOT
FOOTBALL TURFS
Login
Nirmit
Login
Forgot Password
Sign Up

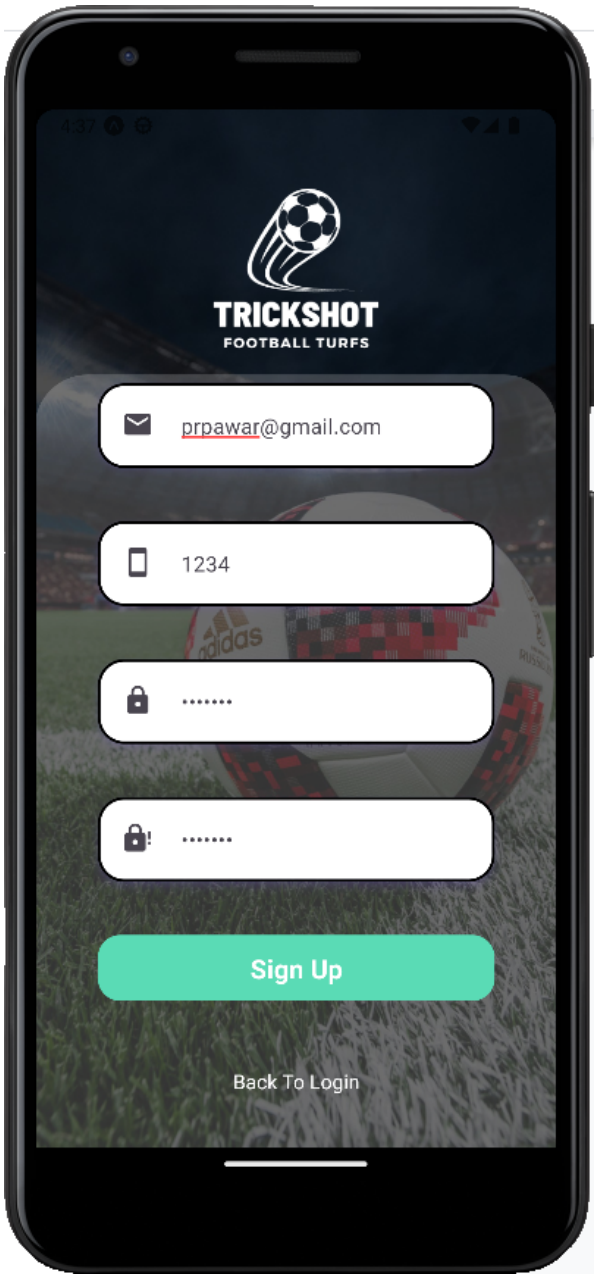

Hi !
Nirmit
TRICKSHOT
FOOTBALL TURFS
Location
Trickshot,NES International School, Cypress Road Opp Sardar Pratap Singh Garden, Swapn...
Available Sports
Football
Cricket
BOOK
Logged in

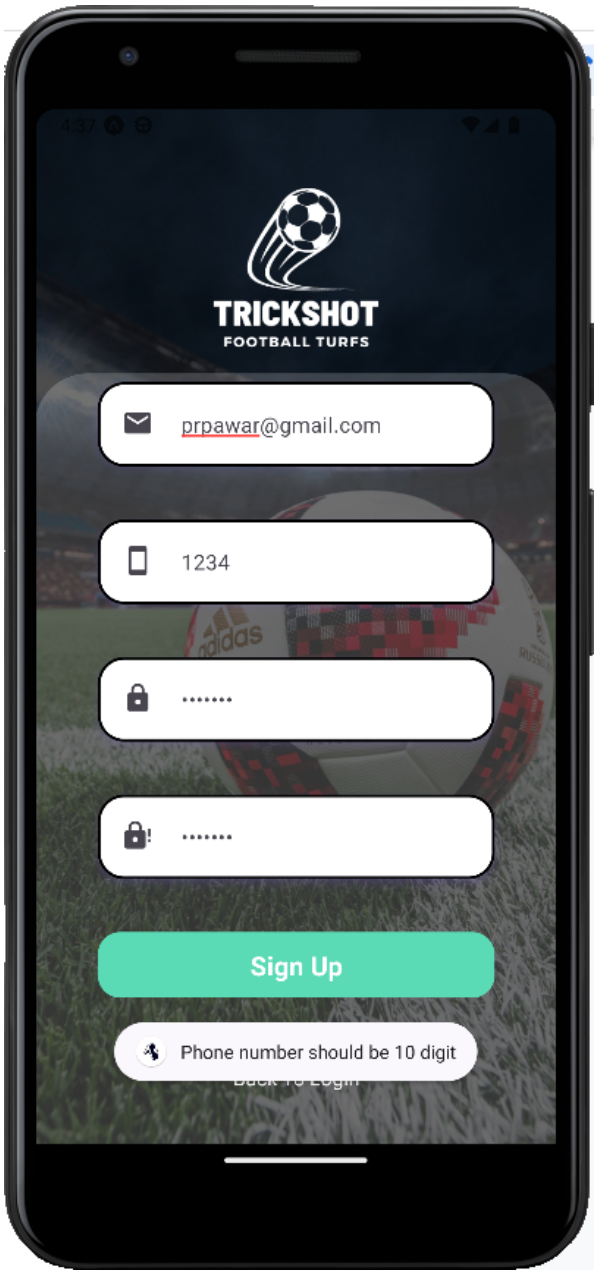
3) This is the signup page. Here user can signup for a new account in the app.
   Validations:
   a) Empty field
   b) Email
   c) Phone number
   d) Password
   e) Confirm Password

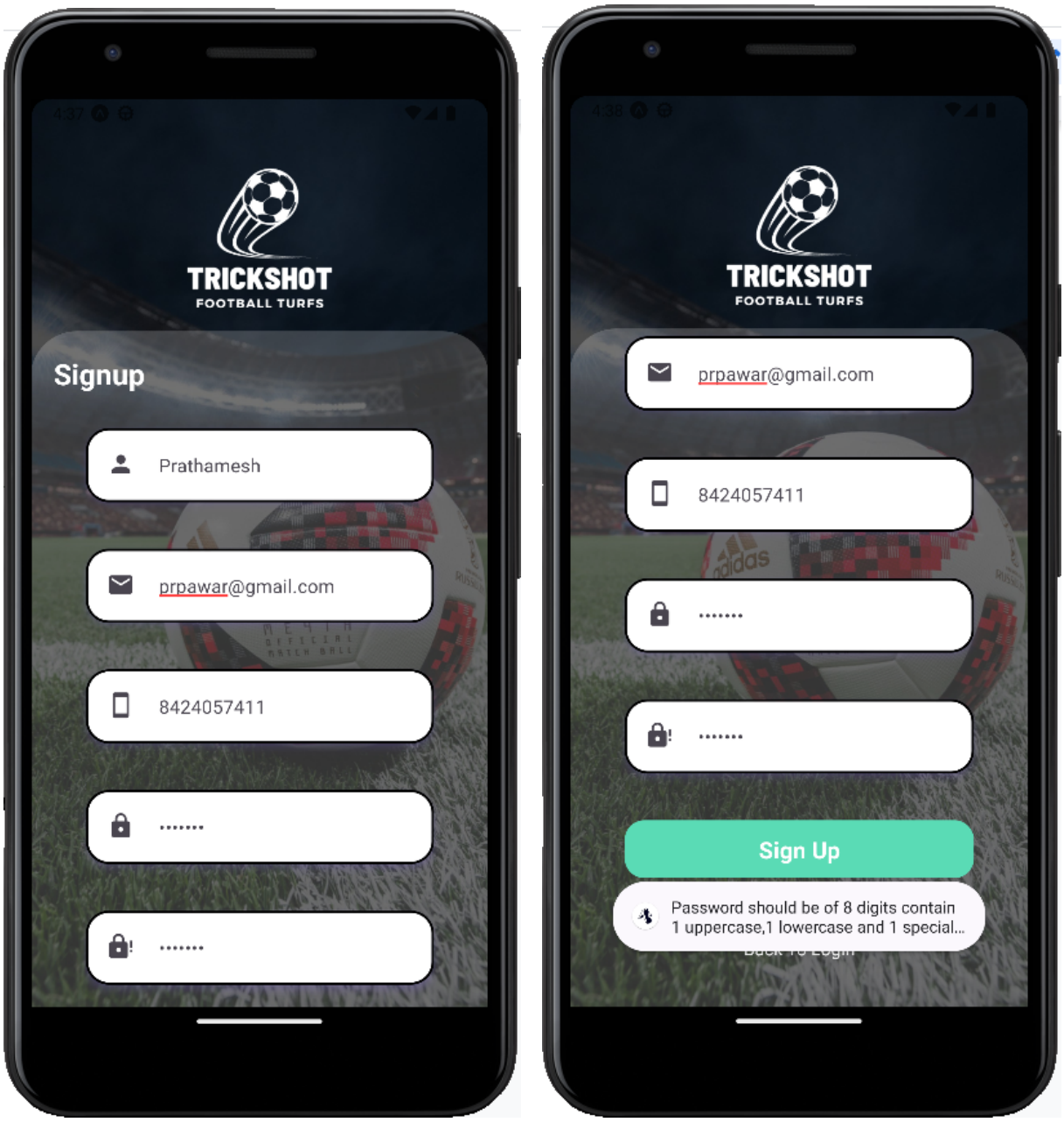


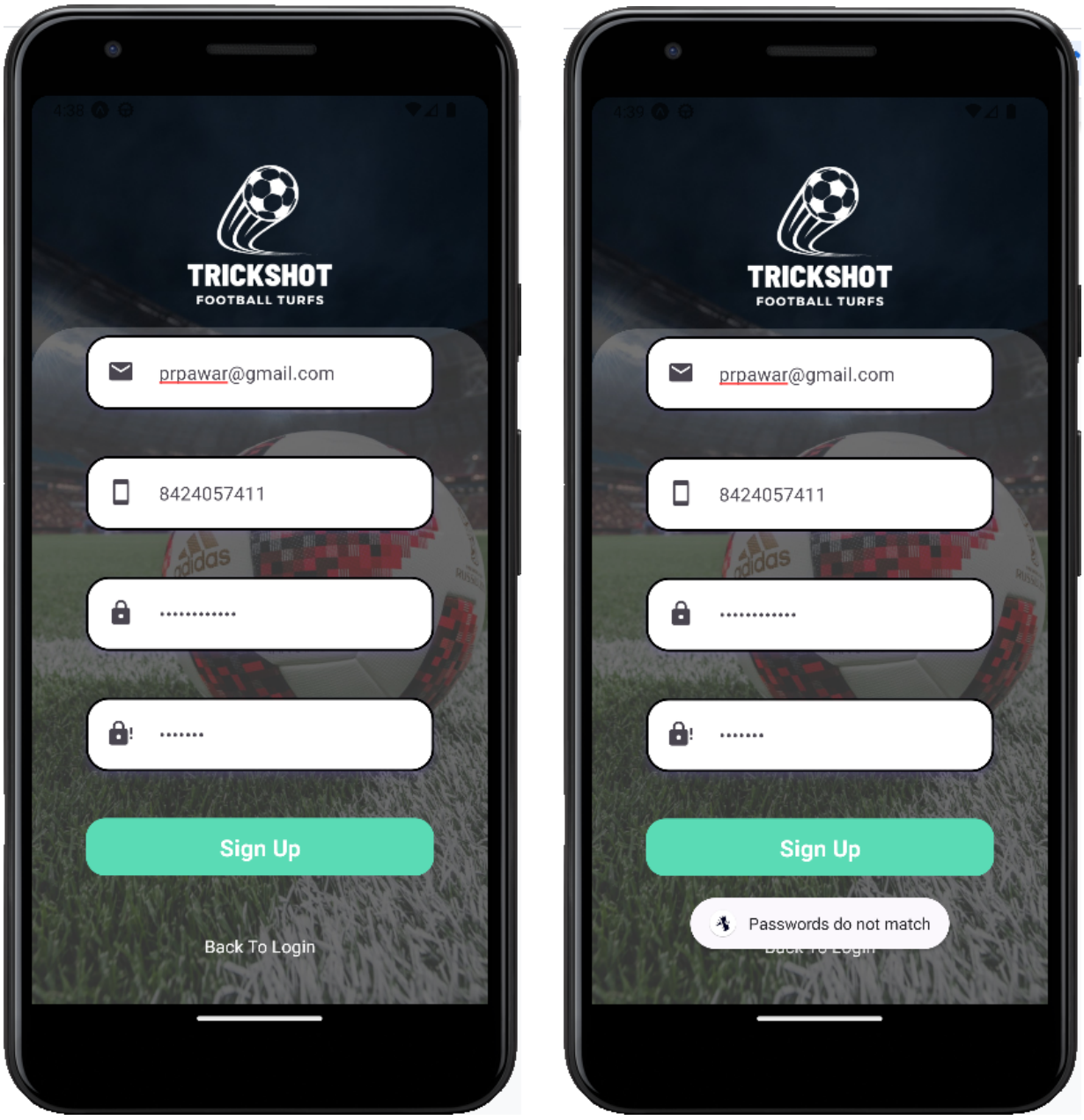



TRICKSHOT
FOOTBALL TURFS
Signup
Prathamesh
prpawar
1234
TRICKSHOT
FOOTBALL TURFS
prpawar
1234
Sign Up
Enter proper email

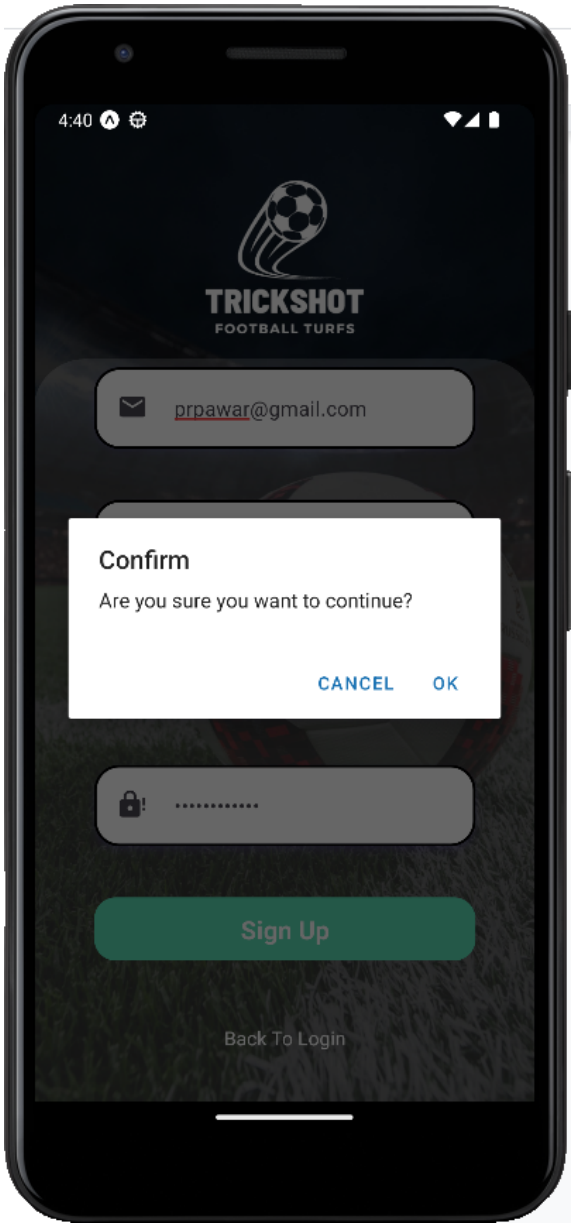

TRICKSHOT
FOOTBALL TURFS
prpawar@gmail.com
1234
Sign Up
Back To Login

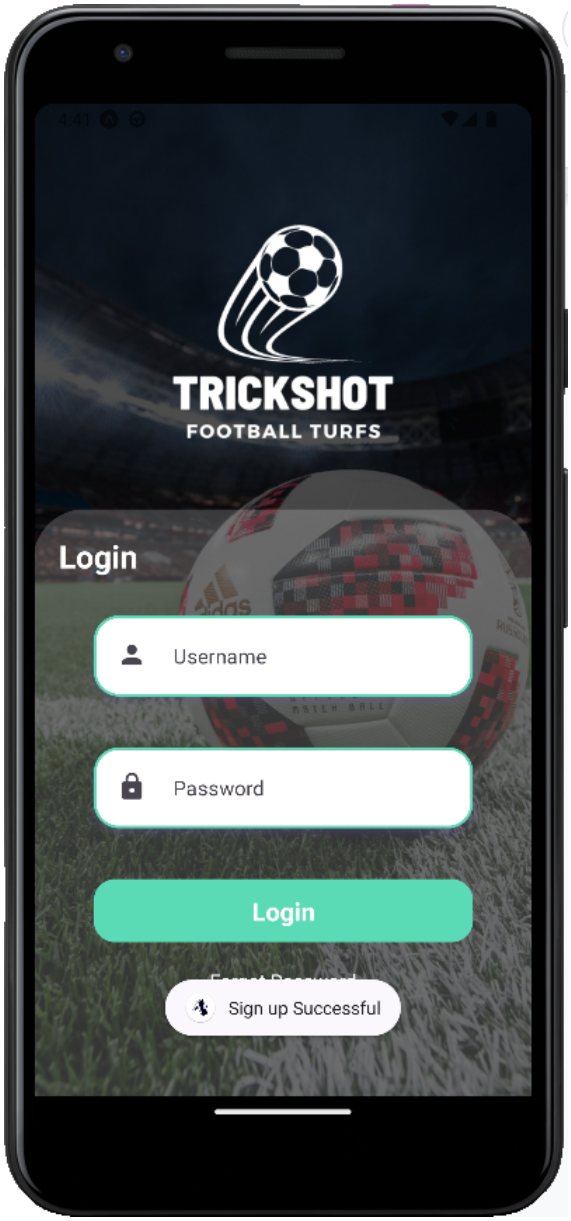

TRICKSHOT
FOOTBALL TURFS
prpawar@gmail.com
1234
Sign Up
Phone number should be 10 digit


TRICKSHOT
FOOTBALL TURFS
Signup
Prathamesh
prpawar@gmail.com
8424057411
TRICKSHOT
FOOTBALL TURFS
prpawar@gmail.com
8424057411
Sign Up
Password should be of 8 digits contain
1 uppercase,1 lowercase and 1 special...


TRICKSHOT
FOOTBALL TURFS
prpawar@gmail.com
8424057411
Sign Up
Back To Login
TRICKSHOT
FOOTBALL TURFS
prpawar@gmail.com
8424057411
Sign Up
Passwords do not match

4) After entering all the information properly the user is signed up.

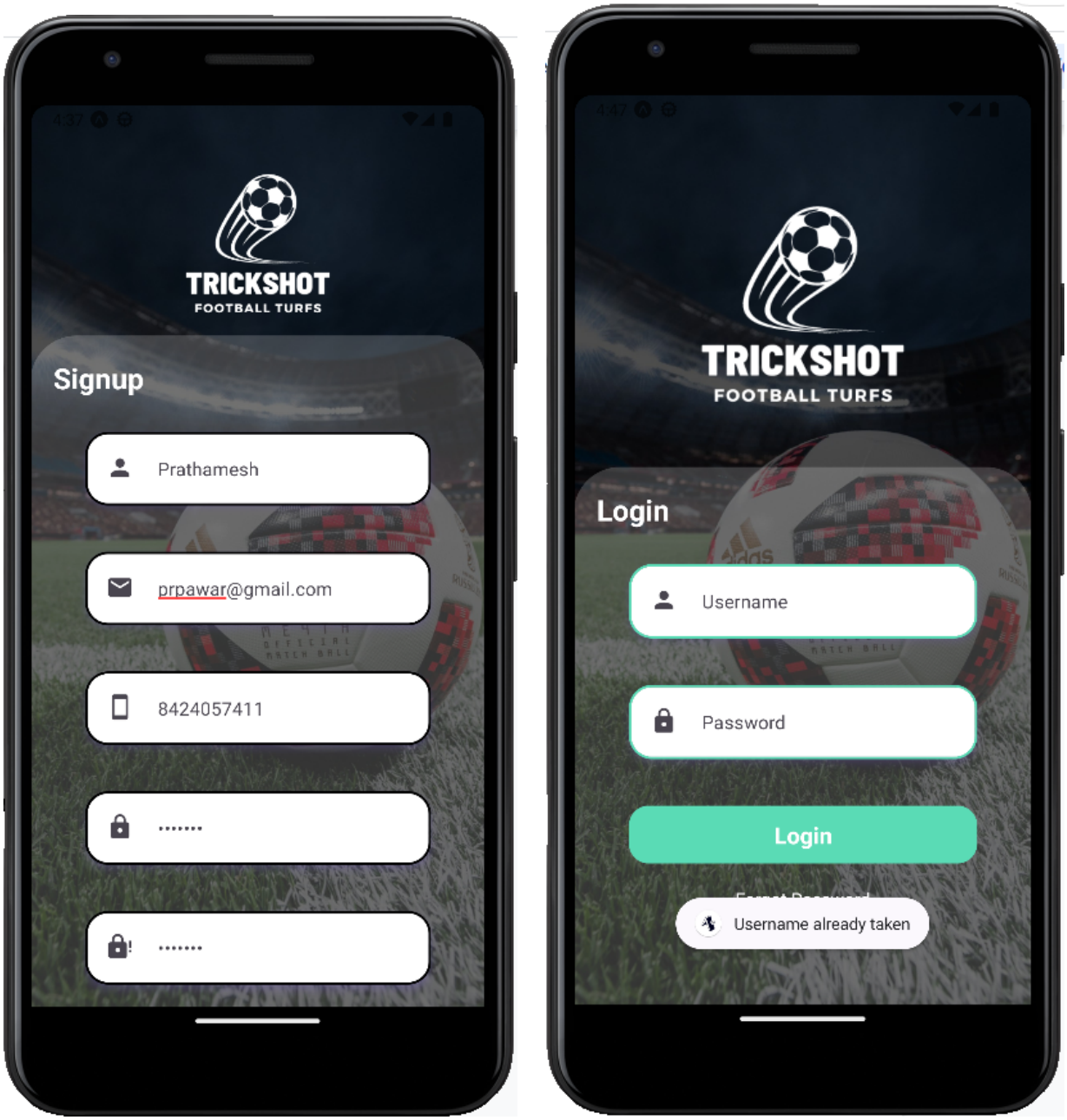




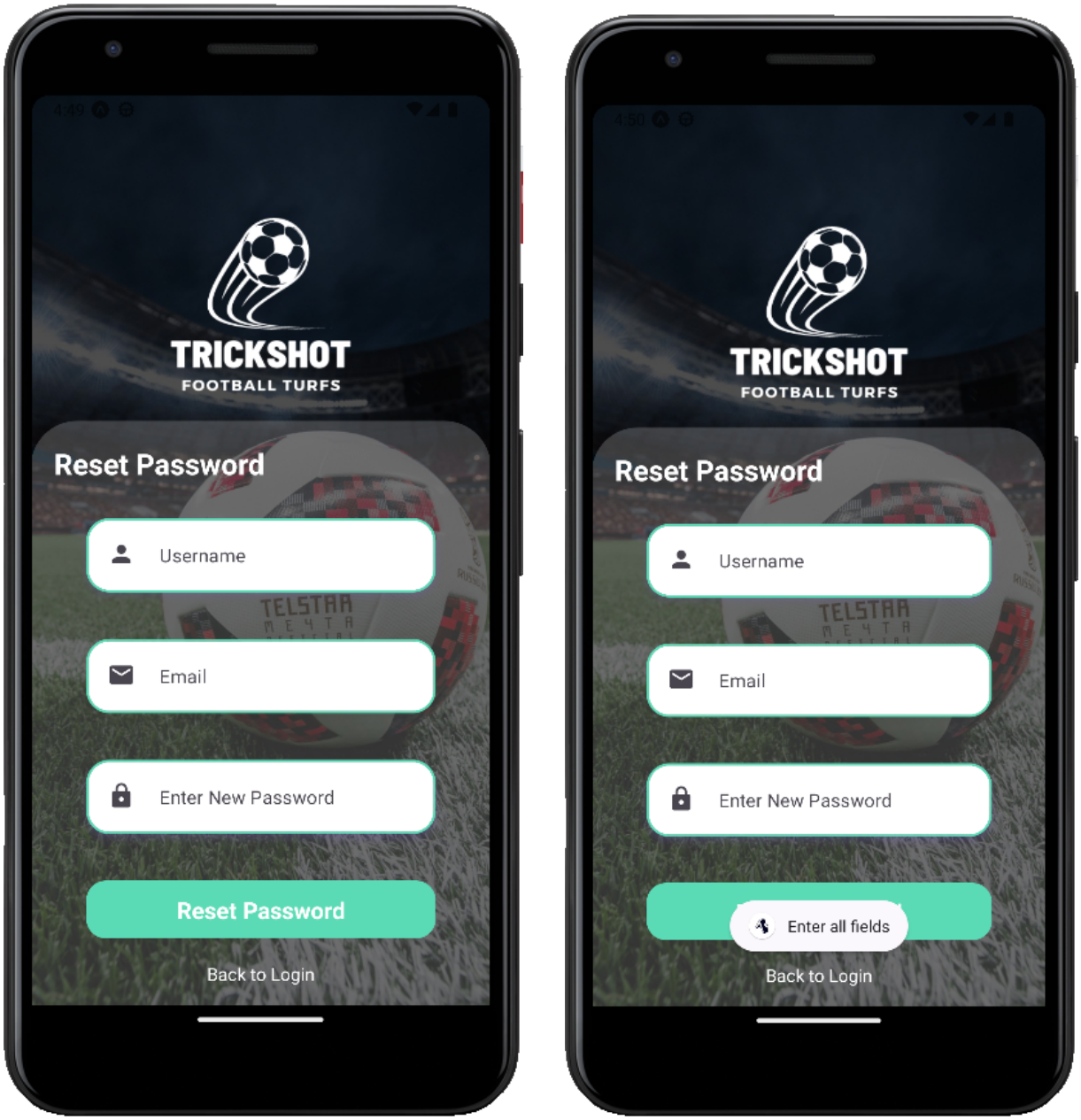
In MongoDb we can see the user is added

```
_id: "Prathamesh"
email: "prpawar@gmail.com"
phoneno: "8424057411"
password: "Prpawar@1234"
_class: "com.example.booking.model.UserModel"
```

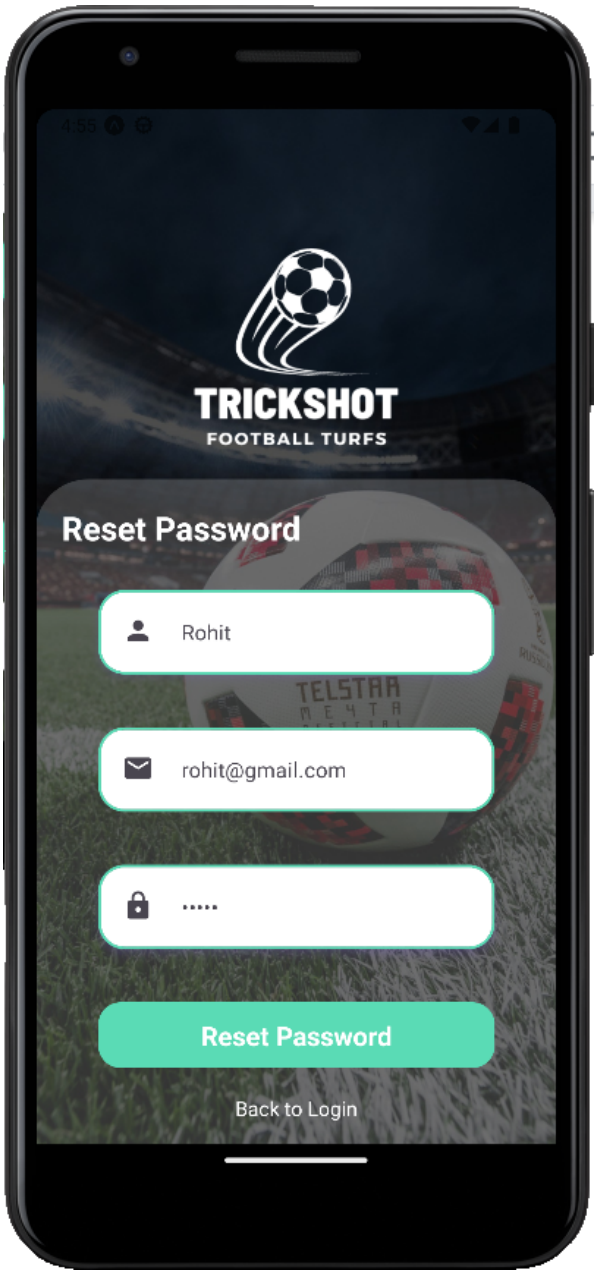
5) If a user tries to signup again with the same username he will get an message that user exists

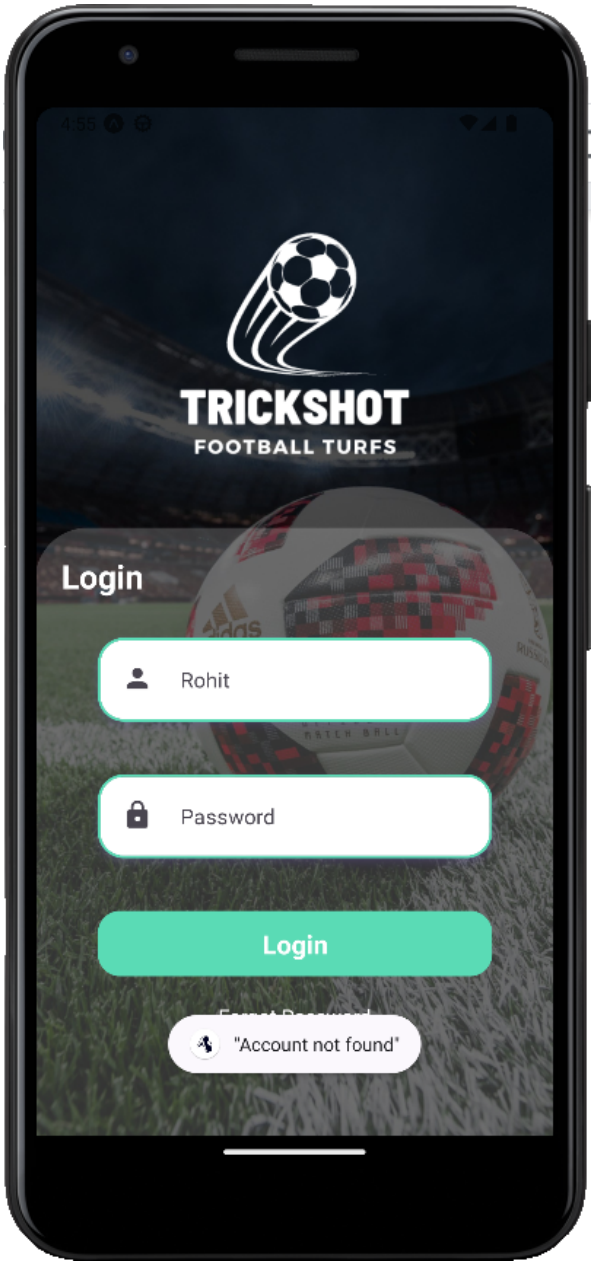


6) This is the reset password page. If a user forgets his/her password then he/she can reset it using this screen. Validations Used:
    a) Empty field
    b) If username and email doesn't match then the password will not update
    c) If account doesnt exist in db it will show account not found




TRICKSHOT
FOOTBALL TURFS
Reset Password
Prathamesh
prathamesh@gmail.com
Reset Password
Back to Login
TRICKSHOT
FOOTBALL TURFS
Login
Username
Password
Login
'Email does not match with the username'


TRICKSHOT
FOOTBALL TURFS
Reset Password
Rohit
rohit@gmail.com
.....
Reset Password
Back to Login


TRICKSHOT
FOOTBALL TURFS
Login
Rohit
Password
Login
"Account not found"

7) If we enter all the info properly we can reset the password

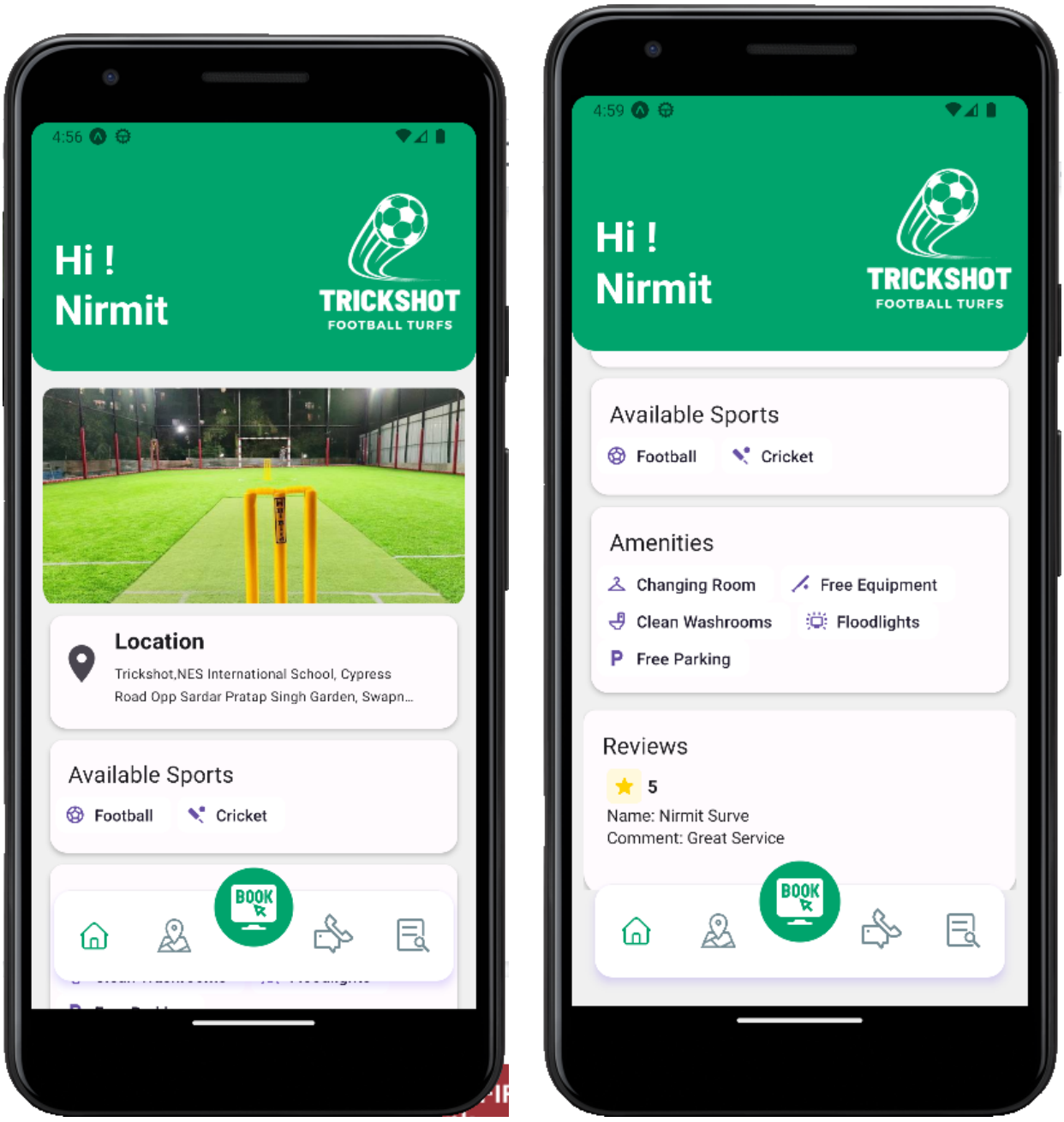
Here you can see the password is changed


```
_id: "Prathamesh"
email: "prpawar@gmail.com"
phoneno: "8424057411"
password: "Prpawar@1234"
_class: "com.example.booking.model.UserModel"
```

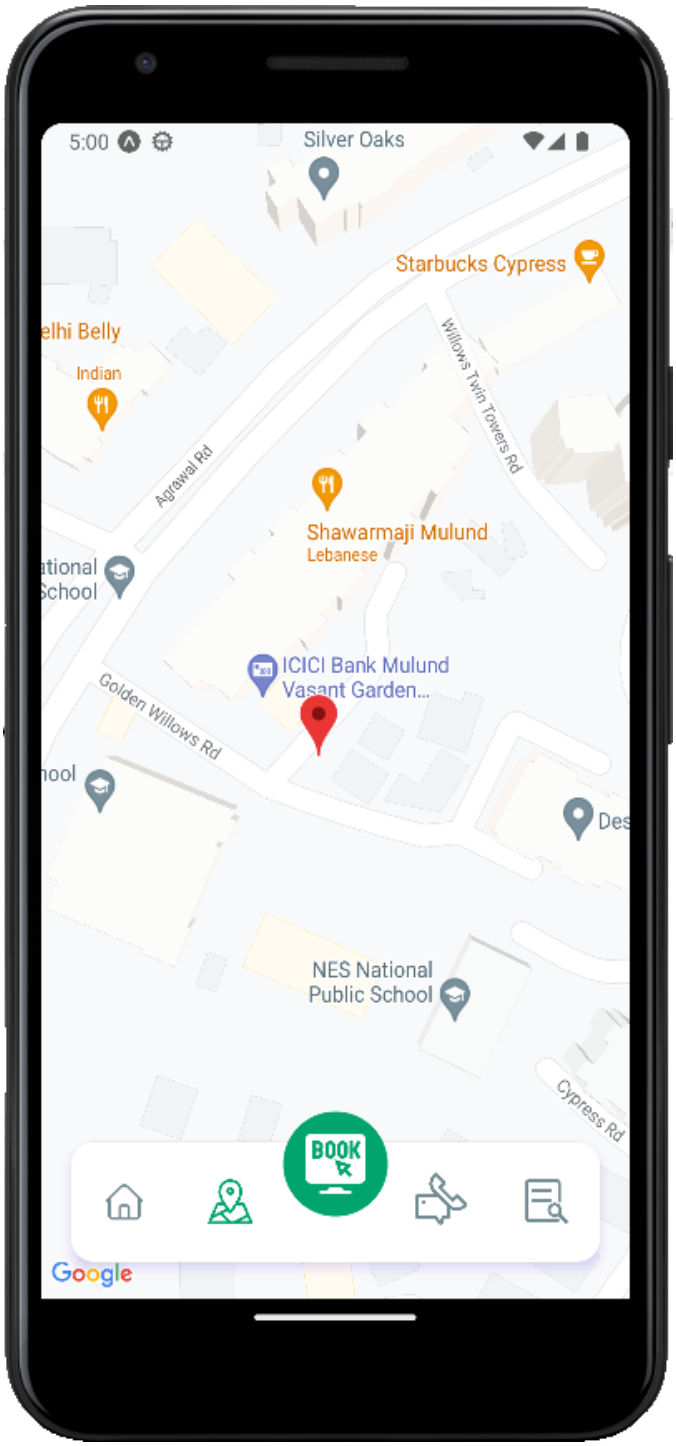


```
_id: "Prathamesh"
email: "prpawar@gmail.com"
phoneno: "8424057411"
password: "Prathamesh@1234"
_class: "com.example.booking.model.UserModel"
```

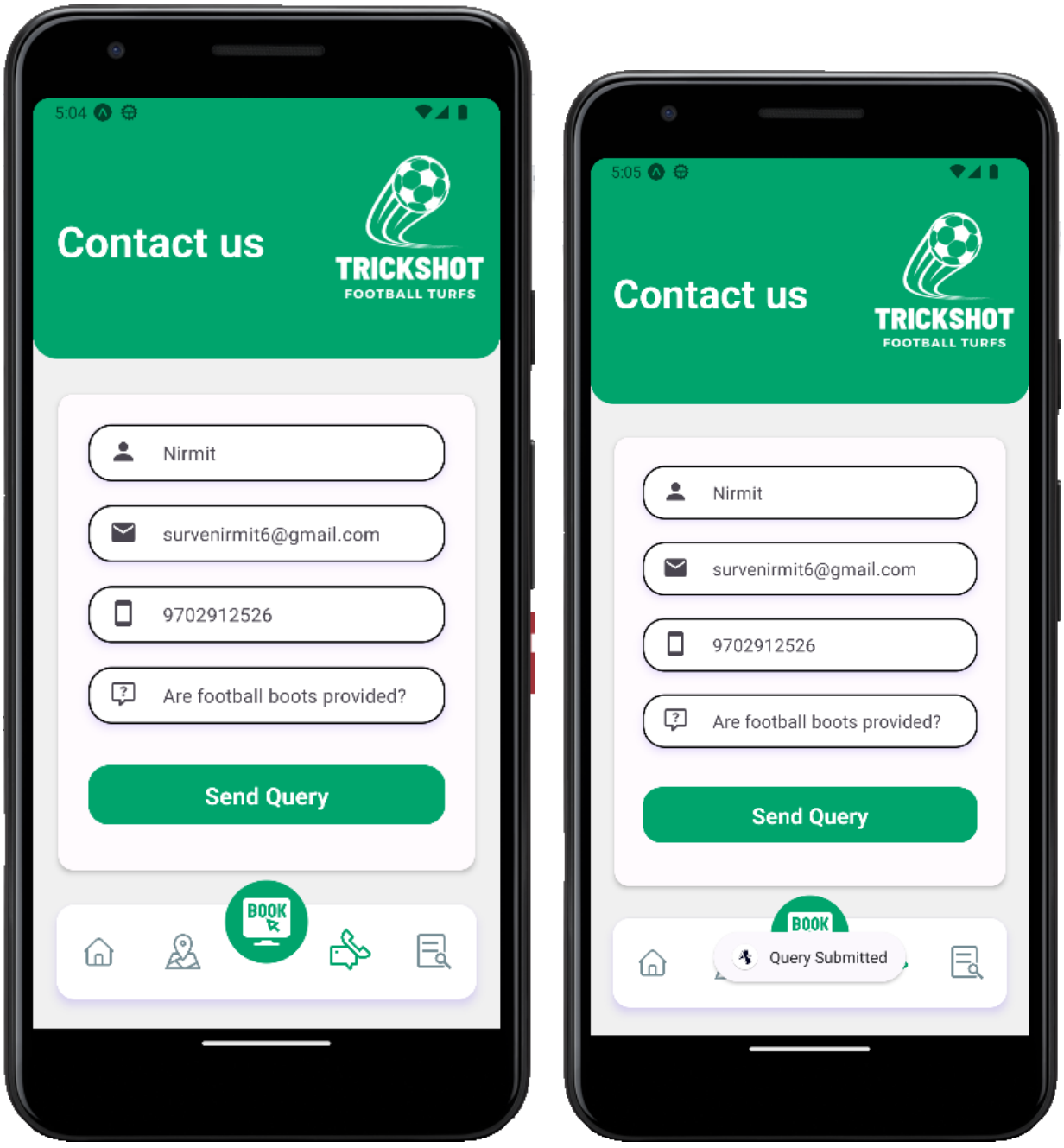
8) This is the Homepage. It contains all the information about the turf and also shows the user reviews . Here I have logged in by Nirmit username



9) This is the location tab. It consists of a Google Maps location of the turf for the user to easily navigate to it



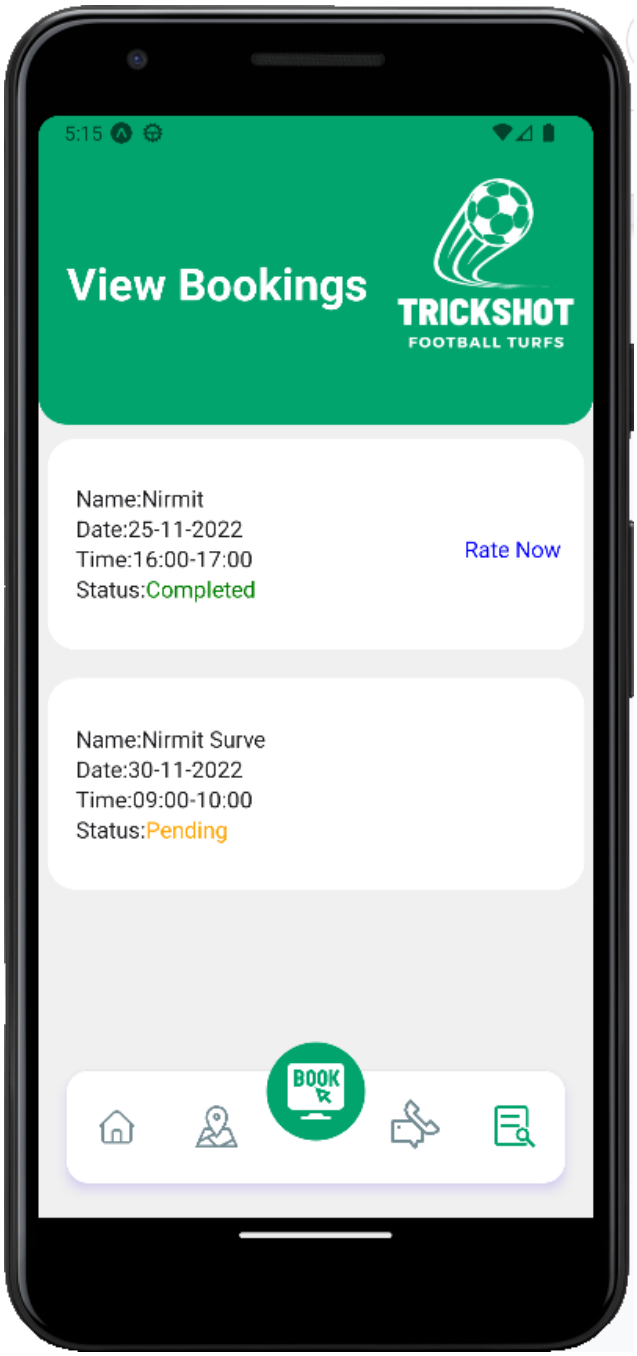
10) This is the contact form. Here the user can submit his/her queries to the turf owner and the owner can see them in the admin app



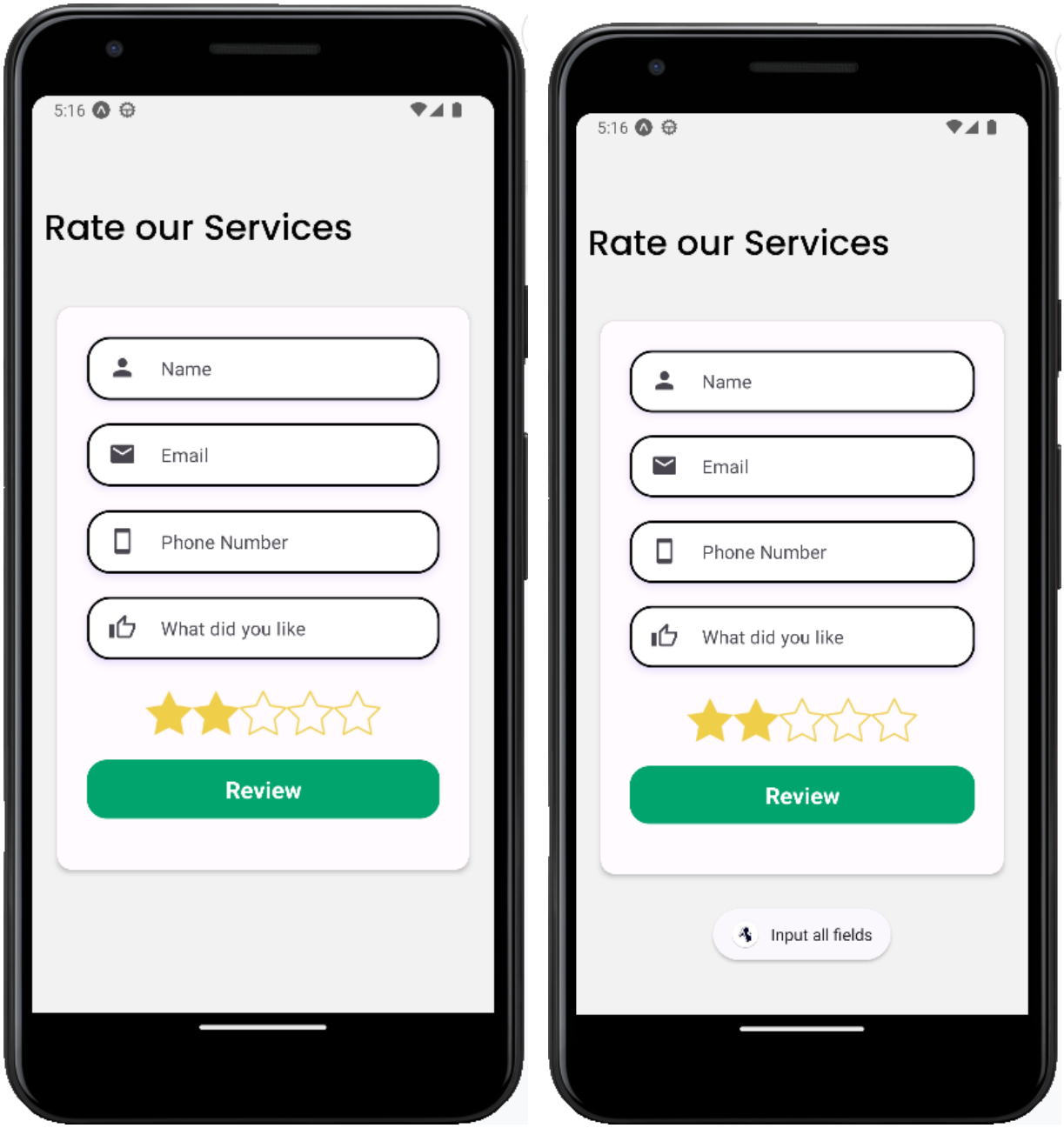
This queries are updated in mongodb as follows:

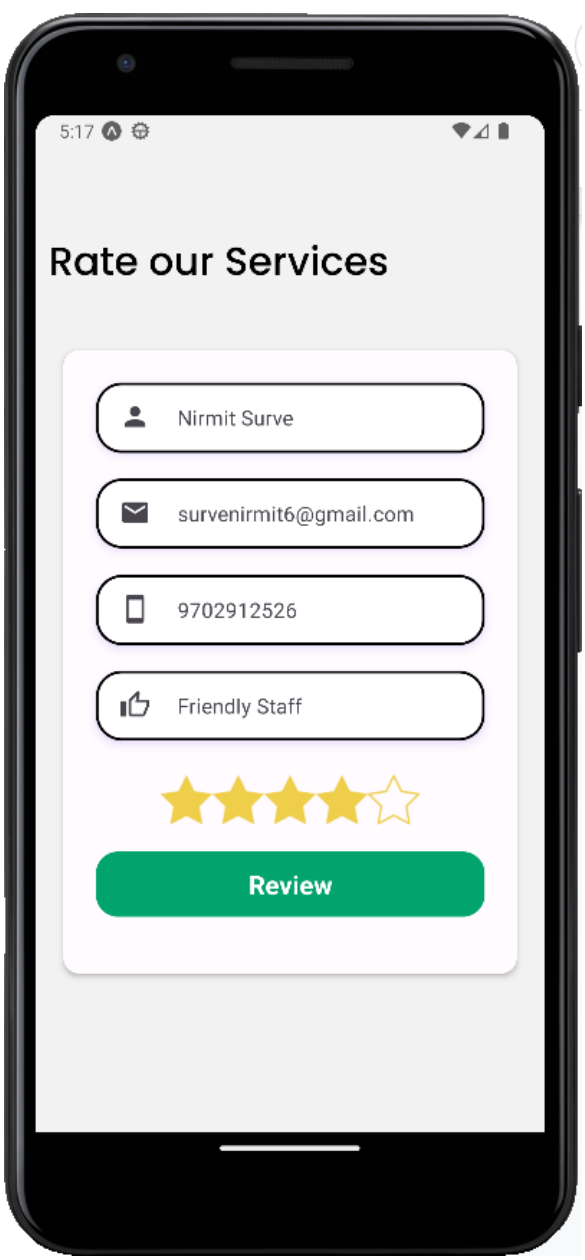
```
_id: ObjectId('63849ce91751611cf34a6d47')
name: "Nirmit"
email: "survenirmit6@gmail.com"
phoneno: "9702912526"
query: "Are football boots provided?"
_class: "com.example.booking.model.ContactModel"
```

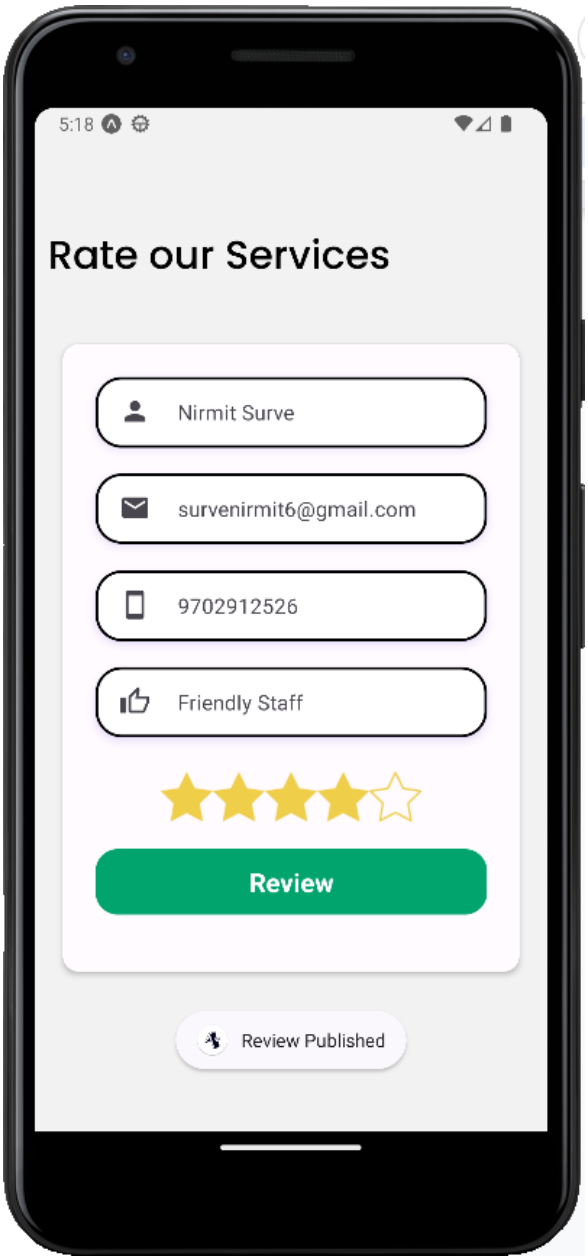
11) This is the view bookings tab. Here the user can see the bookings. If the booking date is less than todays date it shows the status as completed and if it is greater than todays date it shows the status as pending. Also for the completed bookings we have an option of reviewing the service



12) If we click the rate now button we can redirect to the review page

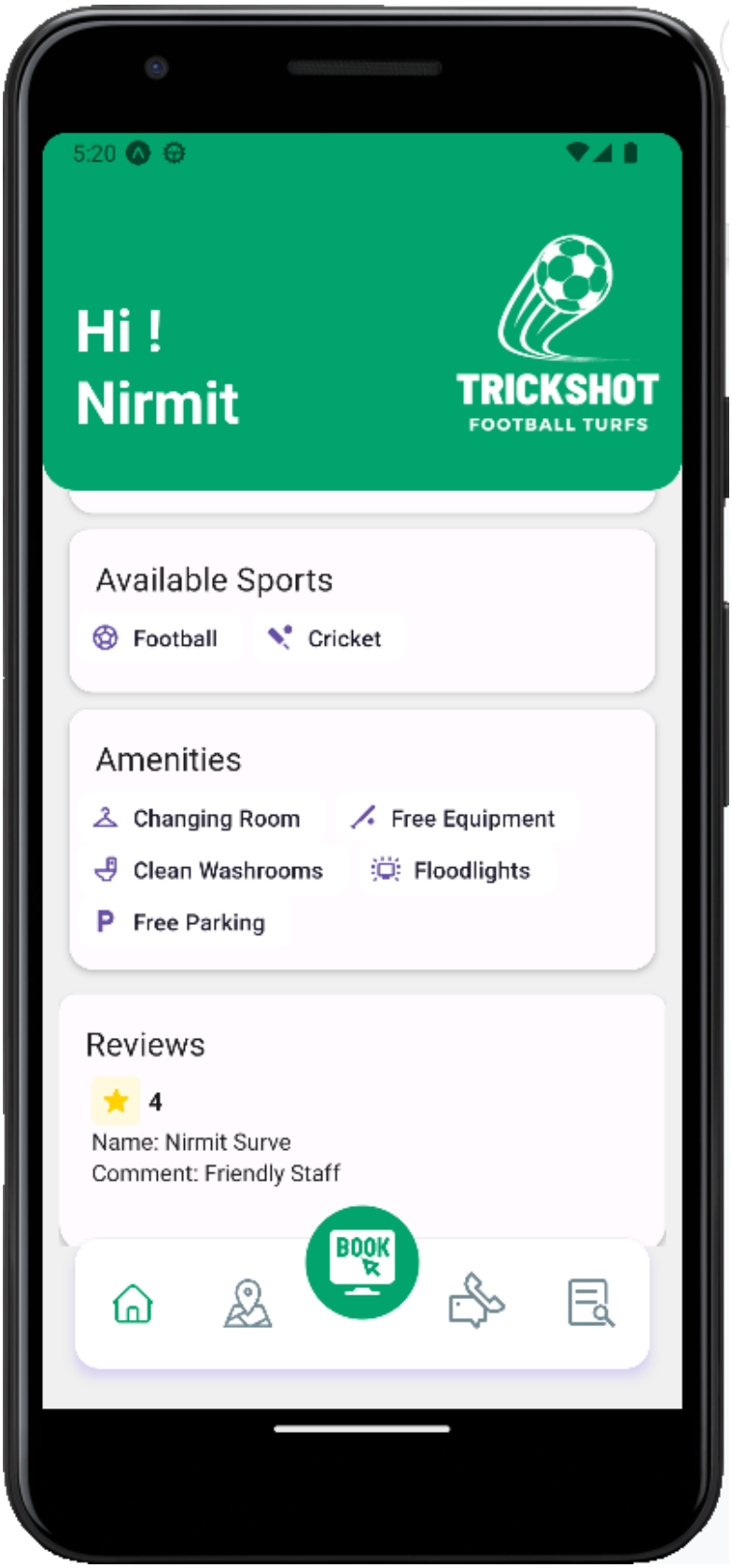




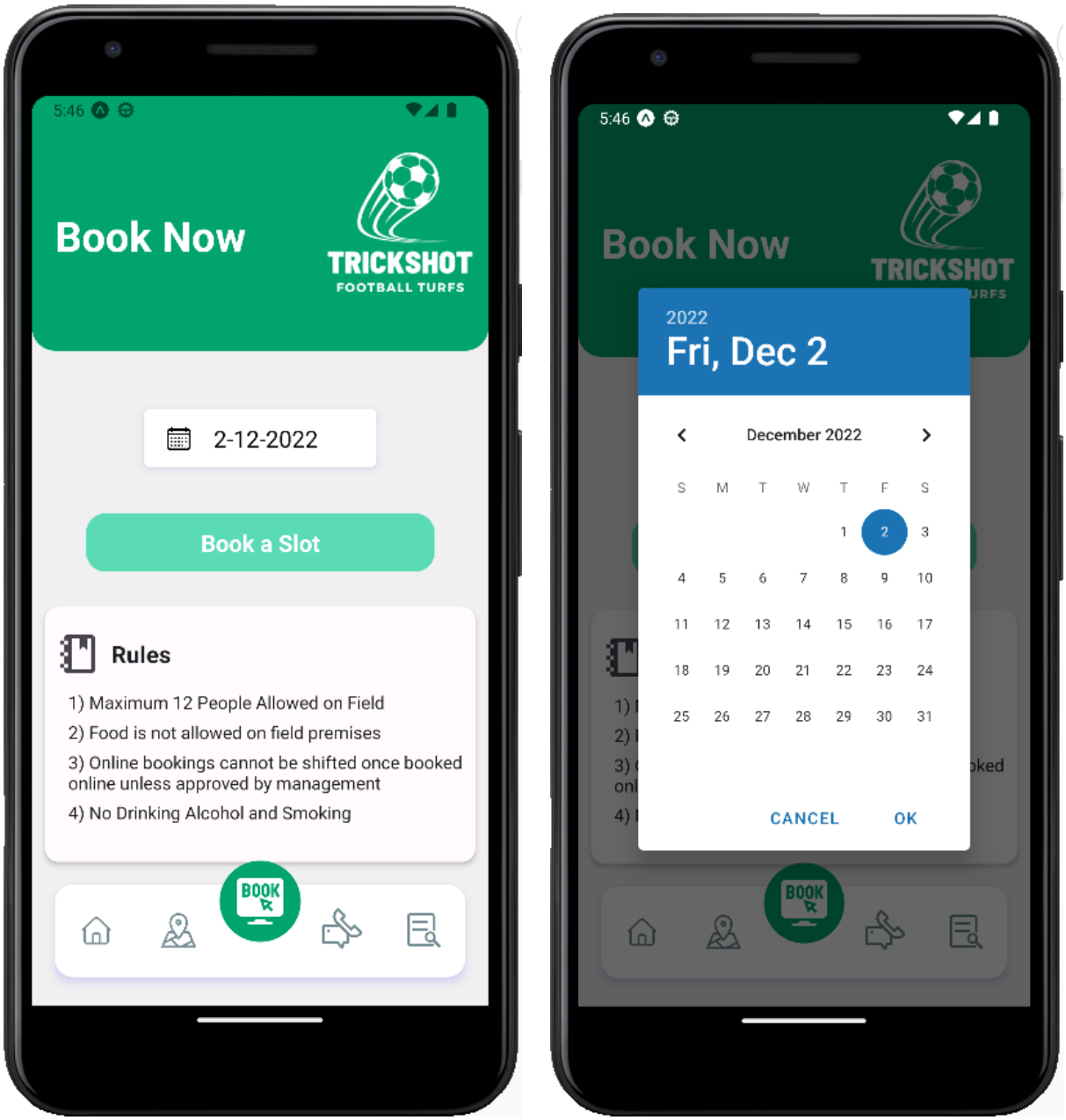


This review is saved in the database and is shown on the homepage

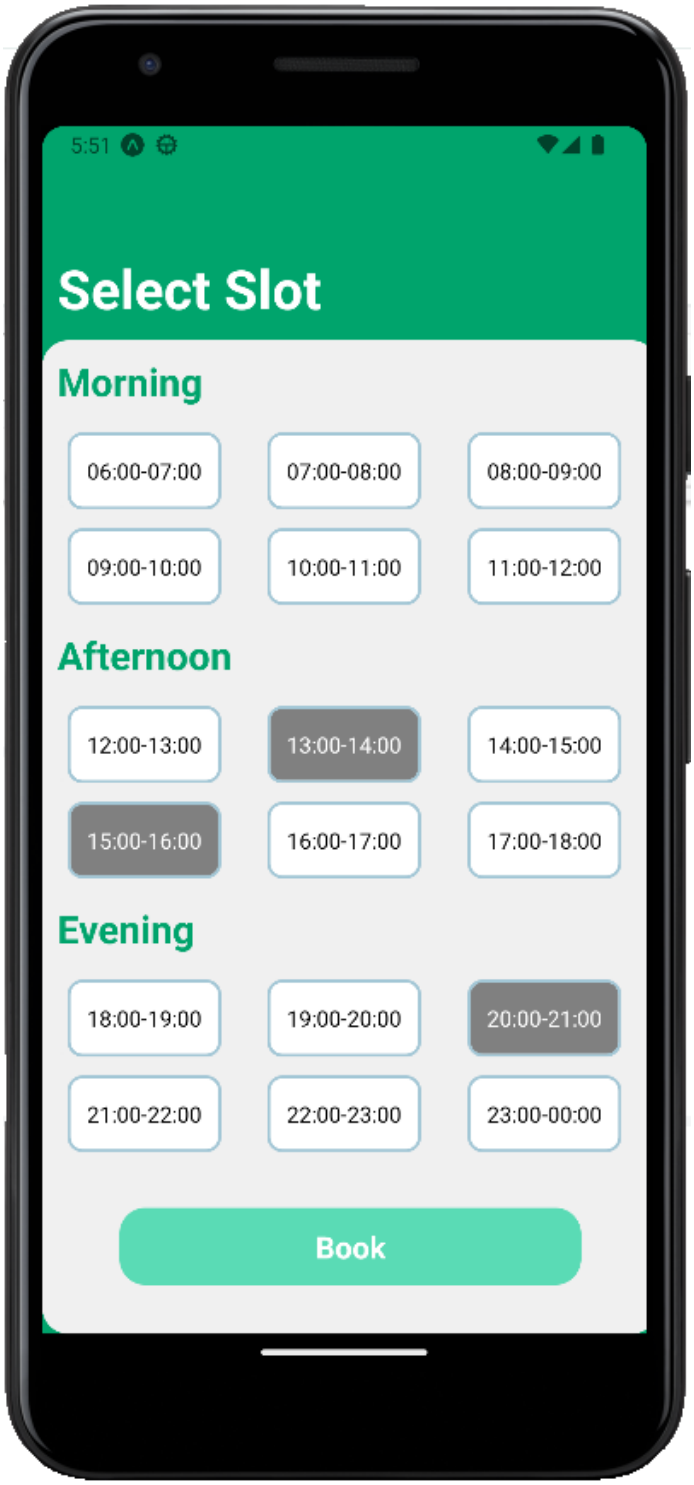
```
_id: "Nirmit Surve"
email: "survenirmit6@gmail.com"
phoneno: "9702912526"
like: "Friendly Staff"
star: "4"
_class: "com.example.booking.model.ReviewModel"
```


5:20
Hi !
Nirmit
TRICKSHOT
FOOTBALL TURFS
Available Sports
Football
Cricket
Amenities
Changing Room
Free Equipment
Clean Washrooms
Floodlights
Free Parking
Reviews
4
Name: Nirmit Surve
Comment: Friendly Staff
BOOK

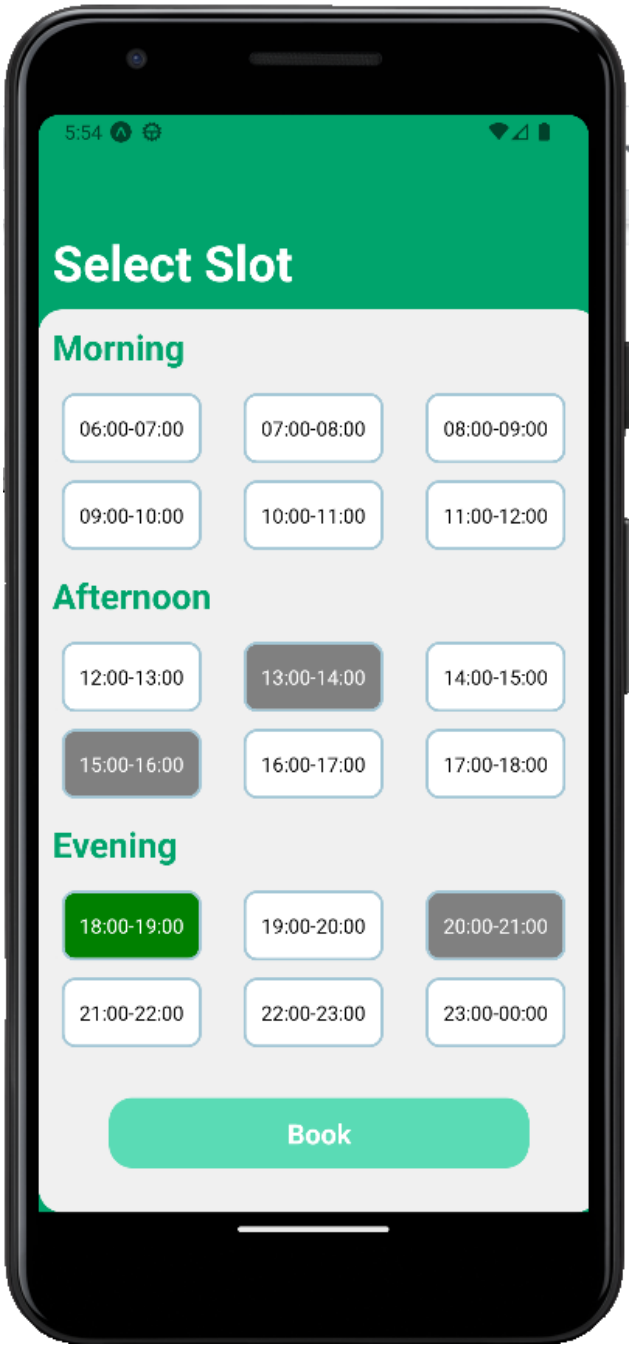
13) This is the Date selection page. Here we select our date and move on to the slot booking page



14) This is the slot booking page. Here we can see only the slots which are empty, are selectable and those who are booked are marked in grey

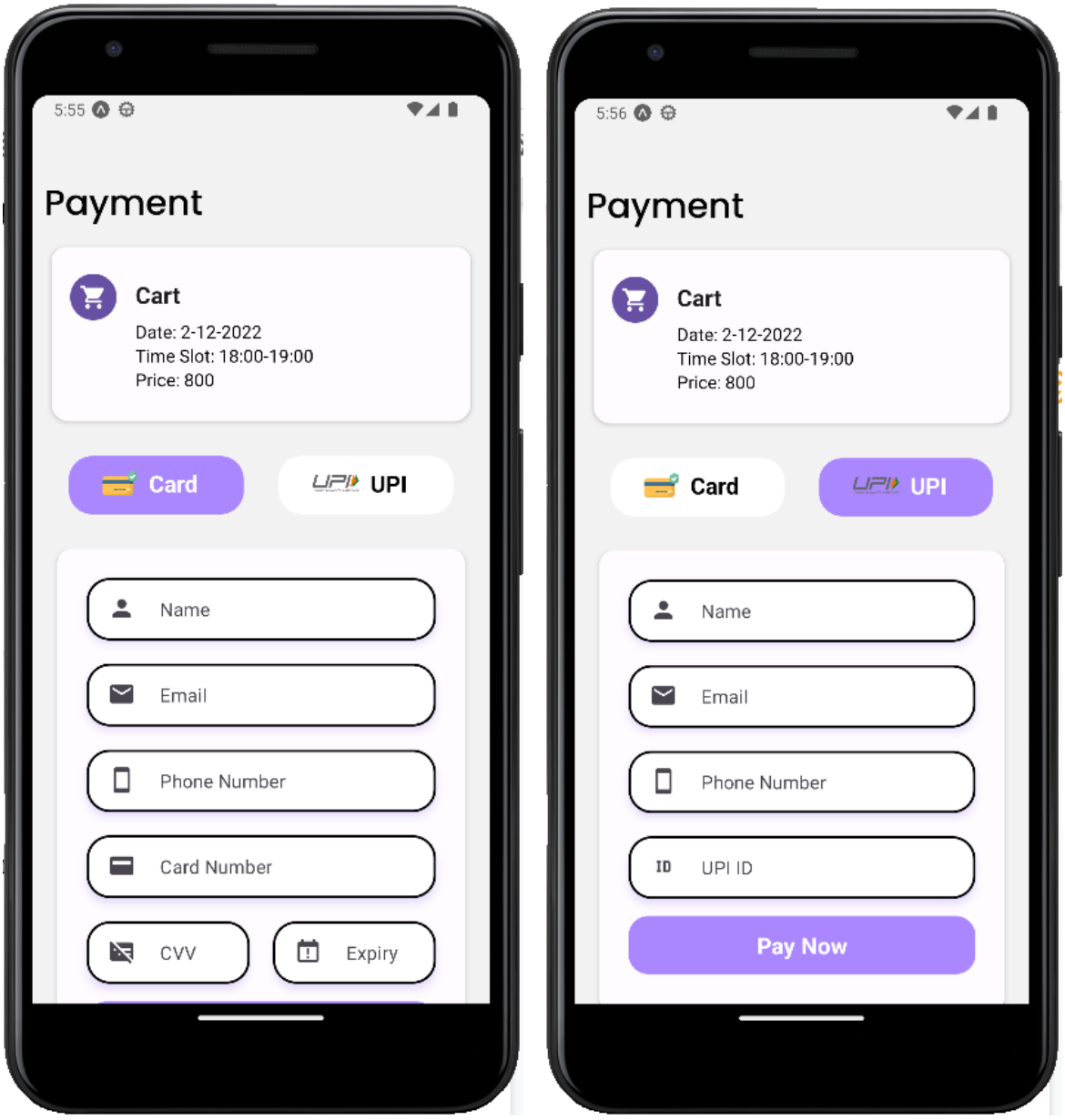


15) Here you can select a slot by clicking the slot and it will change its color to green

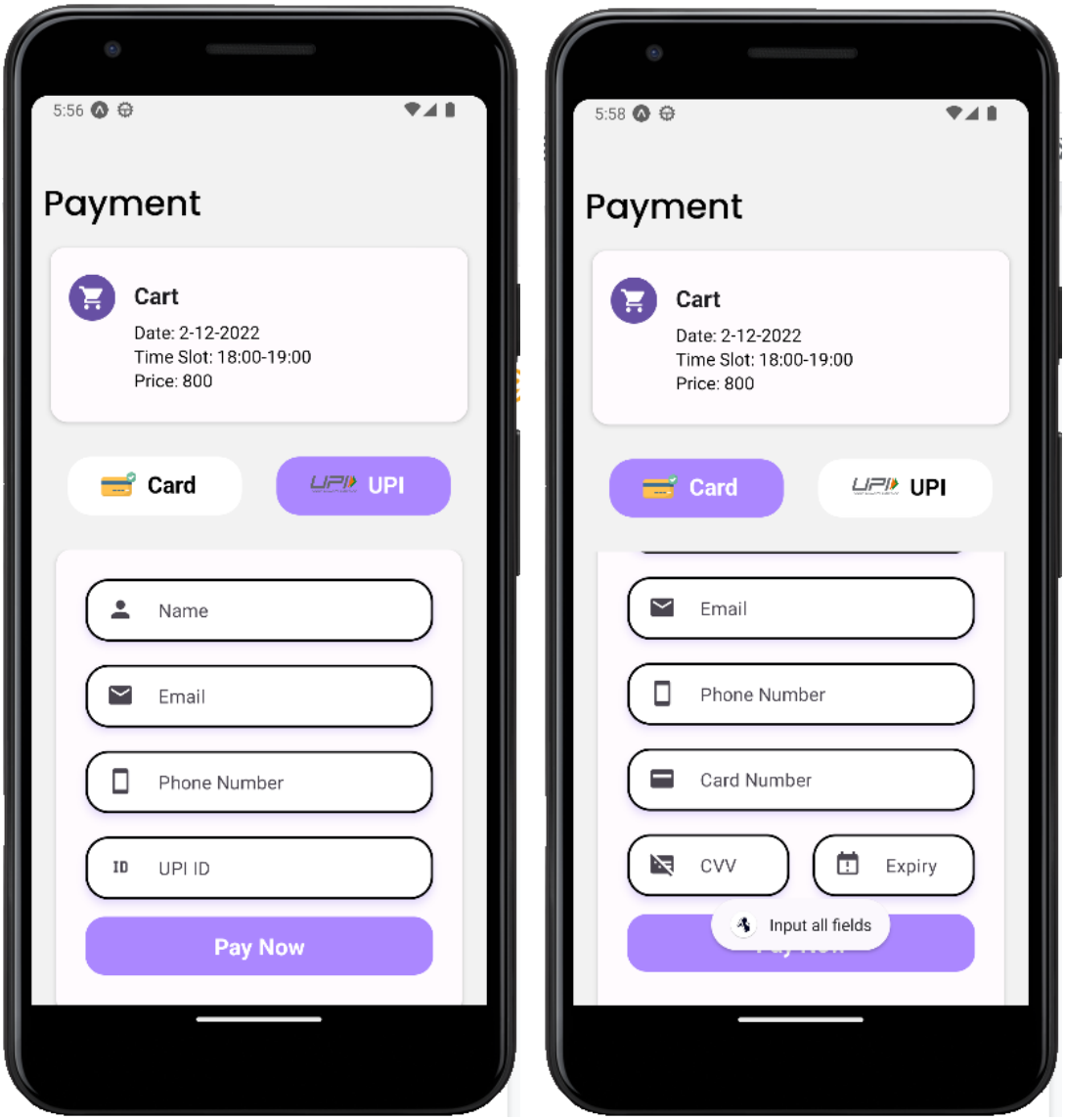


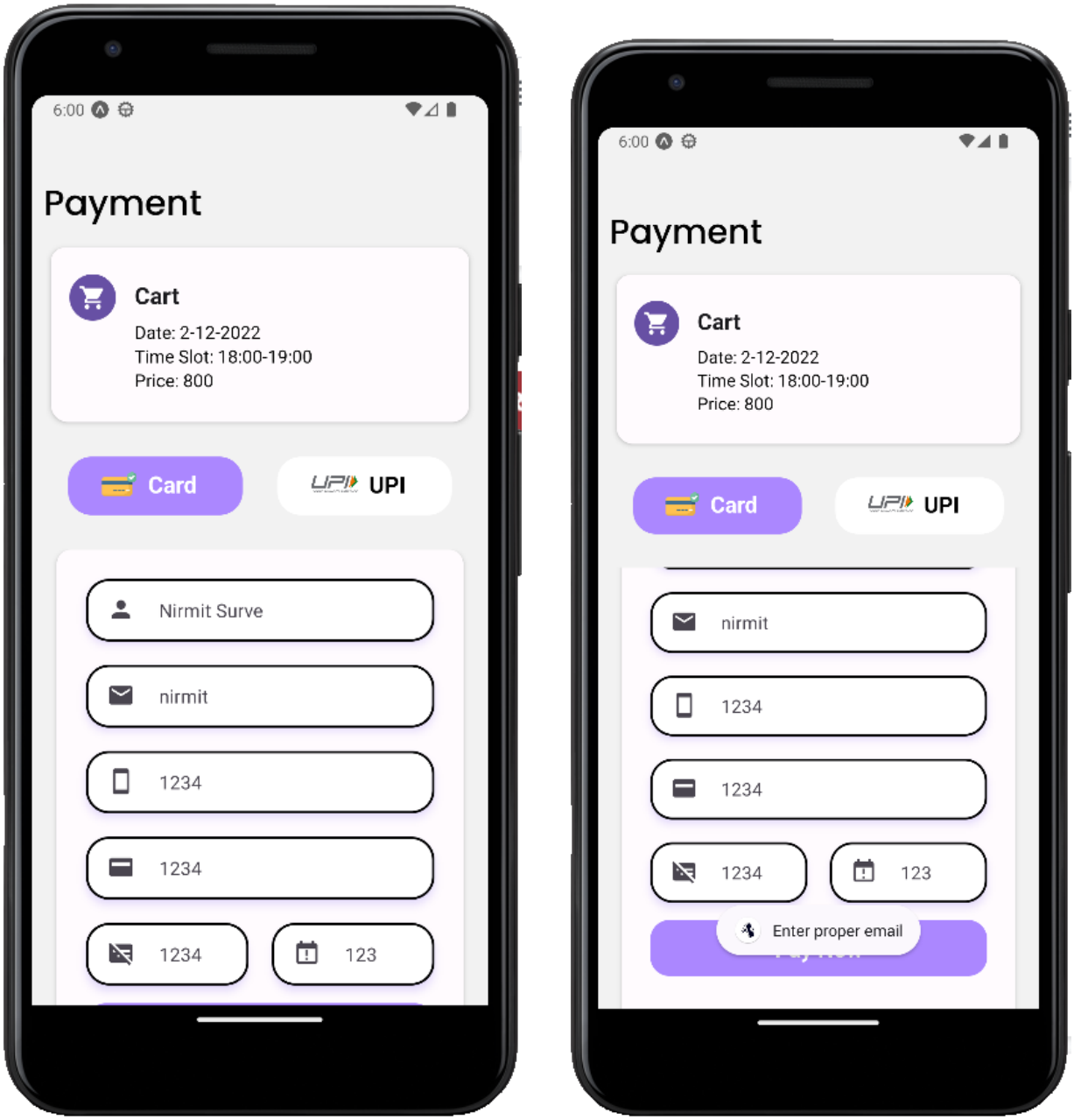
16) We selected the slot and press book now button. After this the user is directed to the payment page

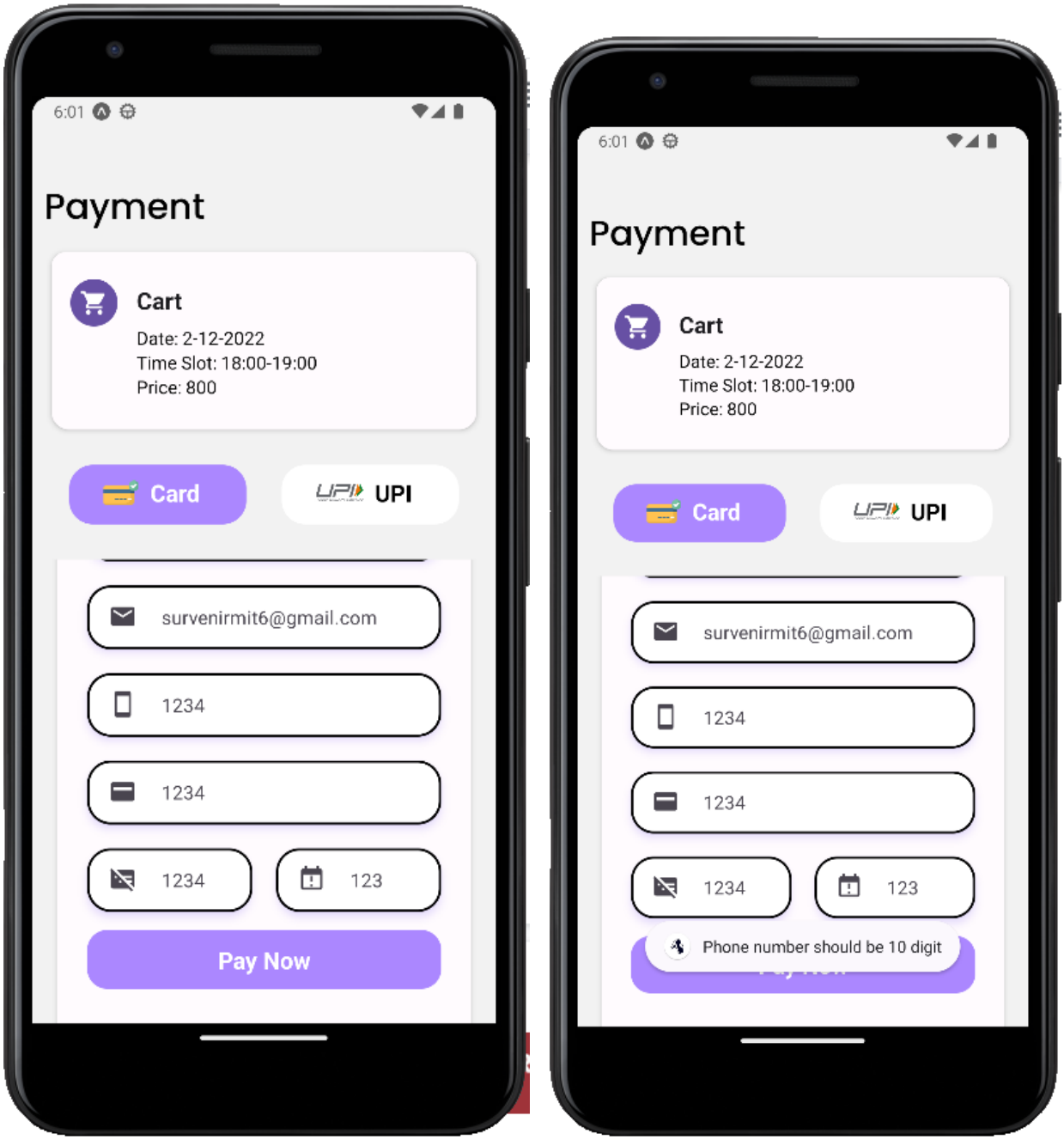
17) This is the payment page. It has 2 methods for payment: Card and UPI

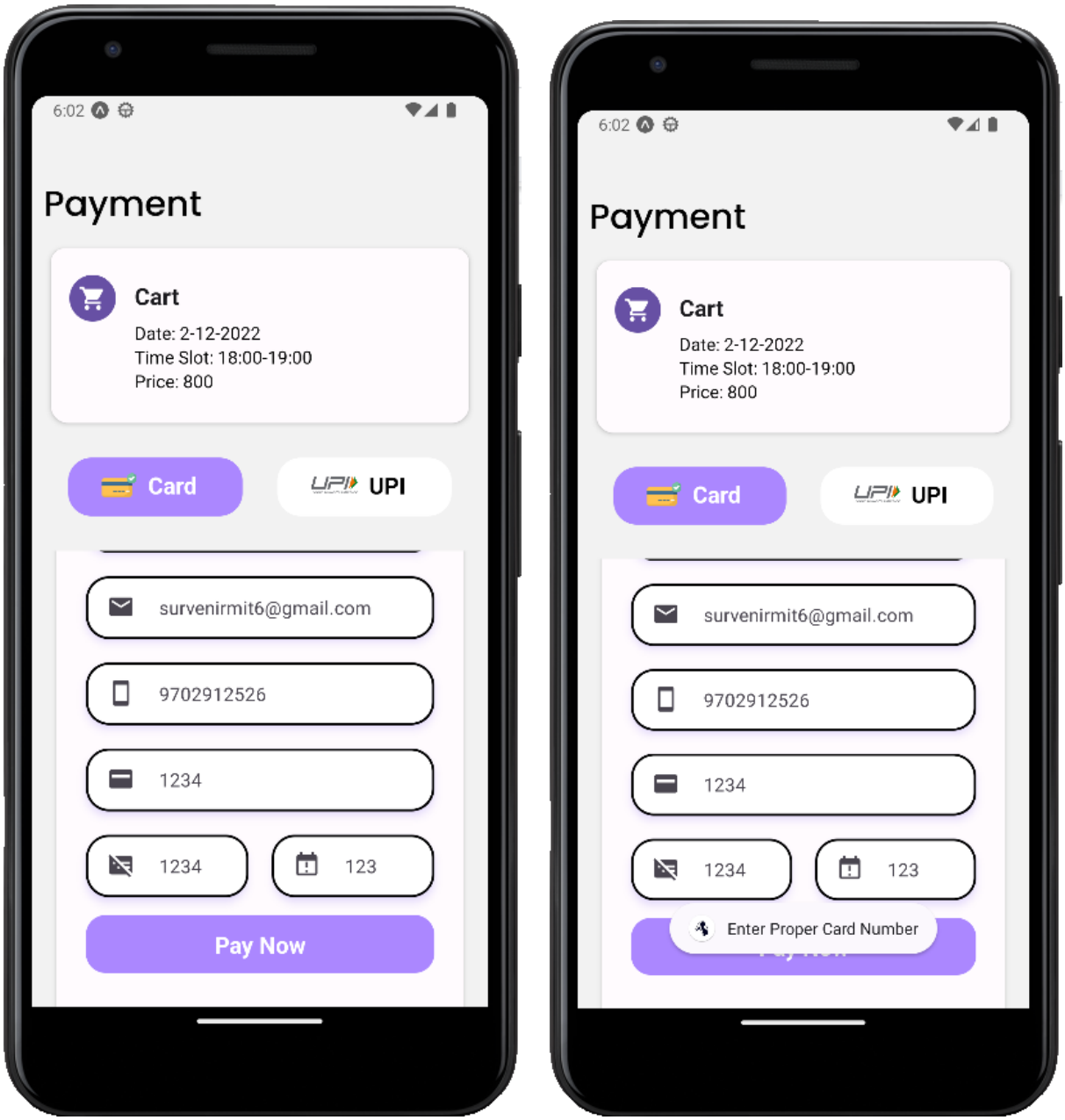


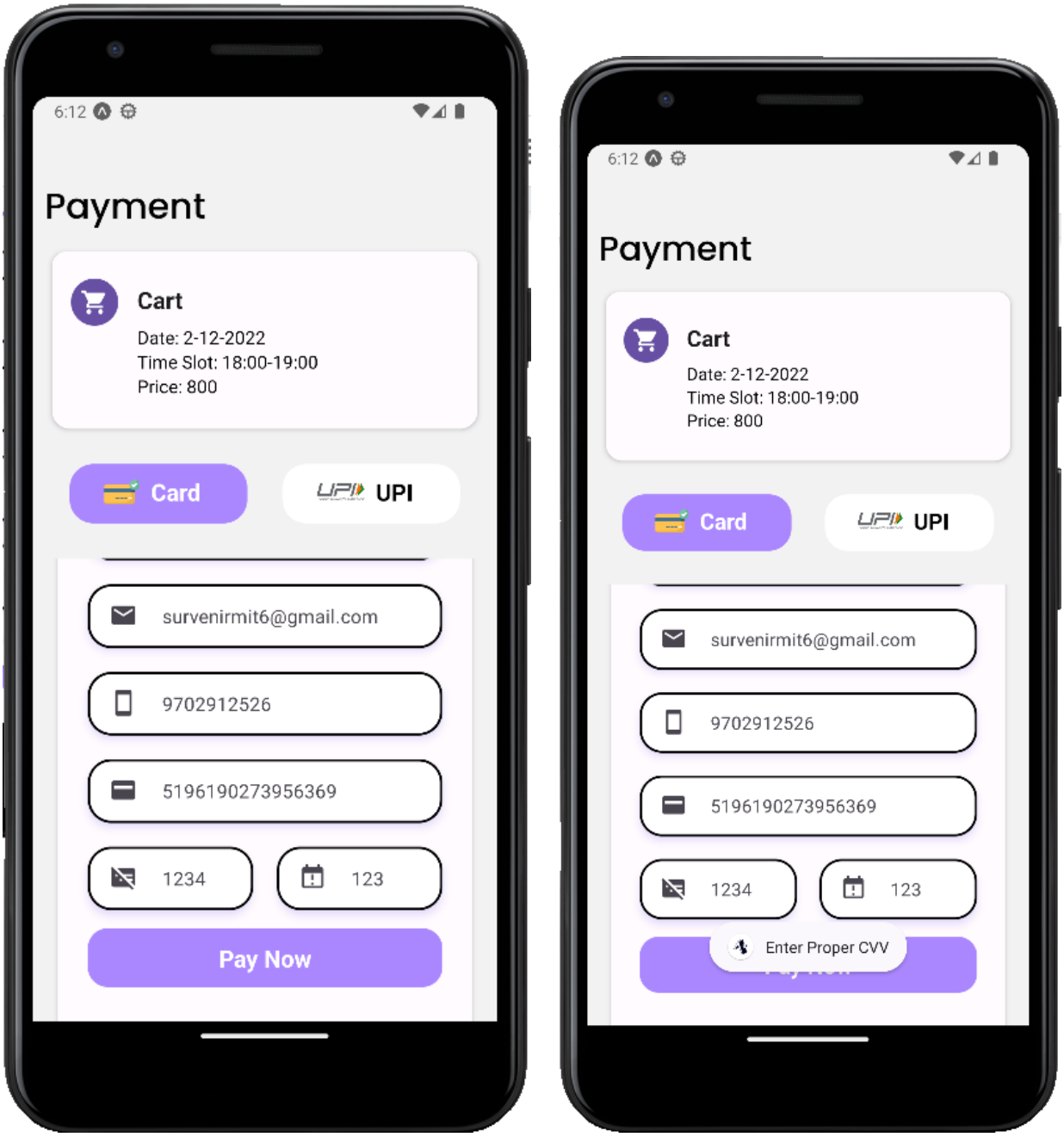
18) This page contains validations for
    a) Empty field
    b) Email
    c) Phone number
    d) Card Number
    e) CVV
    f) Expiry
    g) UPI ID

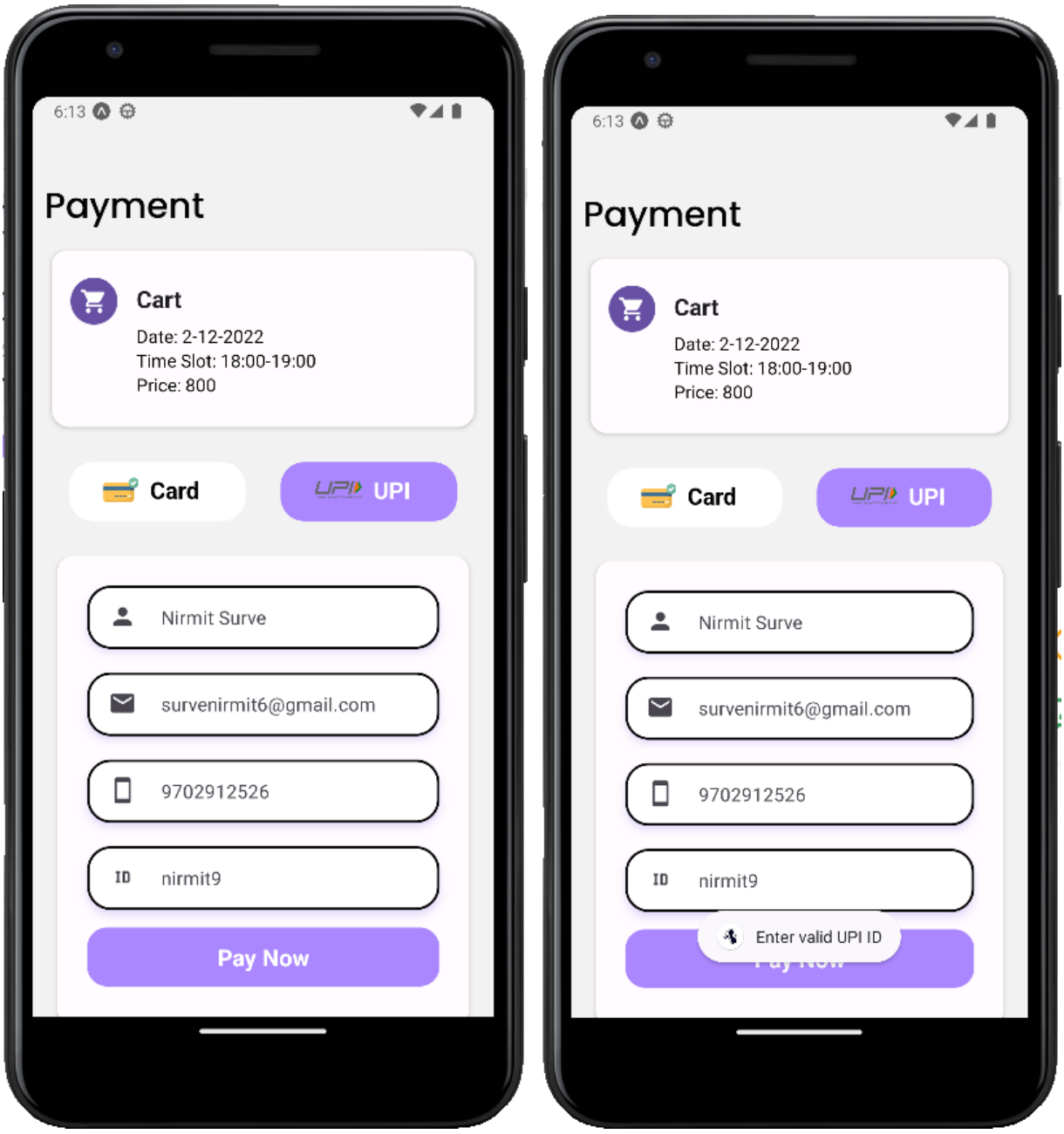



6:00
Payment
Cart
Date: 2-12-2022
Time Slot: 18:00-19:00
Price: 800
Card
UPI
Nirmit Surve
nirmit
1234
1234
1234
123
6:00
Payment
Cart
Date: 2-12-2022
Time Slot: 18:00-19:00
Price: 800
Card
UPI
nirmit
1234
1234
1234
123
Enter proper email

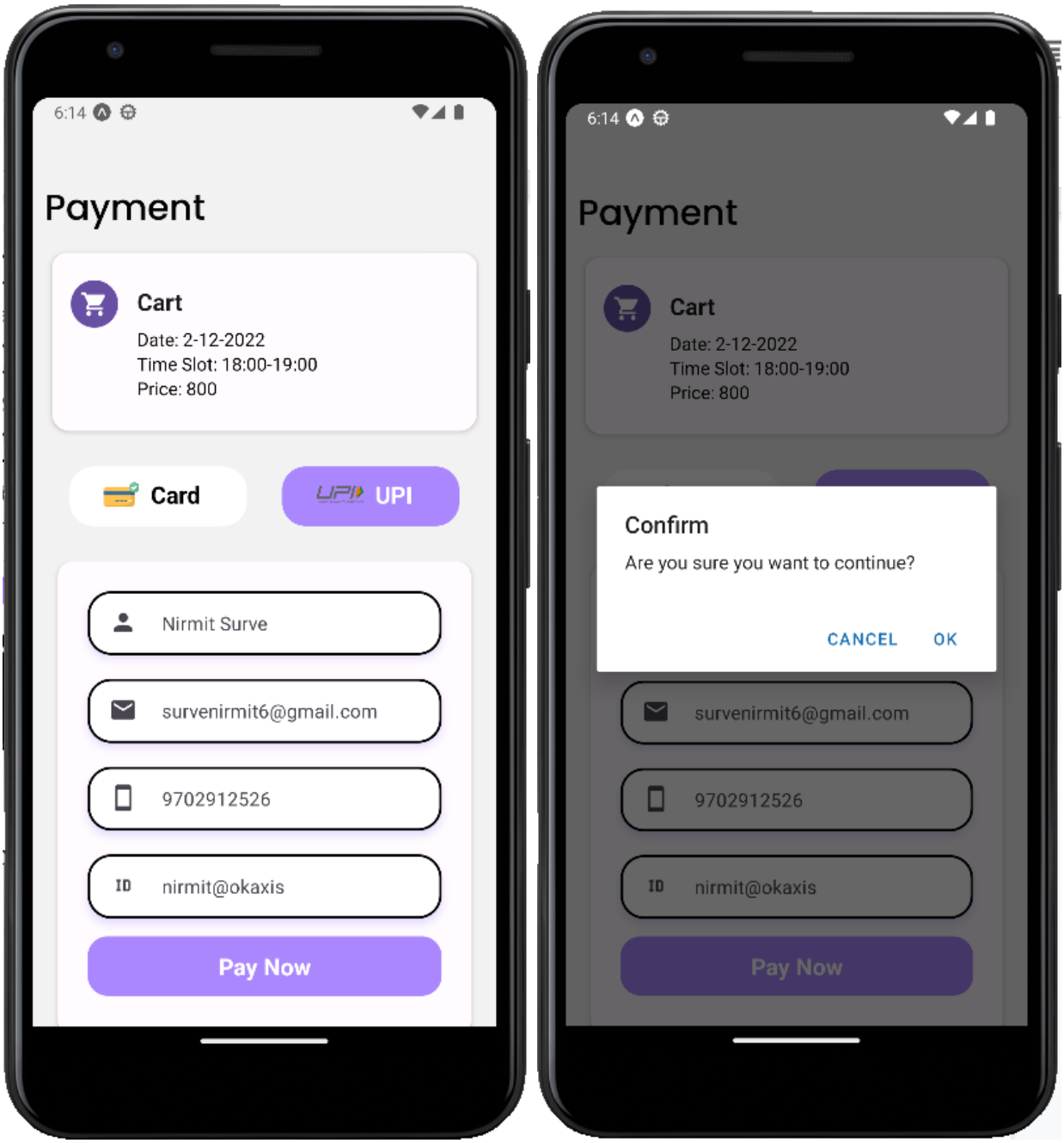

6:01
Payment
Cart
Date: 2-12-2022
Time Slot: 18:00-19:00
Price: 800
Card
UPI
survenirmit6@gmail.com
1234
1234
1234
123
Pay Now
6:01
Payment
Cart
Date: 2-12-2022
Time Slot: 18:00-19:00
Price: 800
Card
UPI
survenirmit6@gmail.com
1234
1234
1234
123
Phone number should be 10 digit

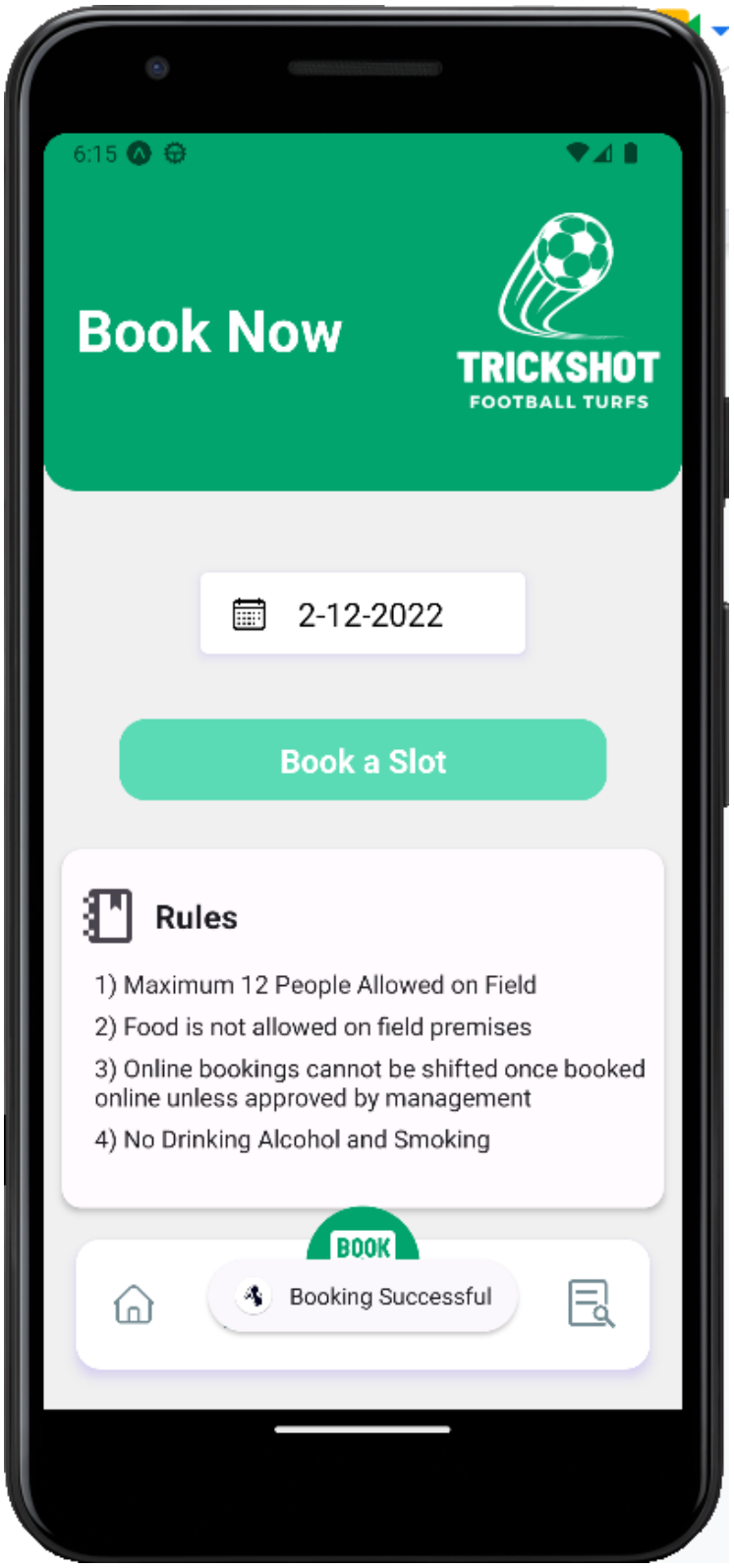

6:02
Payment
Cart
Date: 2-12-2022
Time Slot: 18:00-19:00
Price: 800
Card
UPI
survenirmit6@gmail.com
9702912526
1234
1234
123
Pay Now
6:02
Payment
Cart
Date: 2-12-2022
Time Slot: 18:00-19:00
Price: 800
Card
UPI
survenirmit6@gmail.com
9702912526
1234
1234
123
Enter Proper Card Number


6:12
Payment
Cart
Date: 2-12-2022
Time Slot: 18:00-19:00
Price: 800
Card
UPI
survenirmit6@gmail.com
9702912526
5196190273956369
1234
123
Pay Now
6:12
Payment
Cart
Date: 2-12-2022
Time Slot: 18:00-19:00
Price: 800
Card
UPI
survenirmit6@gmail.com
9702912526
5196190273956369
1234
123
Enter Proper CVV



If all information is entered correctly we can book a slot by clicking the pay now button


6:14
Payment
Cart
Date: 2-12-2022
Time Slot: 18:00-19:00
Price: 800
Card
UPI
Nirmit Surve
survenirmit6@gmail.com
9702912526
nirmit@okaxis
Pay Now
6:14
Payment
Cart
Date: 2-12-2022
Time Slot: 18:00-19:00
Price: 800
Confirm
Are you sure you want to continue?
CANCEL
OK
survenirmit6@gmail.com
9702912526
nirmit@okaxis
Pay Now

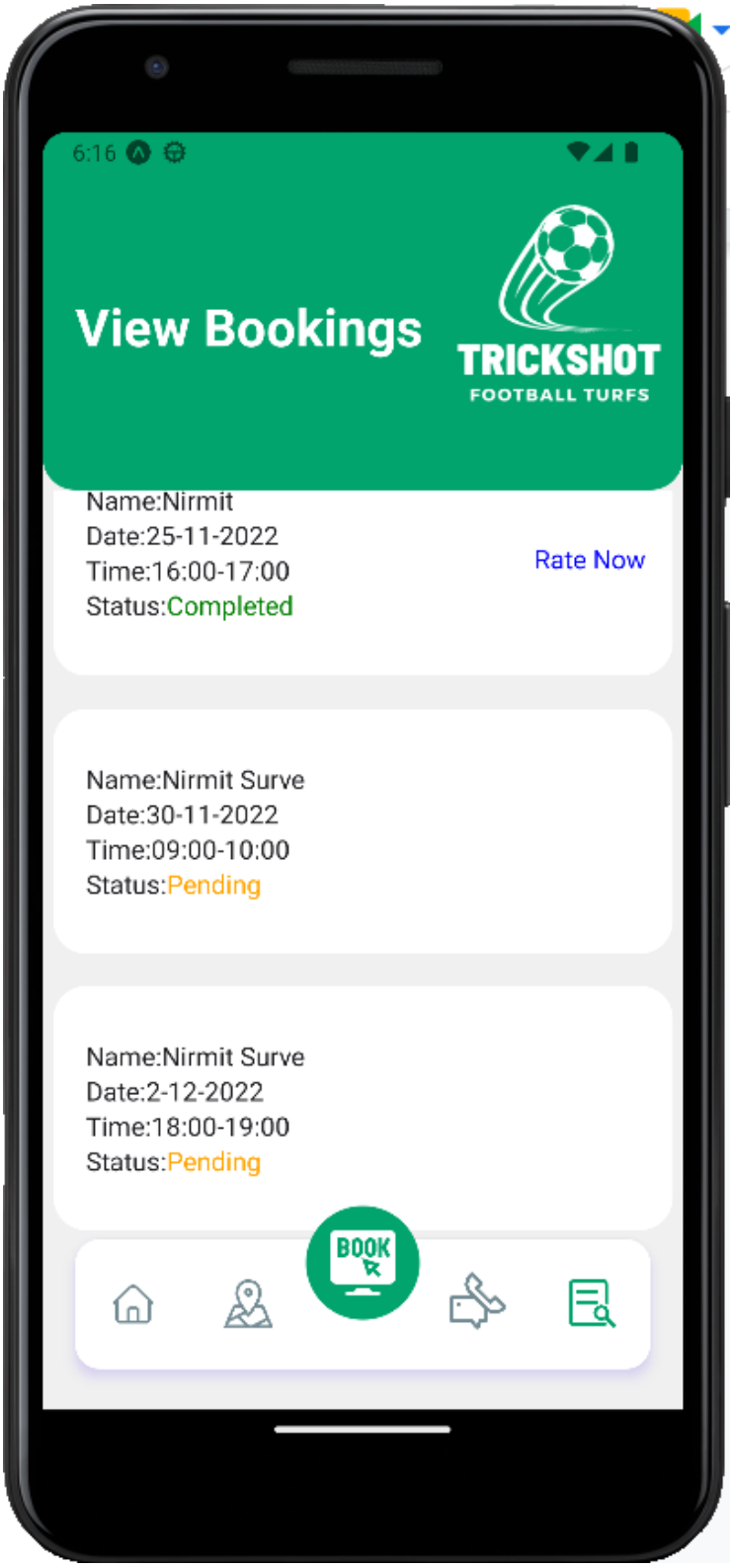

6:15
Book Now
TRICKSHOT
FOOTBALL TURFS
2-12-2022
Book a Slot
Rules
1) Maximum 12 People Allowed on Field
2) Food is not allowed on field premises
3) Online bookings cannot be shifted once booked online unless approved by management
4) No Drinking Alcohol and Smoking
BOOK
Booking Successful

We can see the booking is updated in the database


```
_id: ObjectId('6384ad5a1751611cf34a6d4f')
username: "Nirmit"
name: "Nirmit Surve"
date: "2-12-2022"
time: "18:00-19:00"
email: "survenirmit6@gmail.com"
phoneno: "9702912526"
_class: "com.example.booking.model.TurfModel"
```


---

19) We can see the booking is updated in view booking page

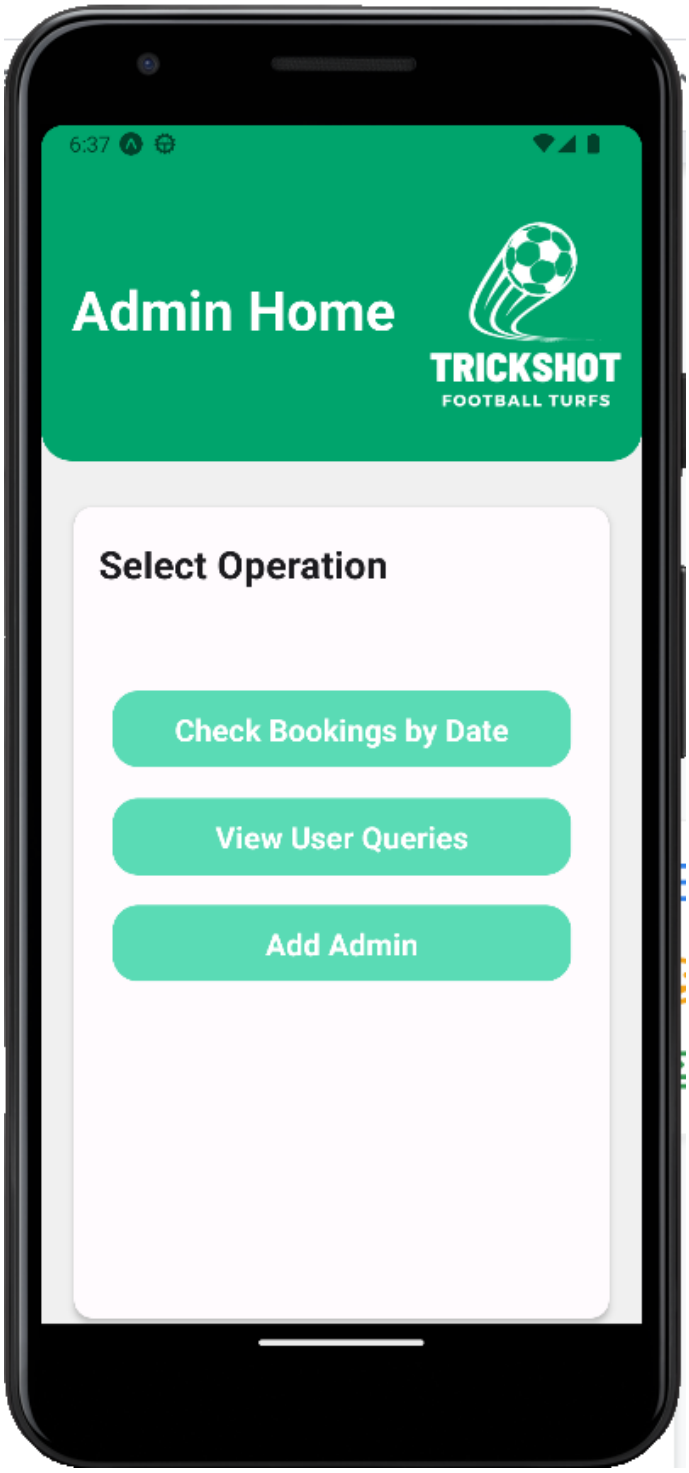


- **Admin App**

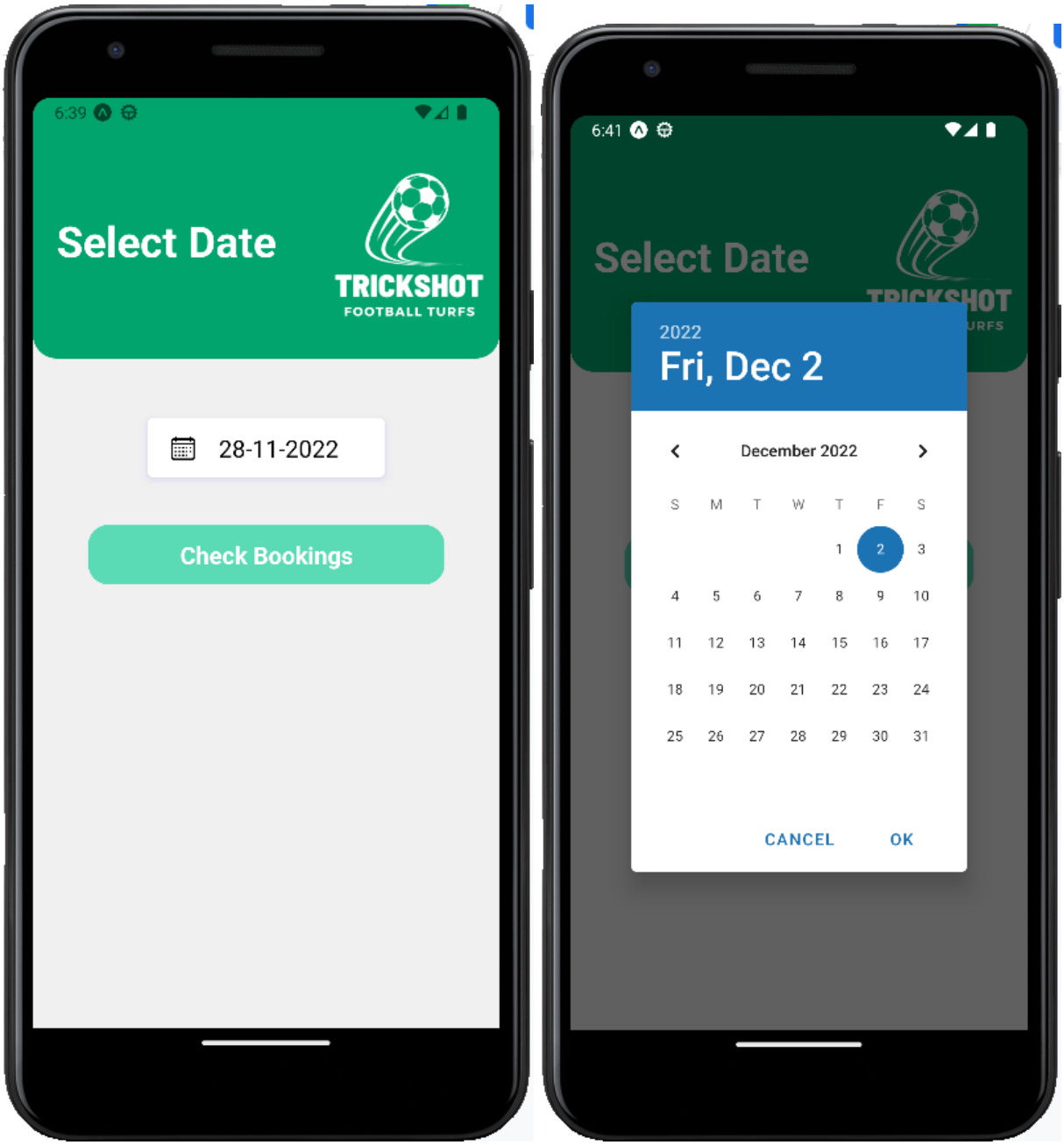
1) This is the landing page for the admin app. It is the login page. This page has the same validations as the user app login page.

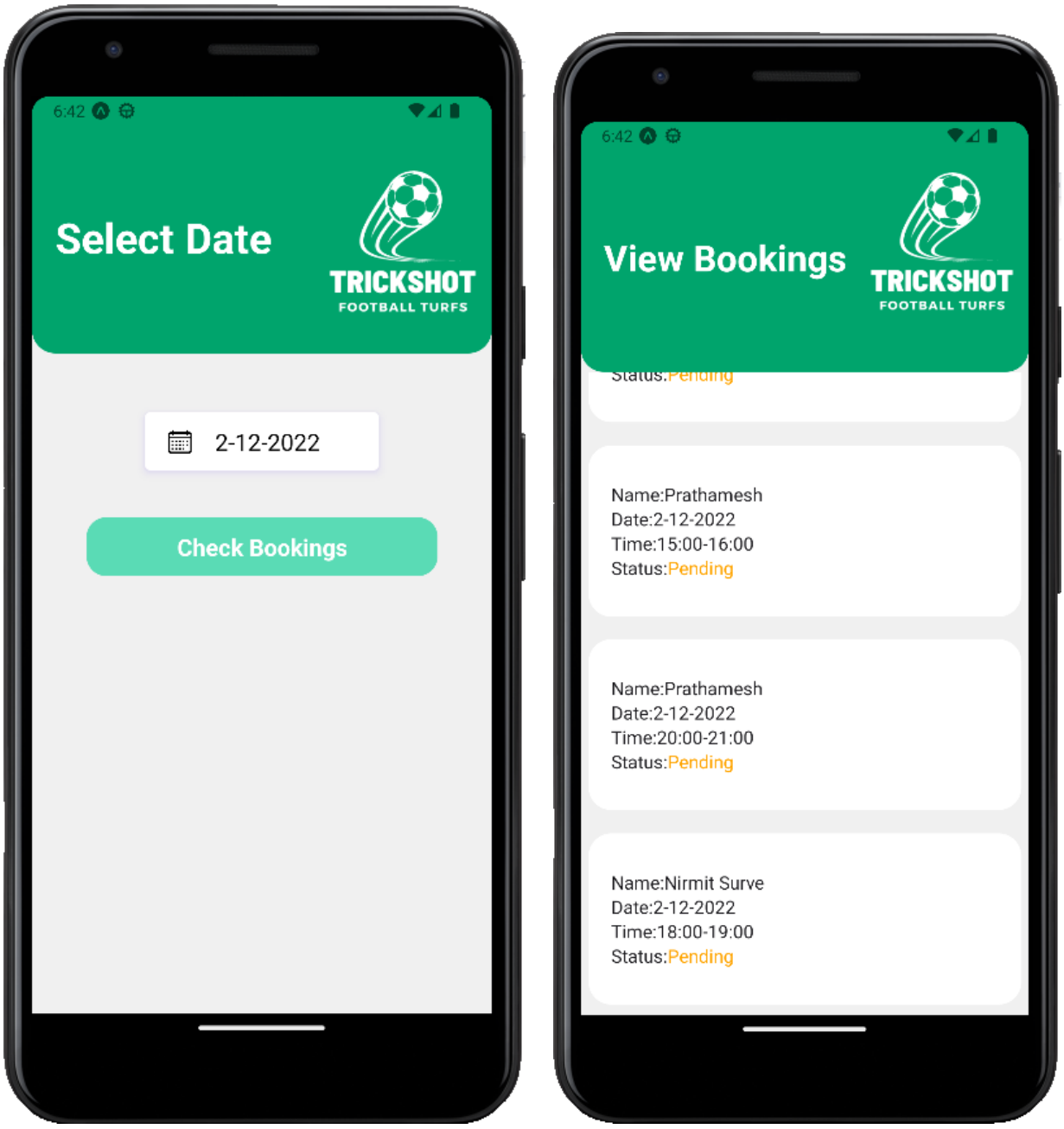
   Here I am logging in by using admin credentials

2) This is the admin homepage. It consists of three options
    a) Check Bookings by Date
    b) View User Queries
    c) Add Admin



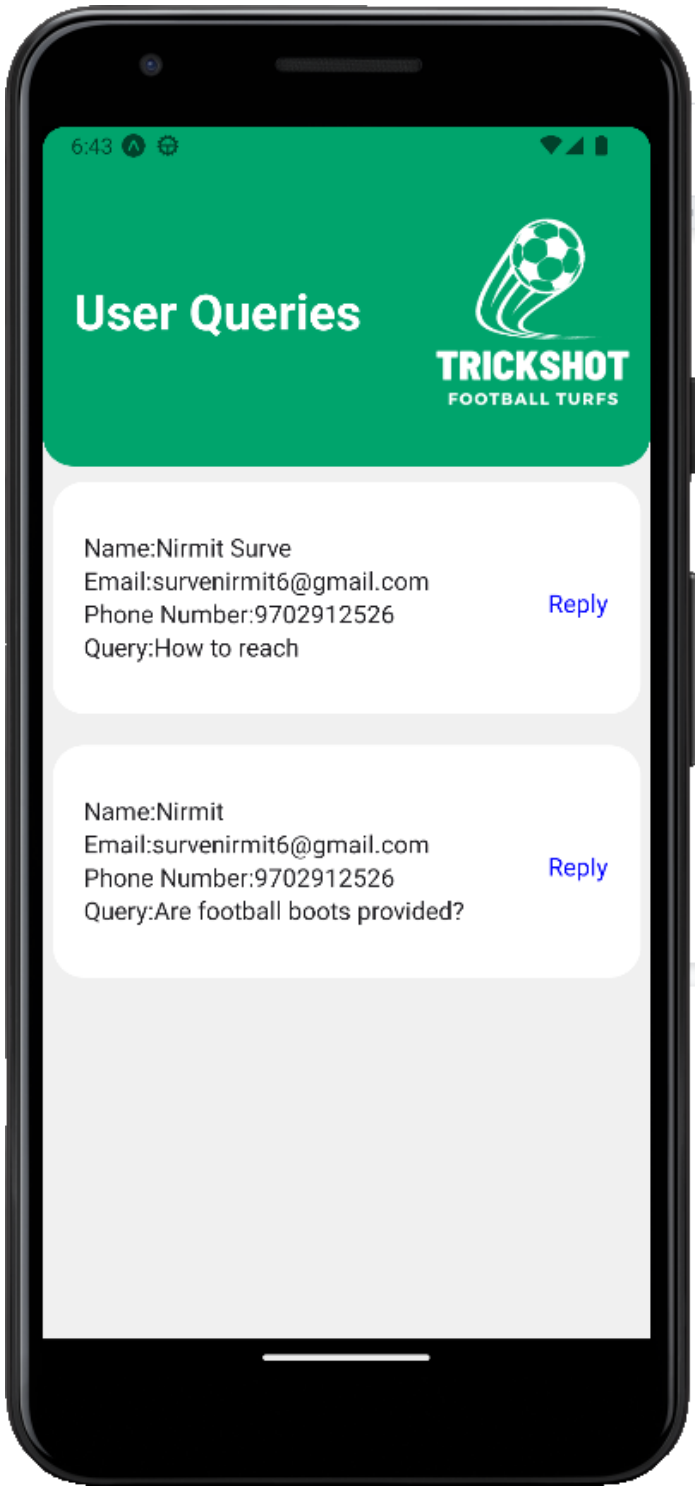
a) Check Bookings by date:
This page shows bookings for a particular date



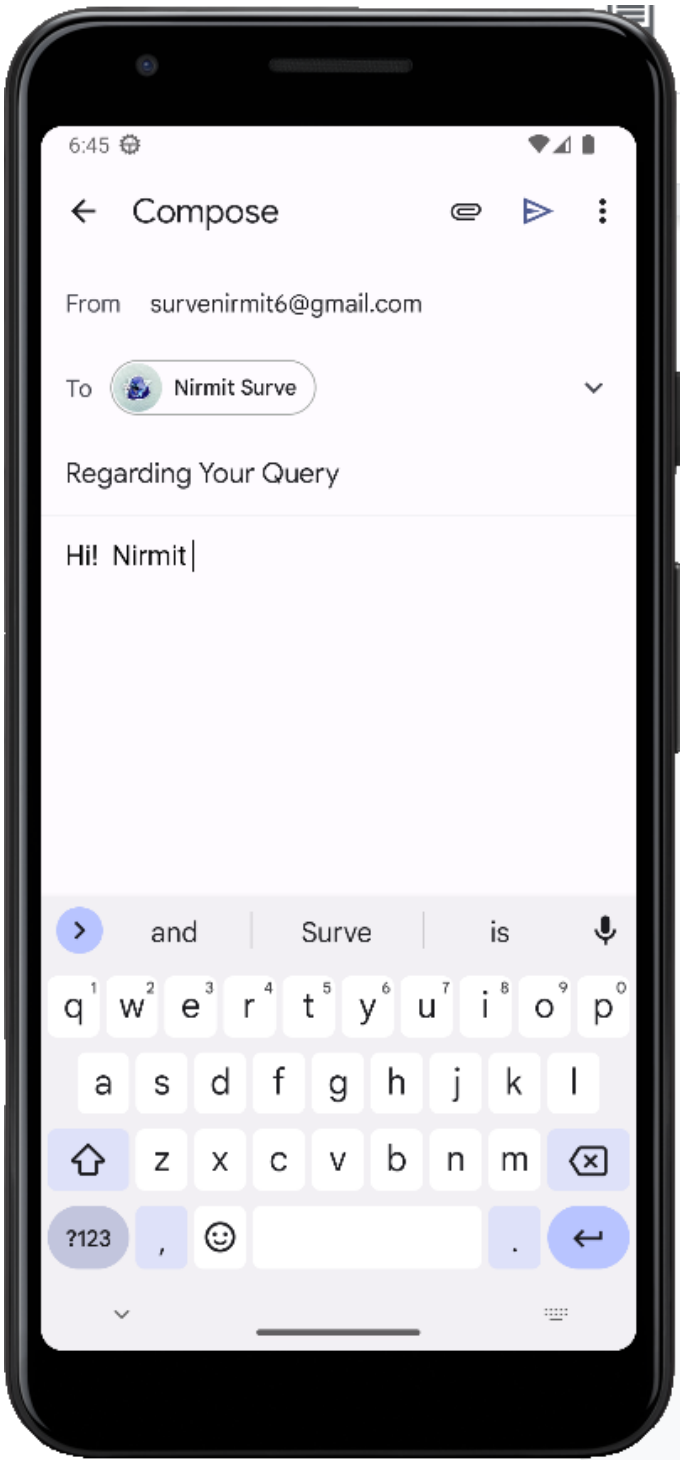

6:42
Select Date
TRICKSHOT
FOOTBALL TURFS
2-12-2022
Check Bookings
6:42
View Bookings
TRICKSHOT
FOOTBALL TURFS
Status:Pending
Name:Prathamesh
Date:2-12-2022
Time:15:00-16:00
Status:Pending
Name:Prathamesh
Date:2-12-2022
Time:20:00-21:00
Status:Pending
Name:Nirmit Surve
Date:2-12-2022
Time:18:00-19:00
Status:Pending

b) View User Queries:

This page displays user queries

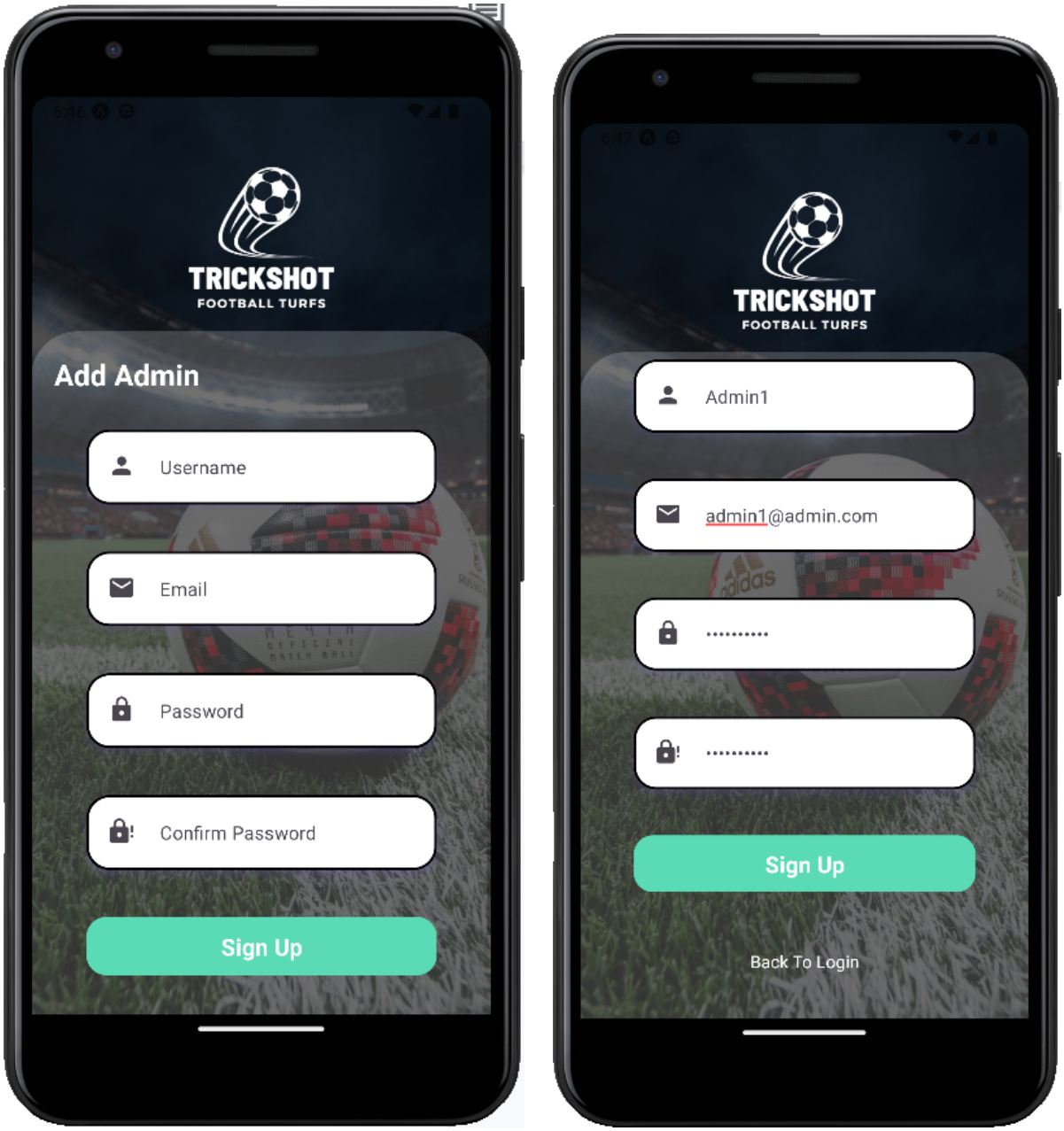


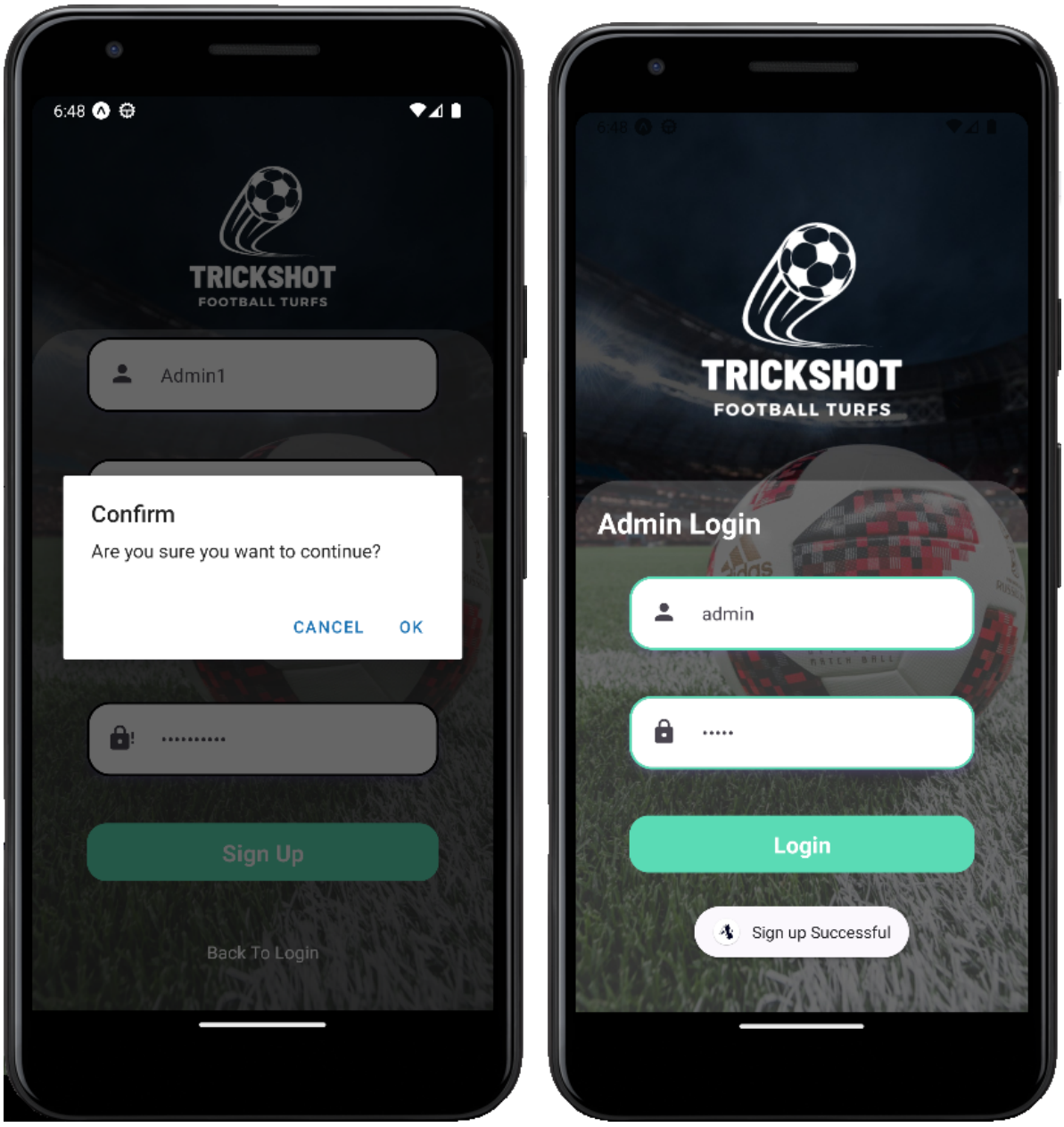
If the turf owner wants to reply to a particular query he can select the reply option and the mail app will open with all the details autofilled. The owner just has to write the response for the query



c) Add Admin

This page is used to add another admin for admin access. It has the same validations as the user signup page. Here I am going to add another Admin



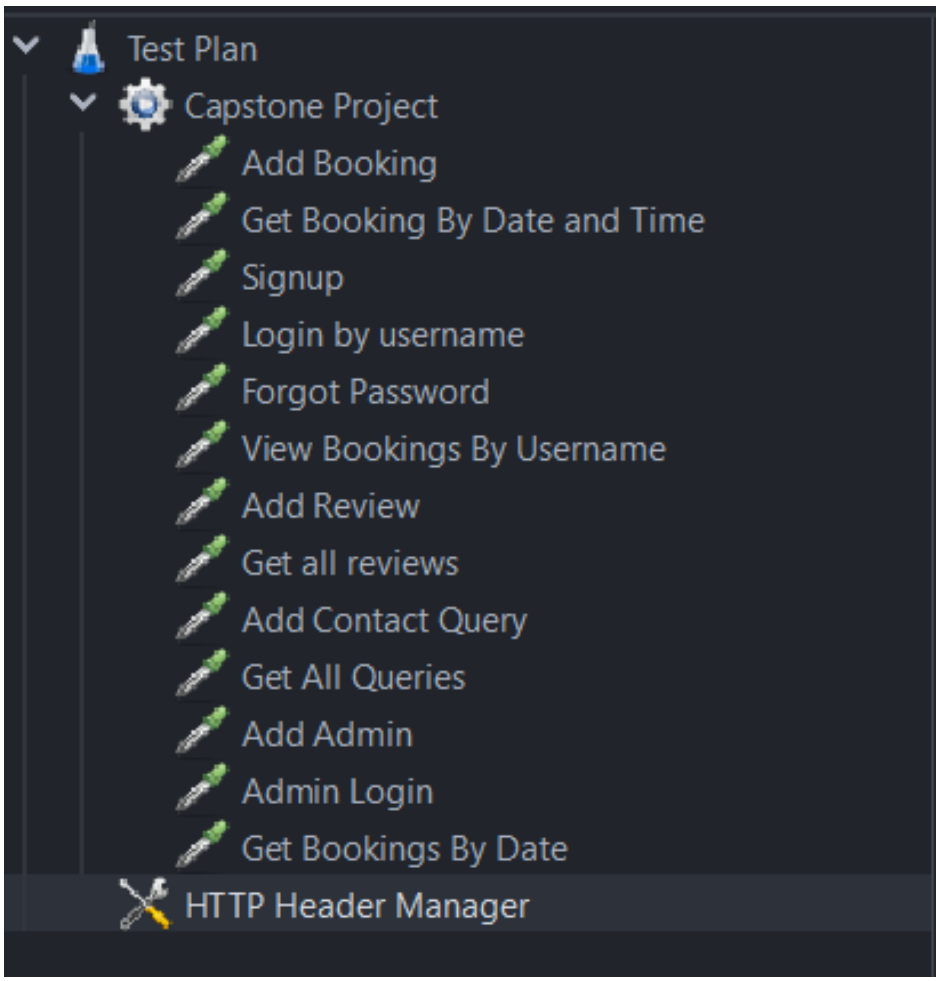

6:48
TRICKSHOT
FOOTBALL TURFS
Admin1
Confirm
Are you sure you want to continue?
CANCEL
OK
Sign Up
Back To Login
TRICKSHOT
FOOTBALL TURFS
Admin Login
admin
Login
Sign up Successful

We can see the admin is added into the database


```
_id: "Admin1"
email: "admin1@admin.com"
password: "Admin@1234"
_class: "com.example.booking.model.AdminModel"
```

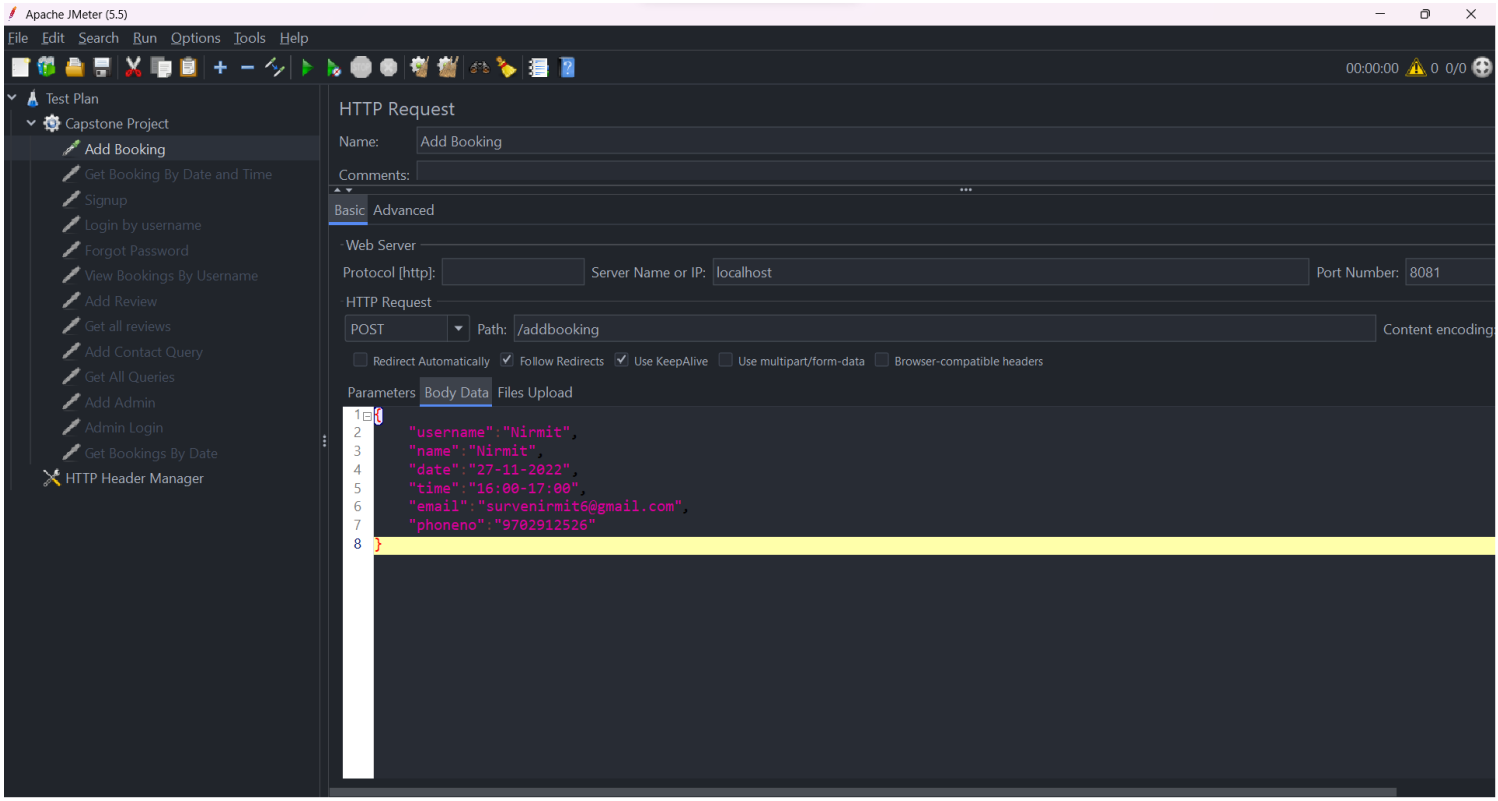
Performance Testing:

These are the methods we are going to test in Jmeter:

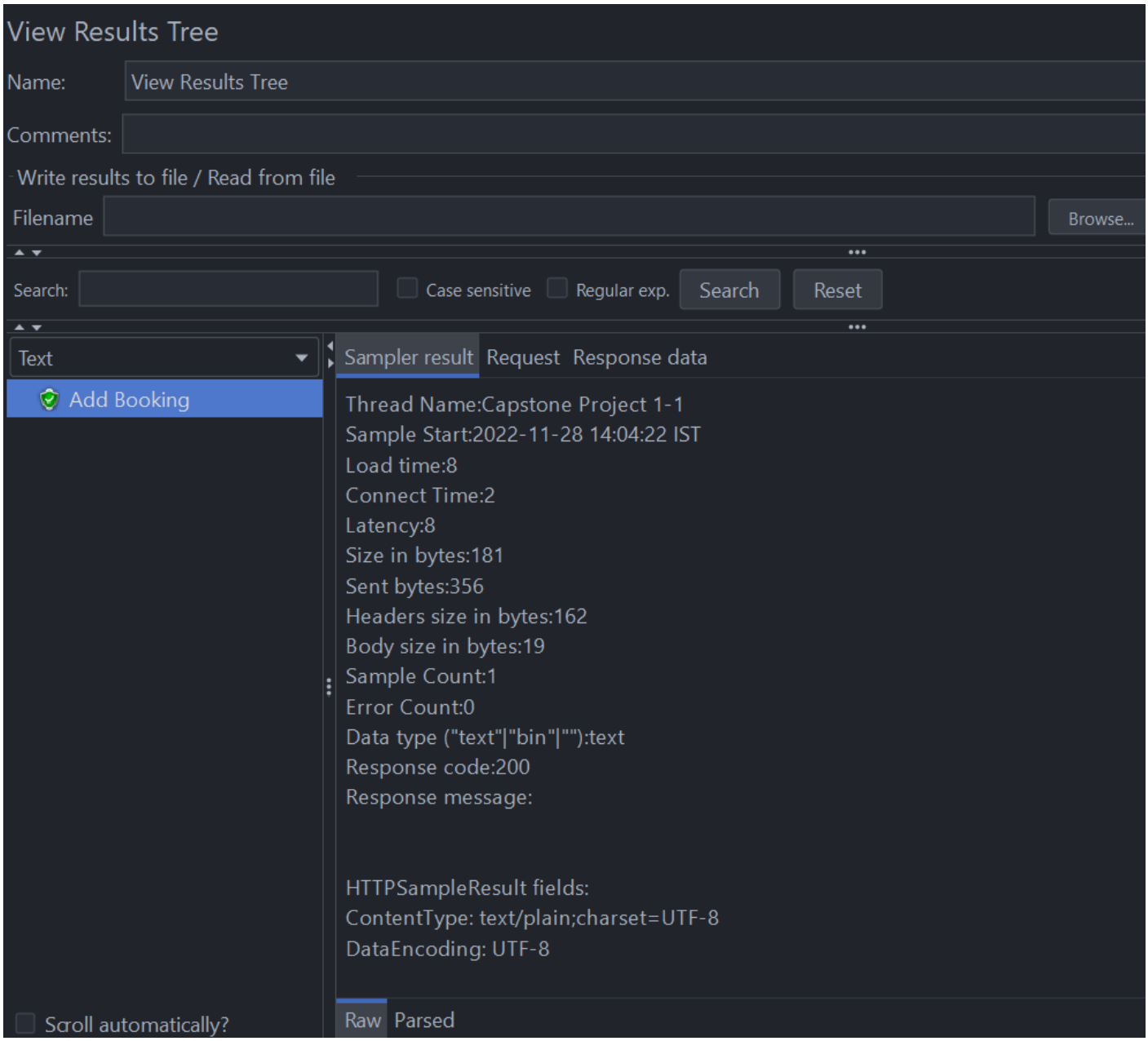


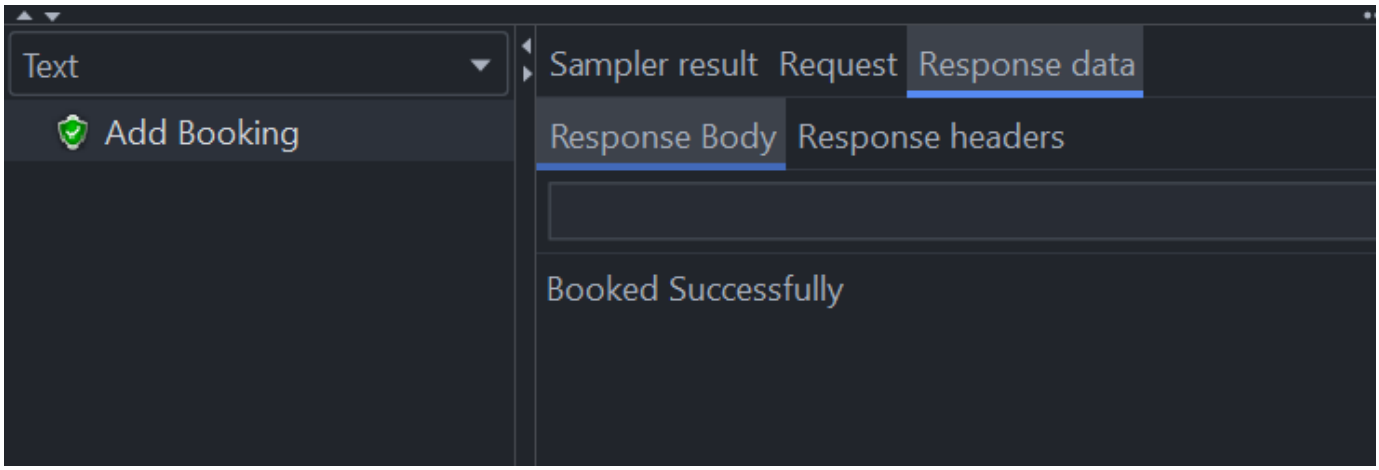
1) Add Booking:

This method has no fail case.



After running the request we can see the result in the result tree

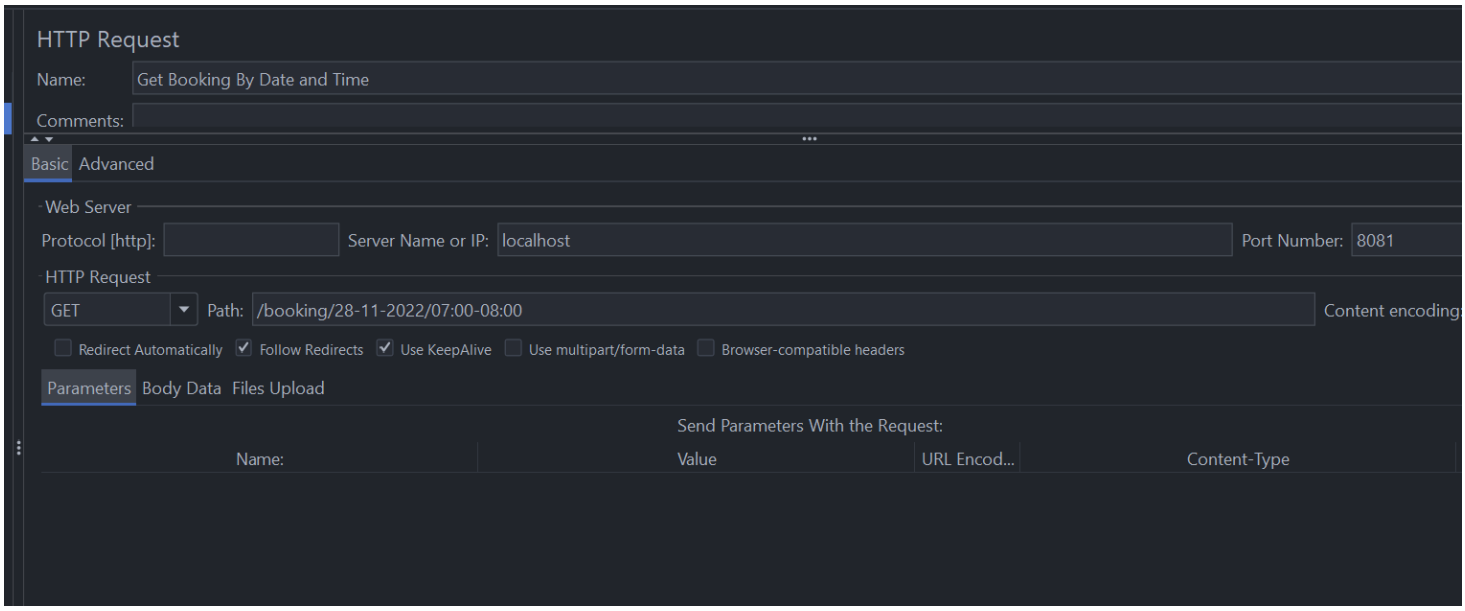

View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Add Booking
Sampler result
Request
Response data
Thread Name:Capstone Project 1-1
Sample Start:2022-11-28 14:04:22 IST
Load time:8
Connect Time:2
Latency:8
Size in bytes:181
Sent bytes:356
Headers size in bytes:162
Body size in bytes:19
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:
HTTPSampleResult fields:
ContentType: text/plain;charset=UTF-8
DataEncoding: UTF-8
Scroll automatically?
Raw
Parsed

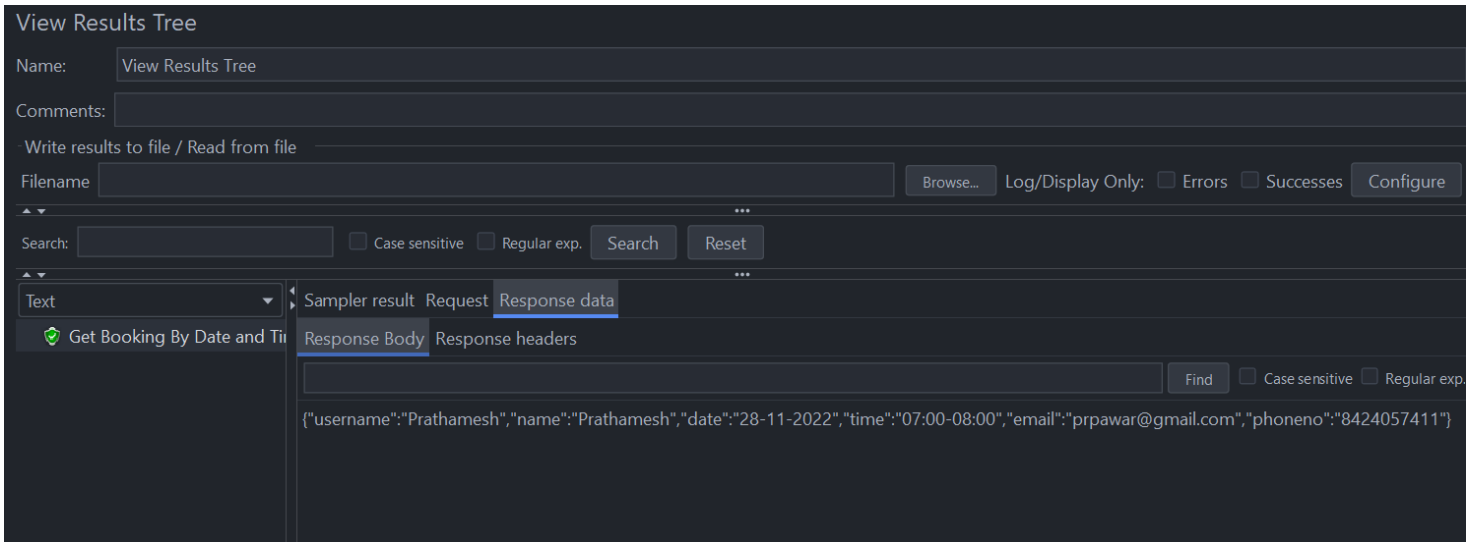


2) Get Booking by Date and Time:

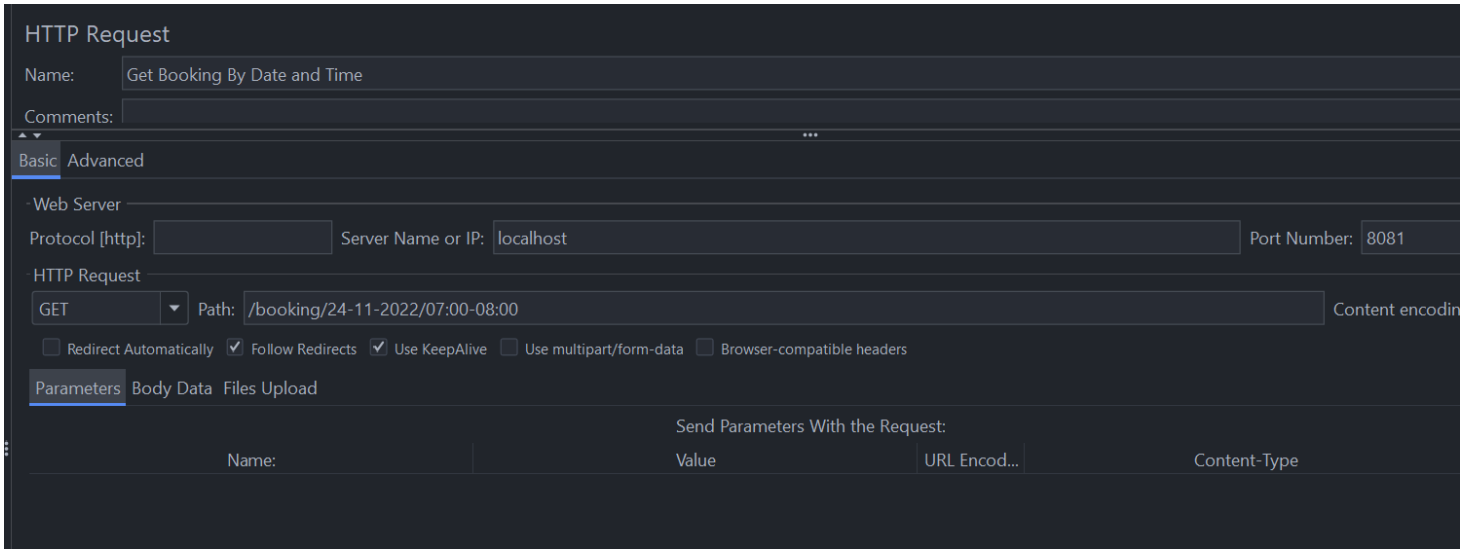
ID	Description	Data	Expected Result	Actual Result	Pass/Fail
1	Get booking info for a date and slot time	Date=28-11-2022 Time=07:00-08:00	Returns info about the booking	As Expected	Pass
2	Get booking info for a date and slot time	Date=24-11-2022 Time=07:00-08:00	Returns Null	As Expected	Pass

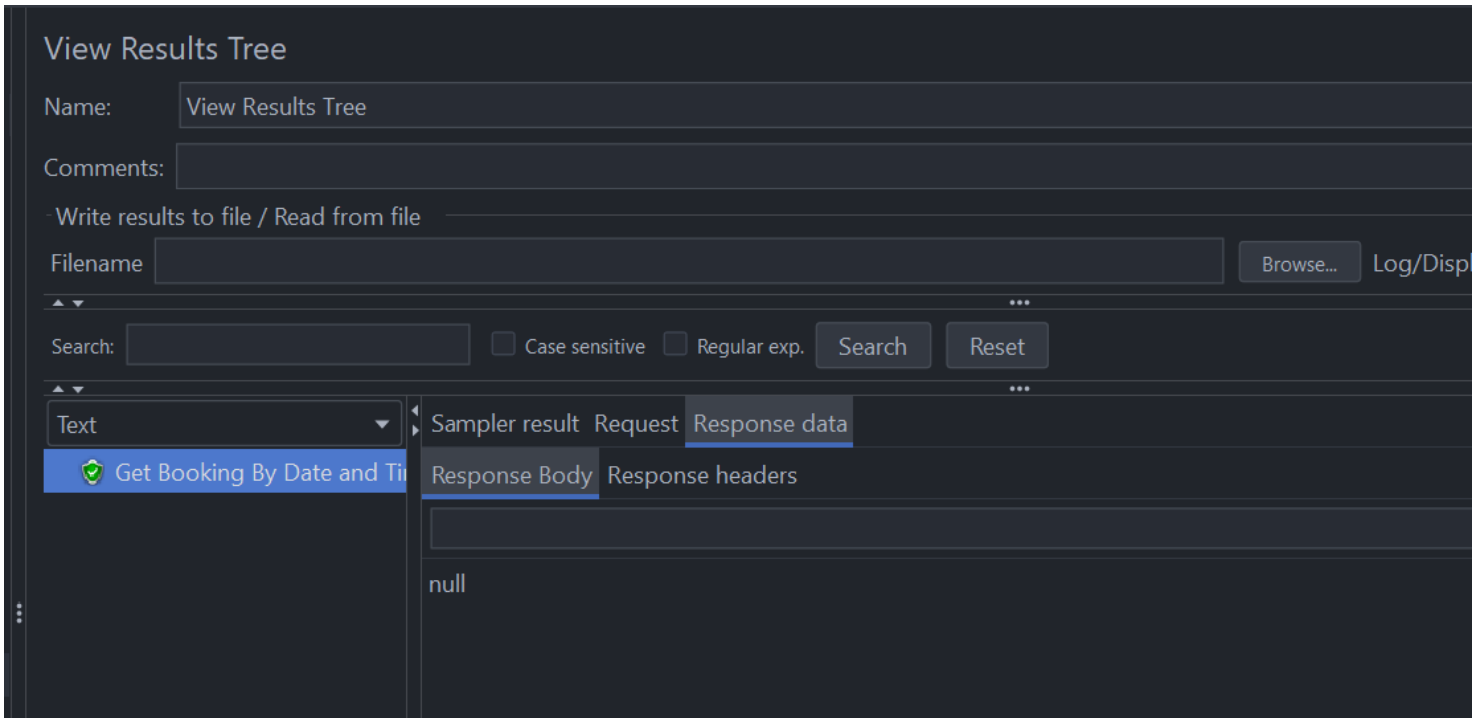
a) Test ID 1:


HTTP Request
Name:
Get Booking By Date and Time
Comments:
Basic
Advanced
Web Server
Protocol [http]:
Server Name or IP:
localhost
Port Number:
8081
HTTP Request
GET
Path:
/booking/28-11-2022/07:00-08:00
Content encoding:
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters
Body Data
Files Upload
Send Parameters With the Request:
Name:
Value
URL Encod...
Content-Type


View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Get Booking By Date and Ti
Sampler result
Request
Response data
Response Body
Response headers
Find
Case sensitive
Regular exp.
{"username":"Prathamesh","name":"Prathamesh","date":"28-11-2022","time":"07:00-08:00","email":"prpawar@gmail.com","phoneno":"8424057411"}

b) Test ID 2:

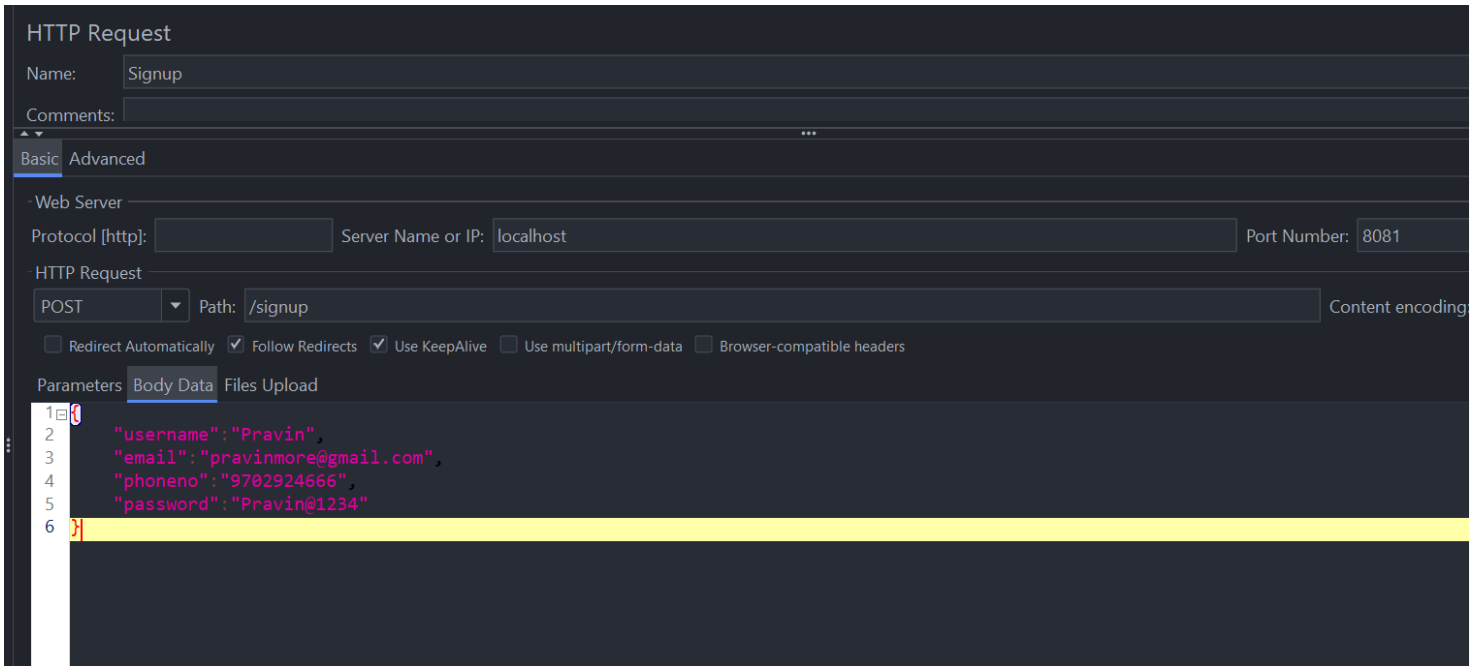

HTTP Request
Name:
Get Booking By Date and Time
Comments:
Basic
Advanced
Web Server
Protocol [http]:
Server Name or IP:
localhost
Port Number:
8081
HTTP Request
GET
Path:
/booking/24-11-2022/07:00-08:00
Content encodin
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters
Body Data
Files Upload
Send Parameters With the Request:
Name:
Value
URL Encod...
Content-Type

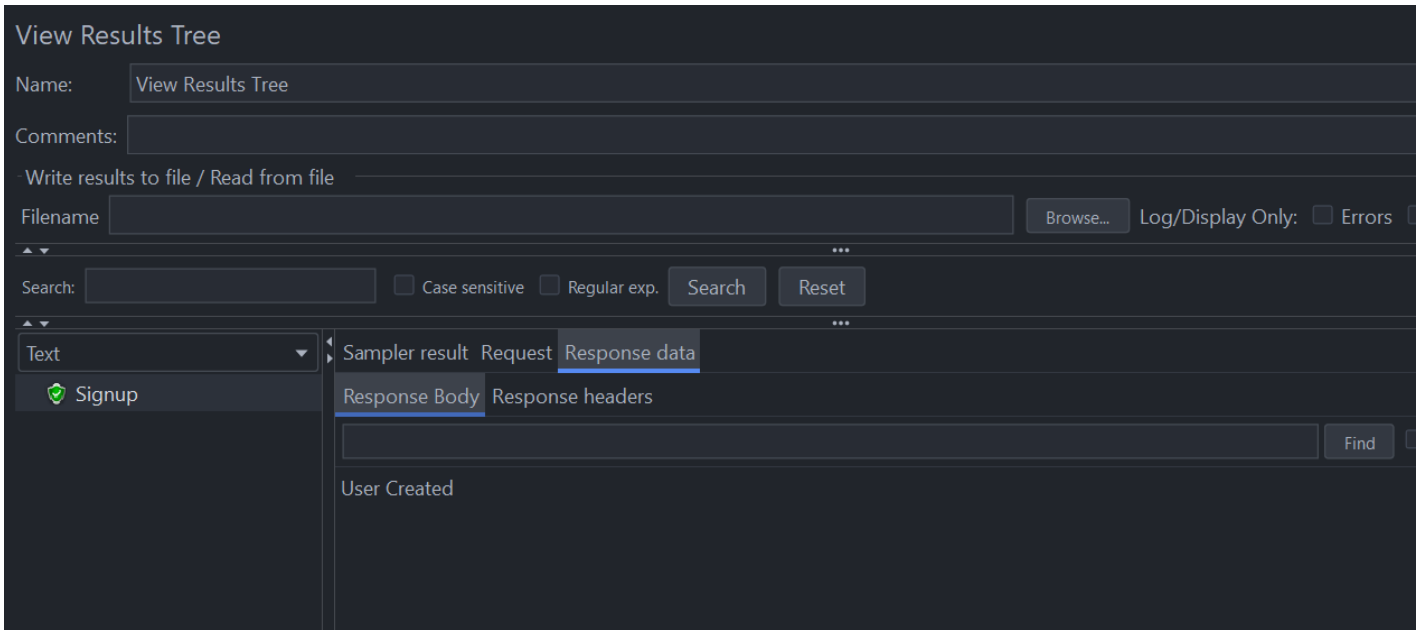

View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Displ
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Get Booking By Date and Ti
Sampler result
Request
Response data
Response Body
Response headers
null

3) Sign Up:

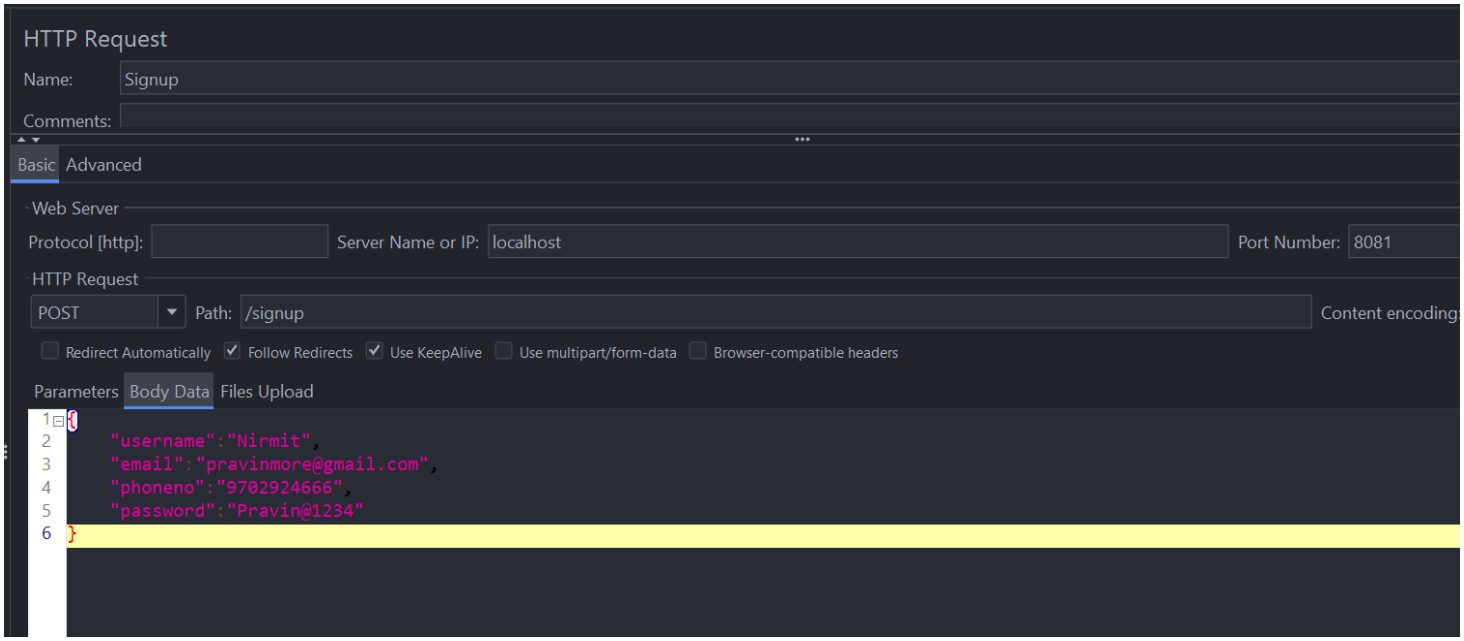
ID	Description	Data	Expected Result	Actual Result	Pass/Fail
1	Signup user	username=Pravin email=pravinmore@gmail.com password=Pravin@1234	Adds user into database	As Expected	Pass
2	Signup user	username=Nirmit email=pravinmore@gmail.com password=Pravin@1234	Username already exists	As Expected	Pass

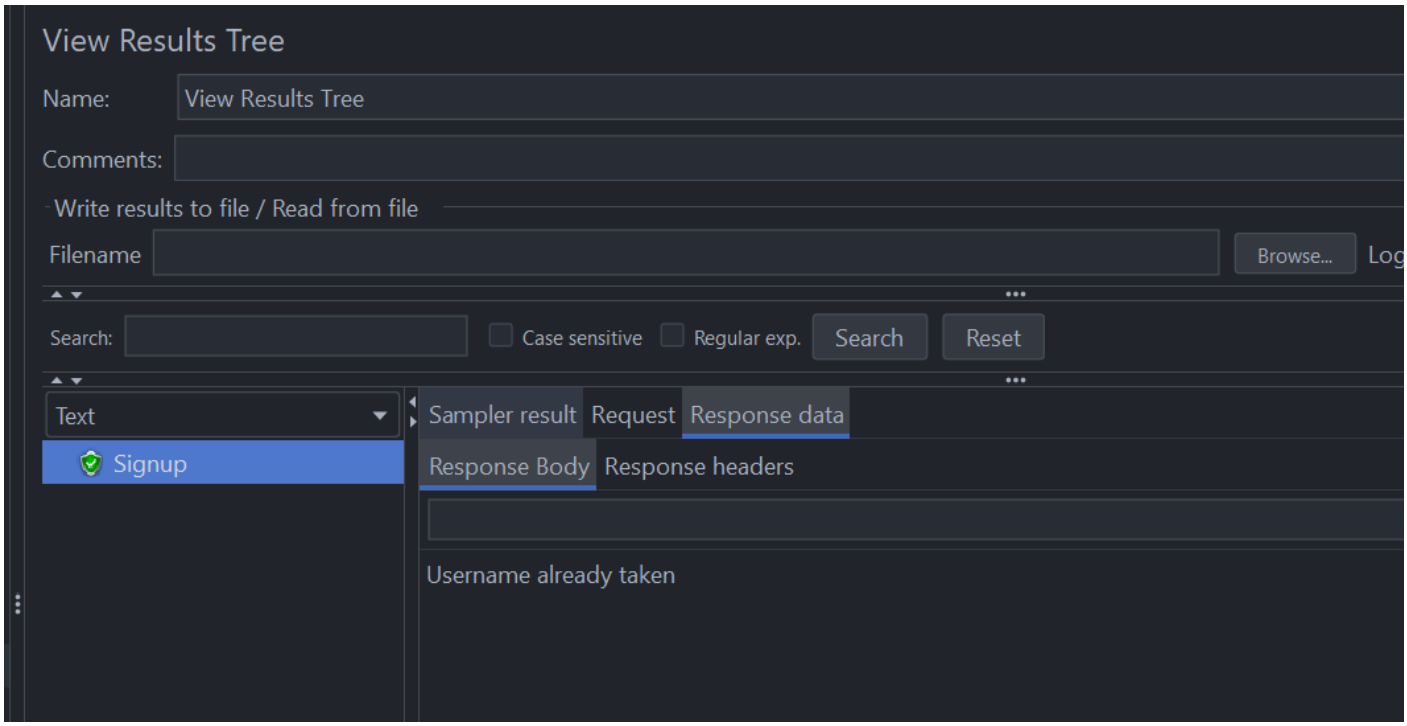
a) Test ID 1:




View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Signup
Sampler result
Request
Response data
Response Body
Response headers
Find
User Created

b) Test ID 2:

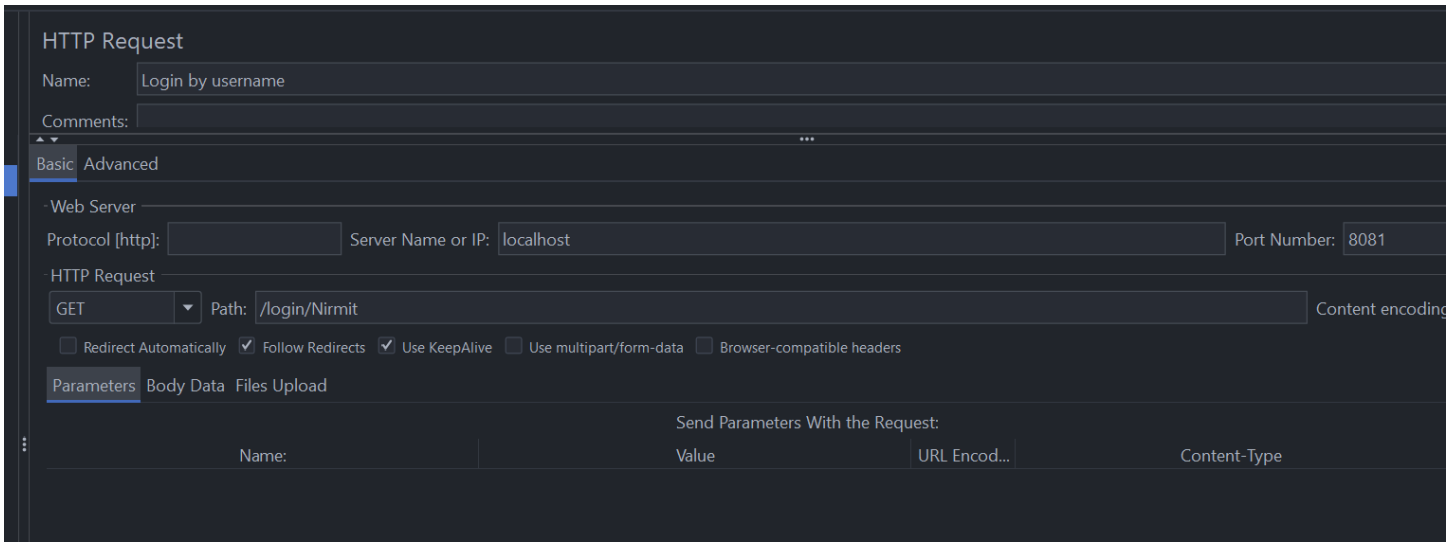

HTTP Request
Name:
Signup
Comments:
Basic
Advanced
Web Server
Protocol [http]:
Server Name or IP:
localhost
Port Number:
8081
HTTP Request
POST
Path:
/signup
Content encoding:
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters
Body Data
Files Upload
{
"username":"Nirmit",
"email":"pravinmore@gmail.com",
"phoneno":"9702924666",
"password":"Pravin@1234"
}

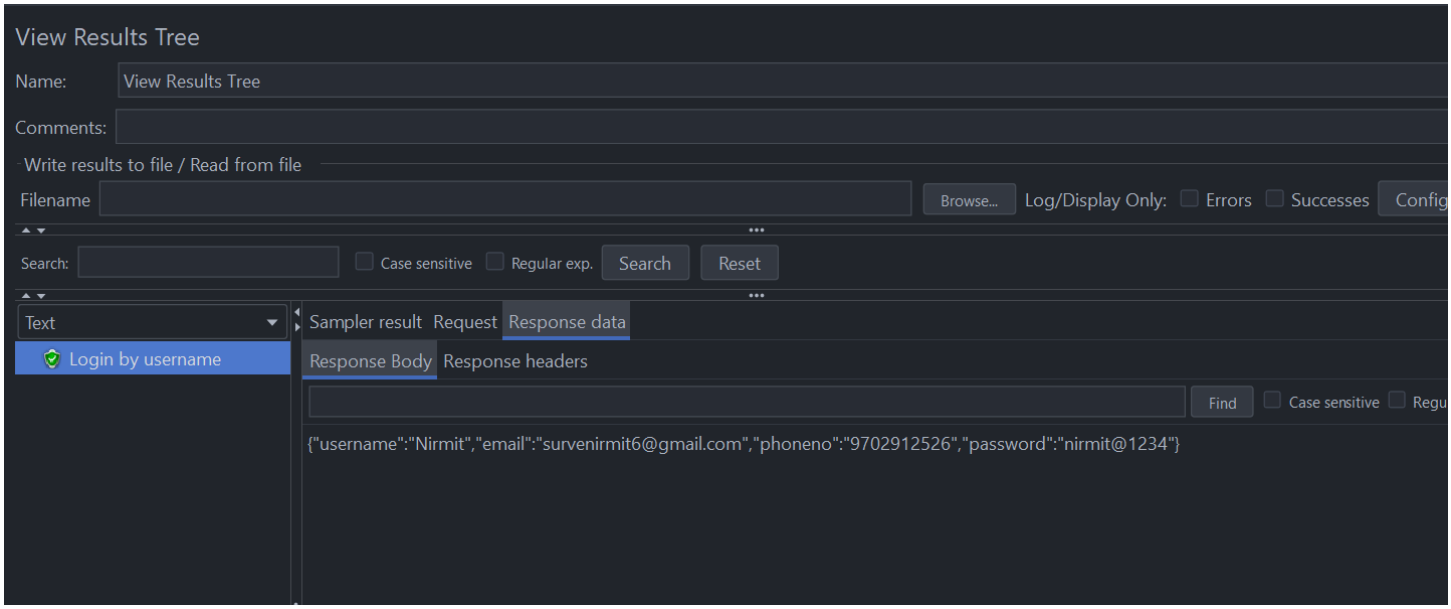


4) Login Data By Username:

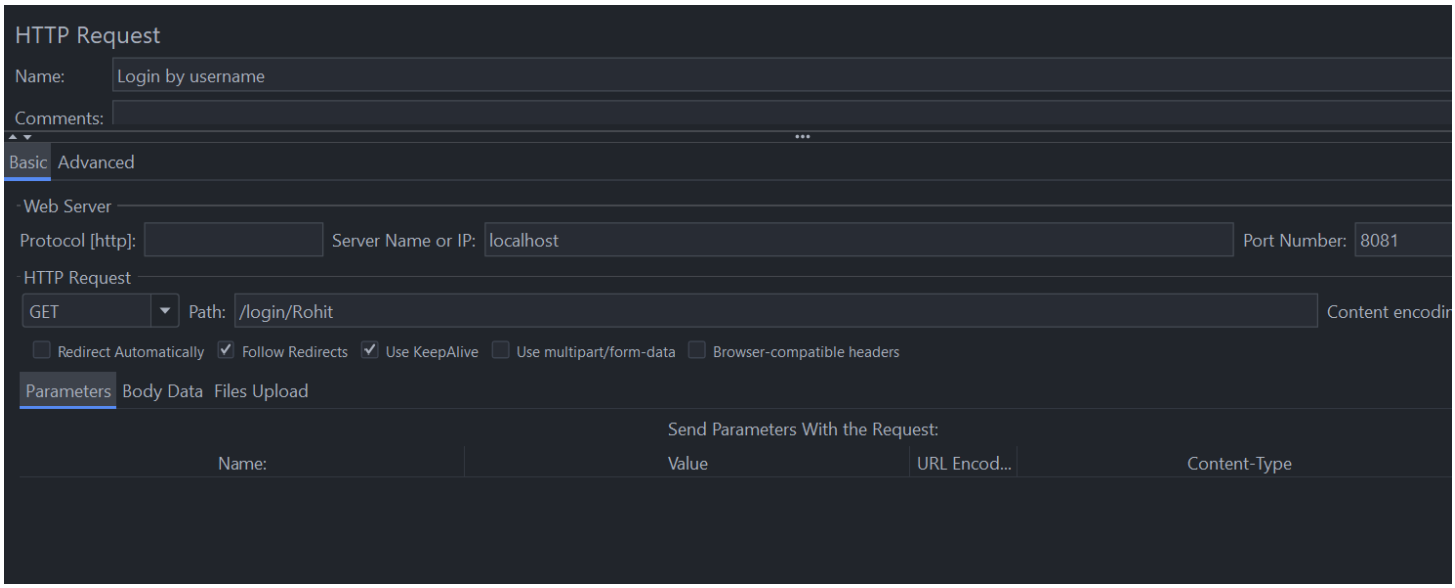
ID	Description	Data	Expected Result	Actual Result	Pass/Fail
1	Get Login Data By Username	username=Nirmit	Returns info about the account	As Expected	Pass
2	Get Login Data By Username	username=Rohit	No response	As Expected	Pass

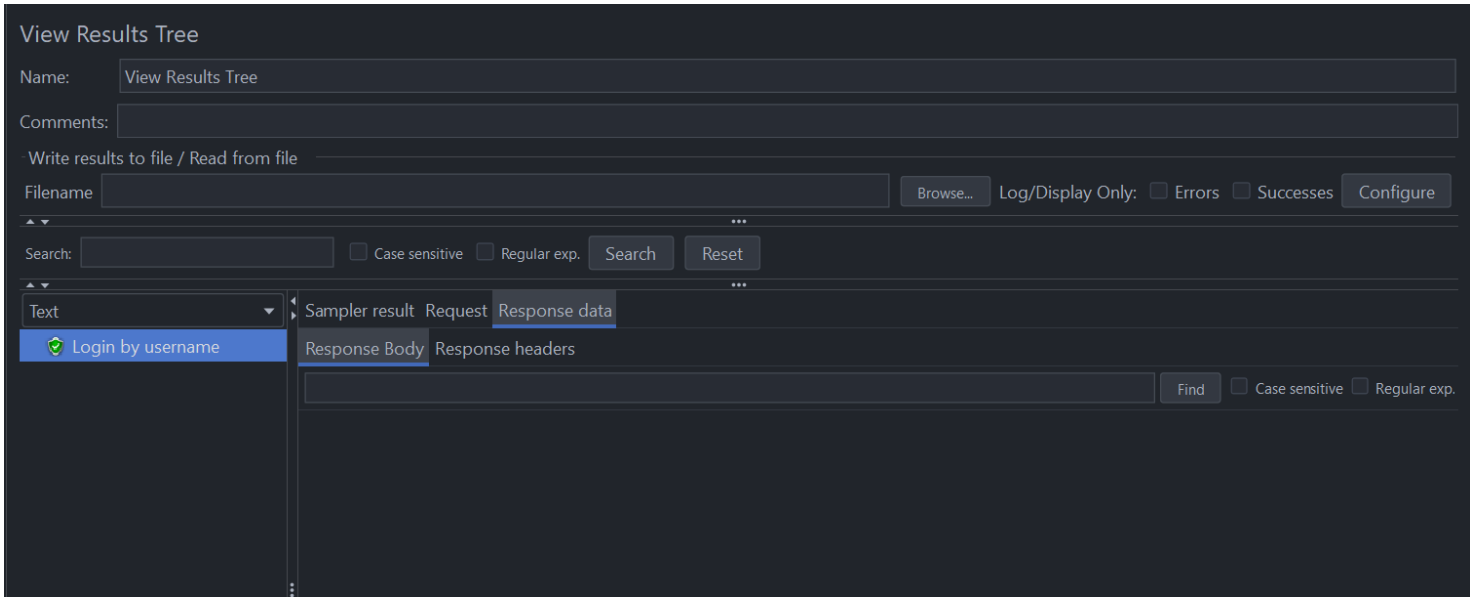
a) Test ID 1:


HTTP Request
Name:
Login by username
Comments:
Basic
Advanced
Web Server
Protocol [http]:
Server Name or IP:
localhost
Port Number:
8081
HTTP Request
GET
Path:
/login/Nirmit
Content encoding
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters
Body Data
Files Upload
Send Parameters With the Request:
Name:
Value
URL Encod...
Content-Type


View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Config
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Login by username
Sampler result
Request
Response data
Response Body
Response headers
Find
Case sensitive
{"username":"Nirmit","email":"survenirmit6@gmail.com","phoneno":"9702912526","password":"nirmit@1234"}

b) Test ID 2:

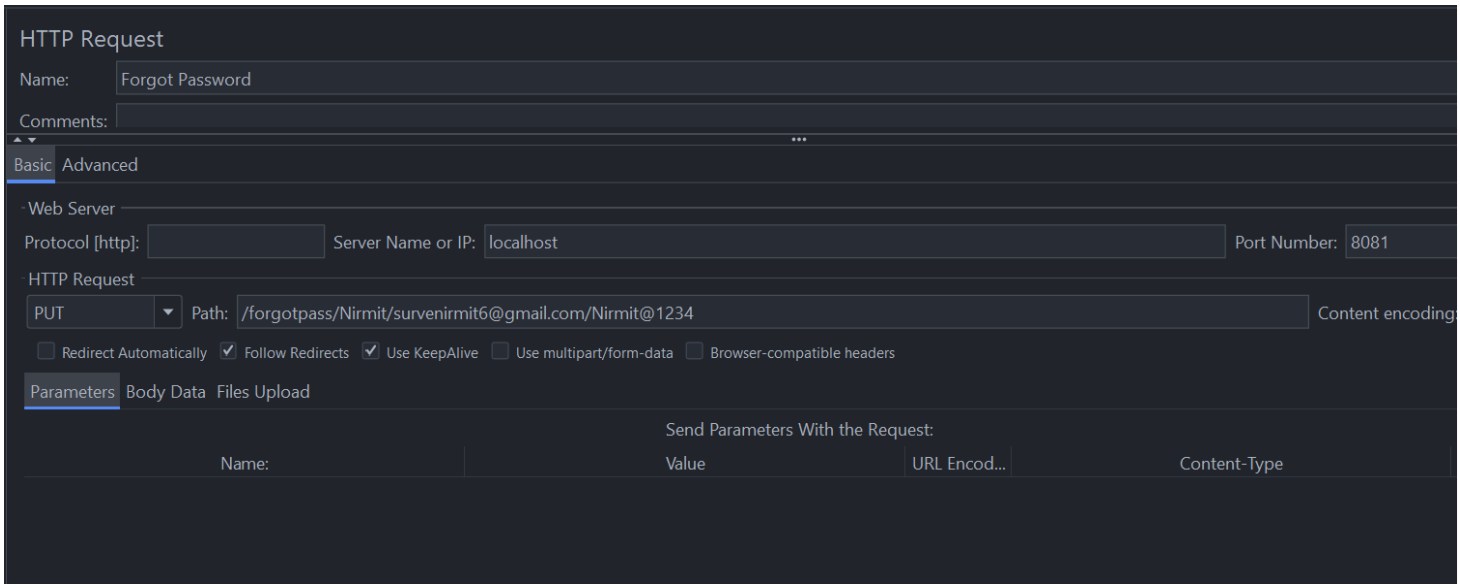

HTTP Request
Name:
Login by username
Comments:
Basic
Advanced
Web Server
Protocol [http]:
Server Name or IP:
localhost
Port Number:
8081
HTTP Request
GET
Path:
/login/Rohit
Content encodir
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters
Body Data
Files Upload
Send Parameters With the Request:
Name:
Value
URL Encod...
Content-Type

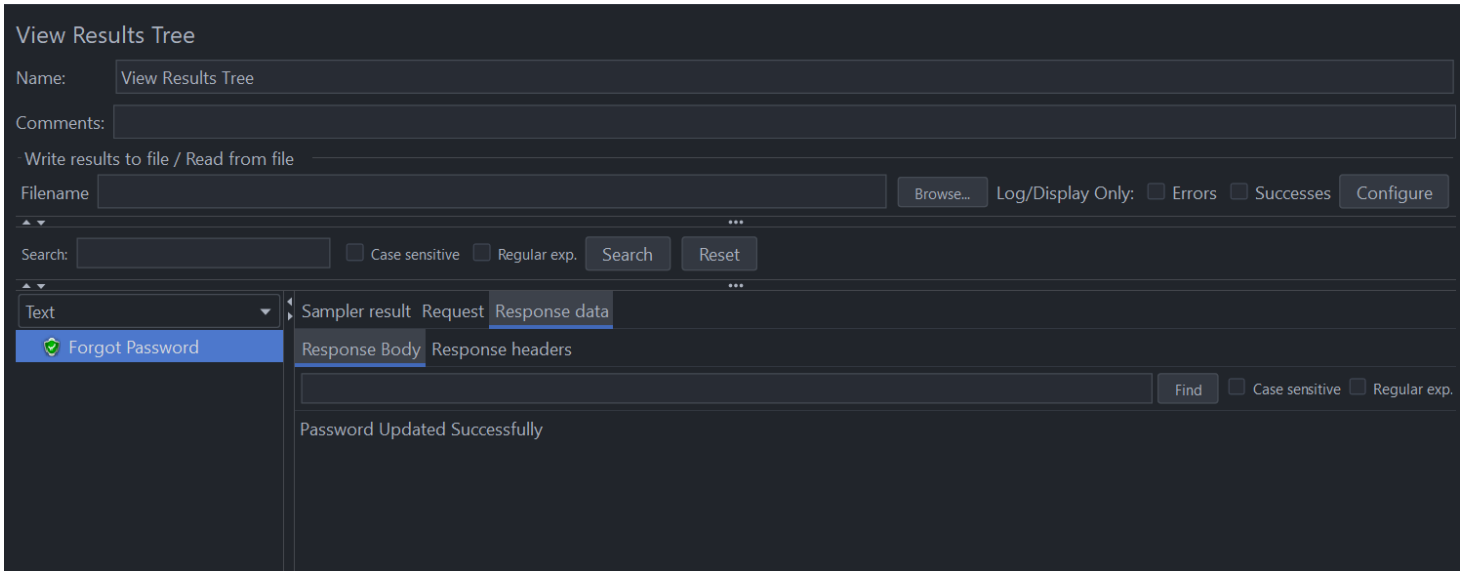

View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Login by username
Sampler result
Request
Response data
Response Body
Response headers
Find
Case sensitive
Regular exp.

5) Forgot Password

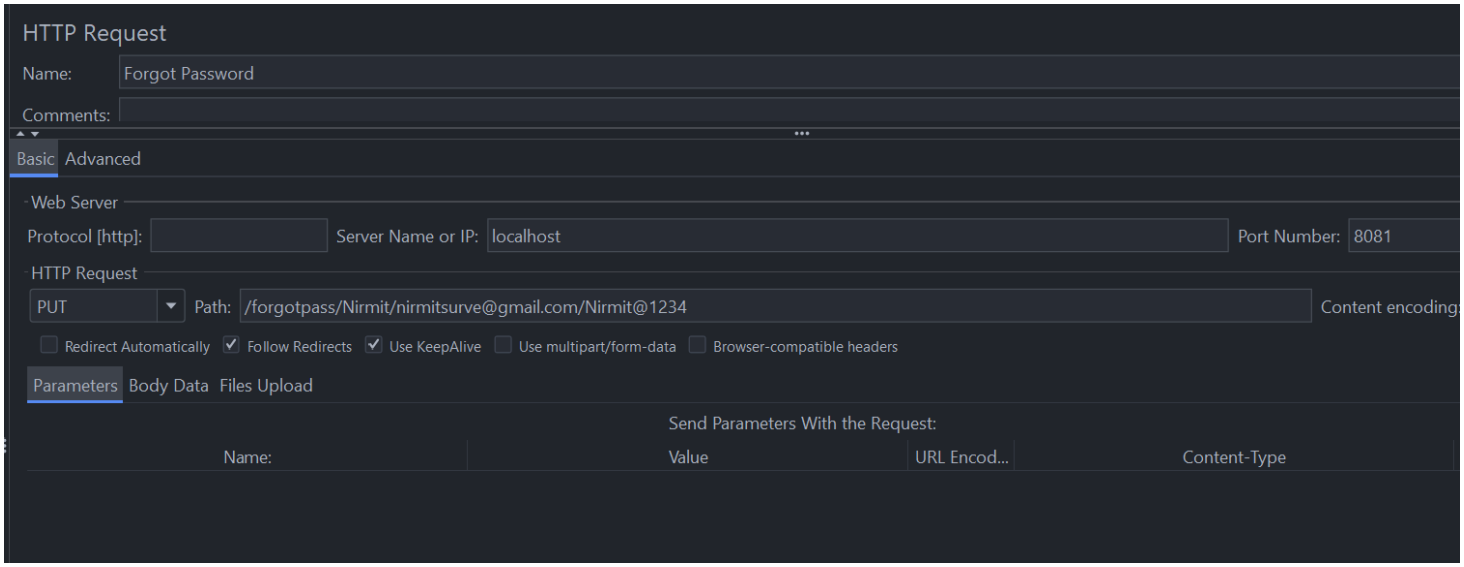
ID	Description	Data	Expected Result	Actual Result	Pass/Fail
1	Change Password	username=Nirmit email=surveniirmit6@gmail.com password=Nirmit@1234	Changes Password for the account	As Expected	Pass
2	Change Password	username=Nirmit email=nirmitsurve@gmail.com password=Nirmit@1234	Email does not match with account	As Expected	Pass
3	Change Password	username=Rohit email=nirmitsurve@gmail.com password=Nirmit@1234	Account Not Found	As Expected	Pass

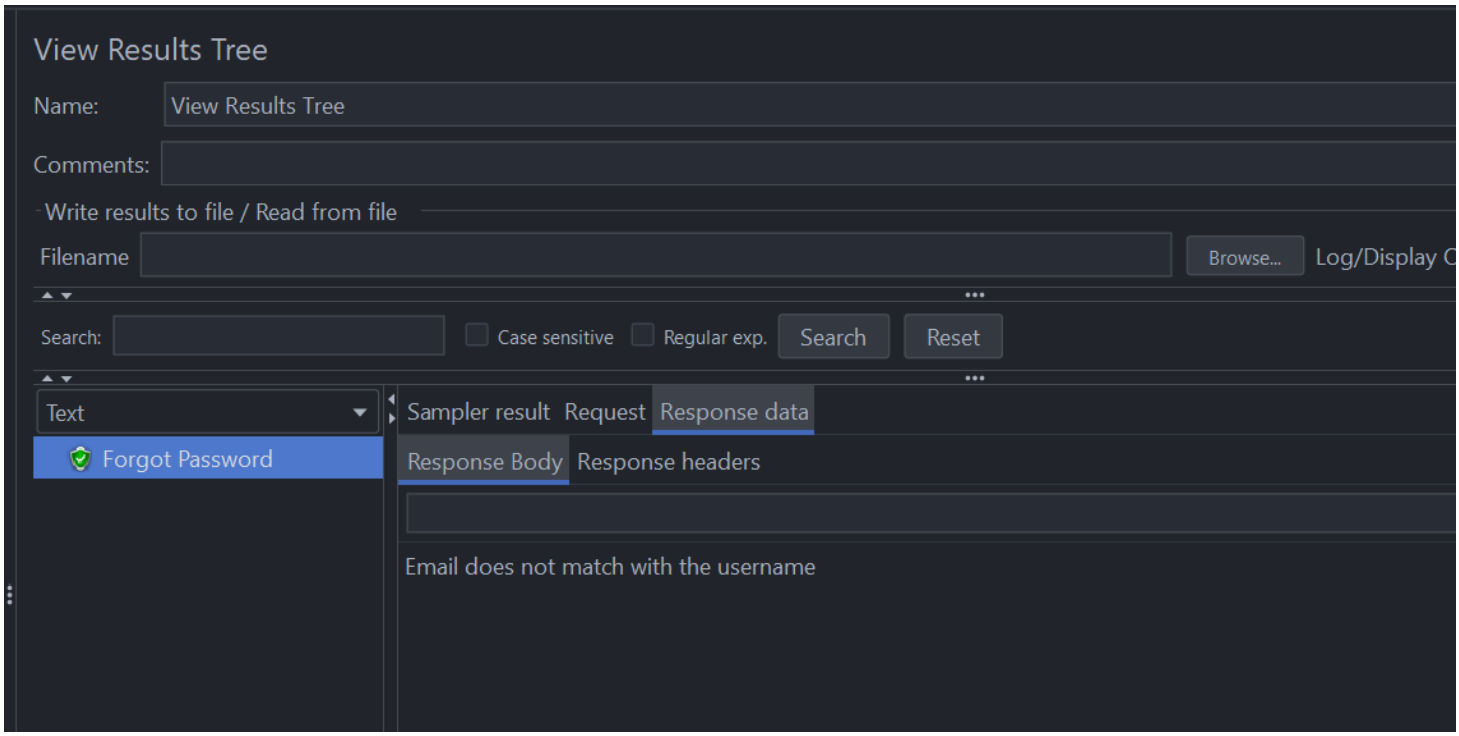
a) Test ID 1:


HTTP Request
Name: Forgot Password
Comments:
Basic Advanced
Web Server
Protocol [http]:
Server Name or IP: localhost
Port Number: 8081
HTTP Request
PUT
Path: /forgotpass/Nirmit/survenirmit6@gmail.com/Nirmit@1234
Content encoding
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters Body Data Files Upload
Send Parameters With the Request:
Name:
Value
URL Encod...
Content-Type

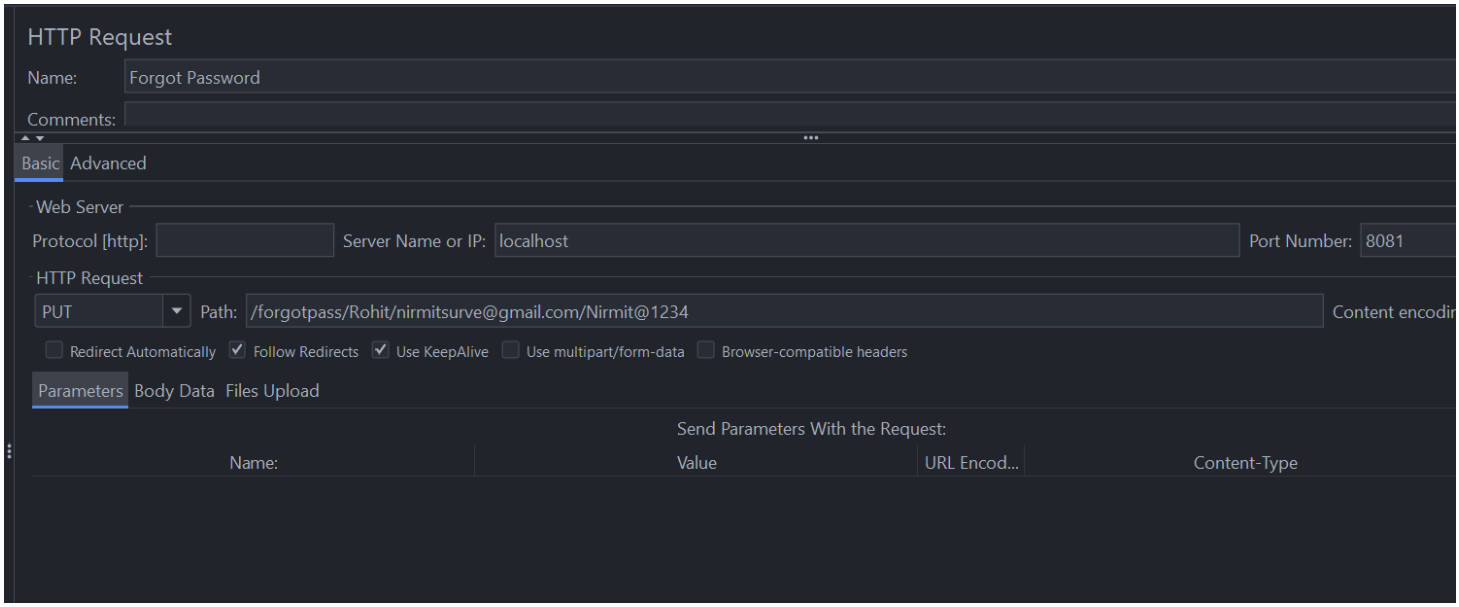

View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Forgot Password
Sampler result
Request
Response data
Response Body
Response headers
Find
Case sensitive
Regular exp.
Password Updated Successfully

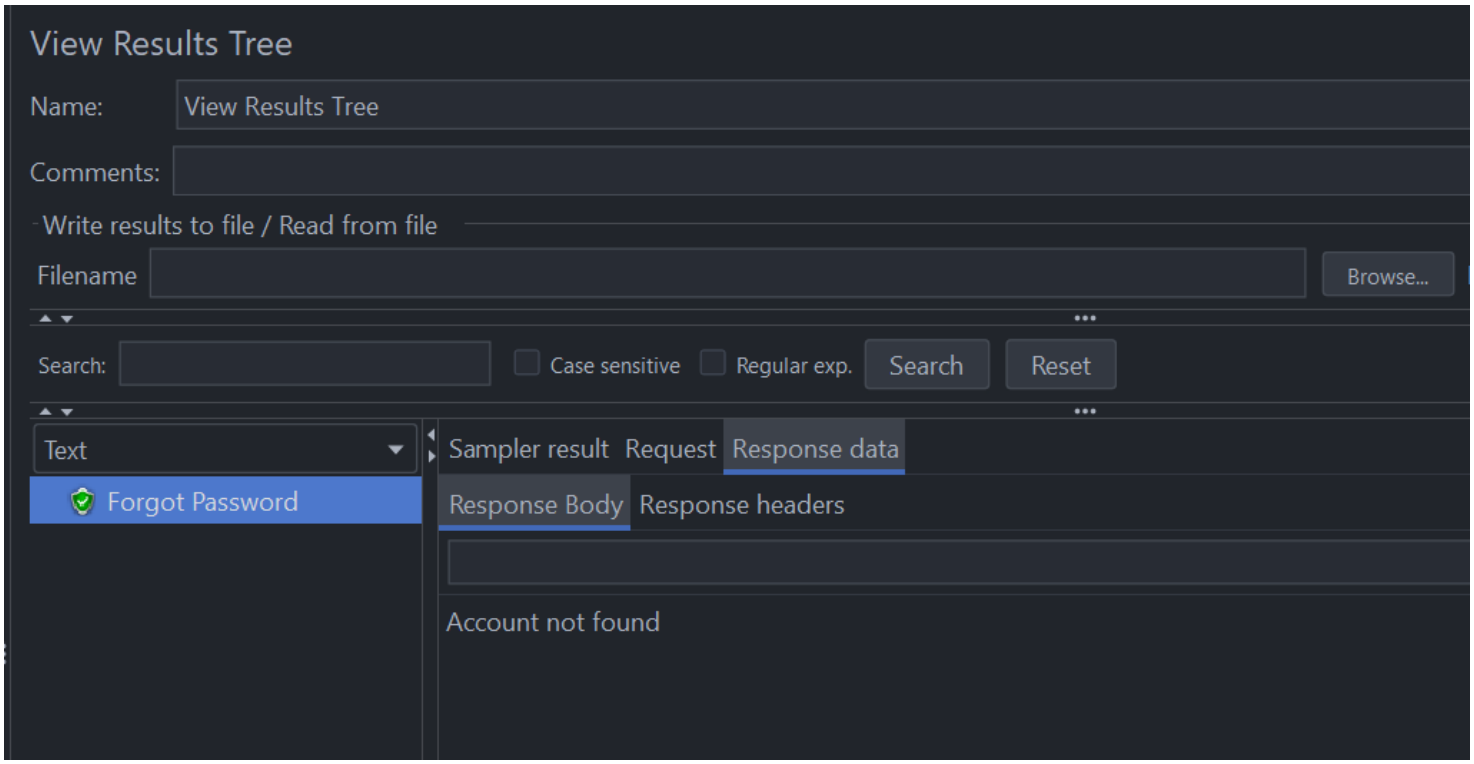
b) Test ID 2:


HTTP Request
Name: Forgot Password
Comments:
Basic Advanced
Web Server
Protocol [http]:
Server Name or IP: localhost
Port Number: 8081
HTTP Request
PUT
Path: /forgotpass/Nirmit/nirmitsurve@gmail.com/Nirmit@1234
Content encoding:
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters Body Data Files Upload
Send Parameters With the Request:
Name:
Value
URL Encod...
Content-Type


View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display C
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Forgot Password
Sampler result
Request
Response data
Response Body
Response headers
Email does not match with the username

c) Test ID 3:

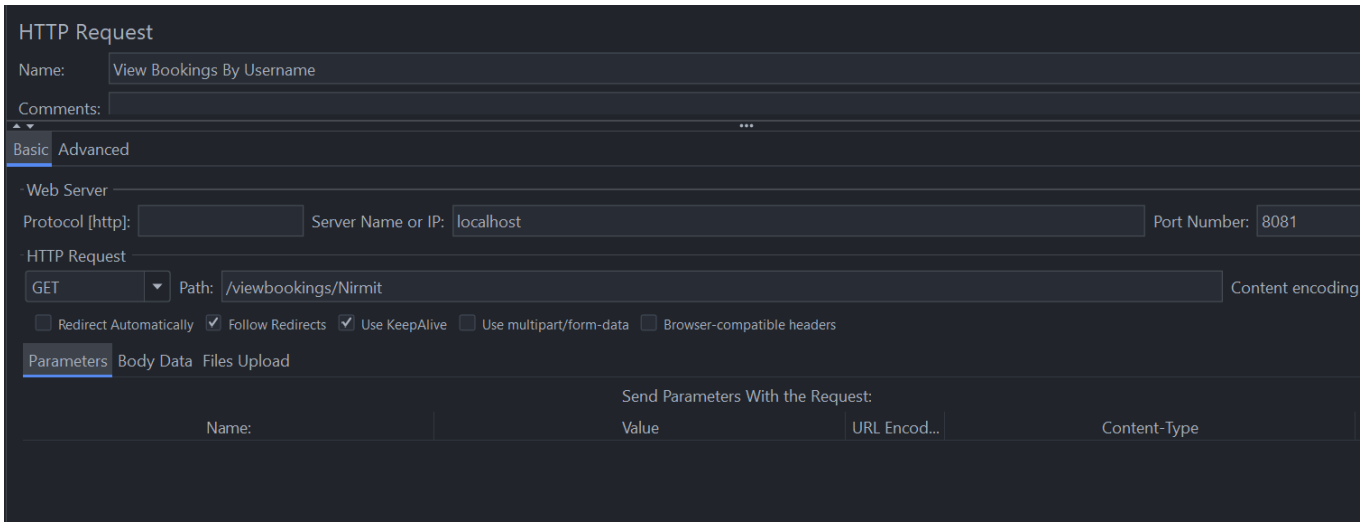

HTTP Request
Name: Forgot Password
Comments:
Basic Advanced
Web Server
Protocol [http]:
Server Name or IP: localhost
Port Number: 8081
HTTP Request
PUT
Path: /forgotpass/Rohit/nirmitsurve@gmail.com/Nirmit@1234
Content encodir
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters Body Data Files Upload
Send Parameters With the Request:
Name:
Value
URL Encod...
Content-Type

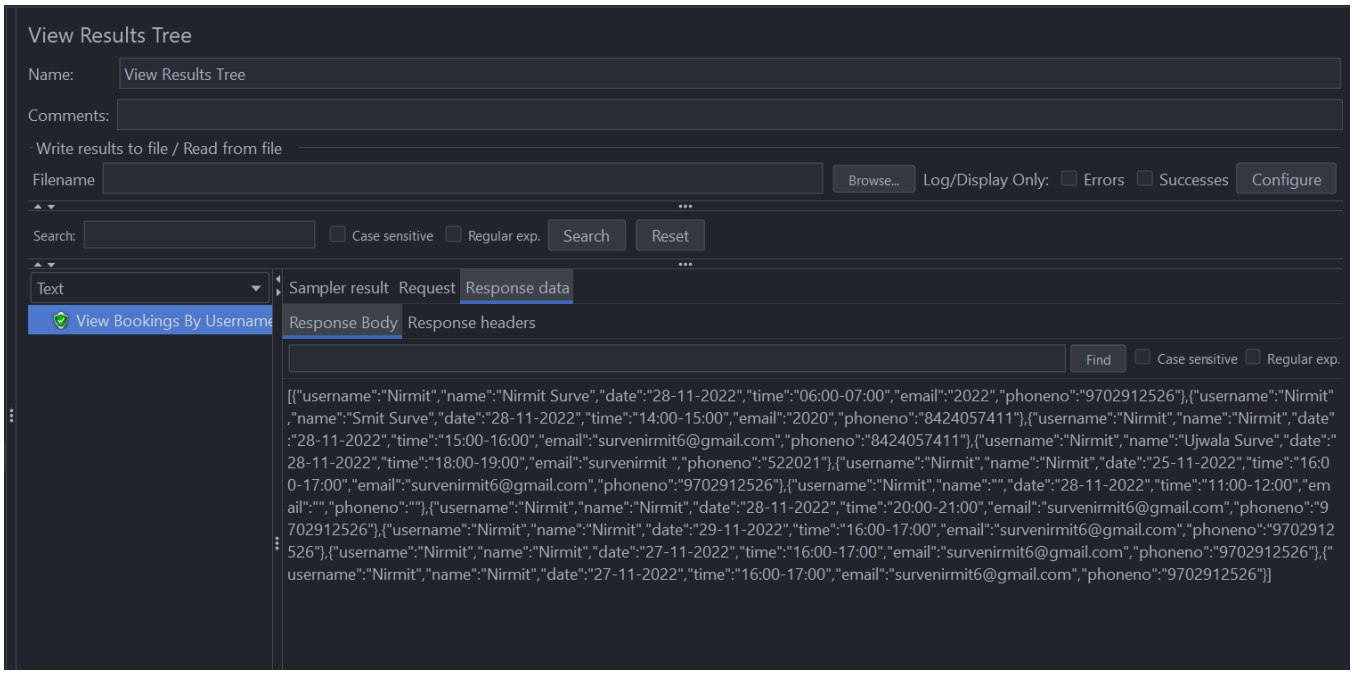

View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Forgot Password
Sampler result
Request
Response data
Response Body
Response headers
Account not found

6) View Bookings by username:

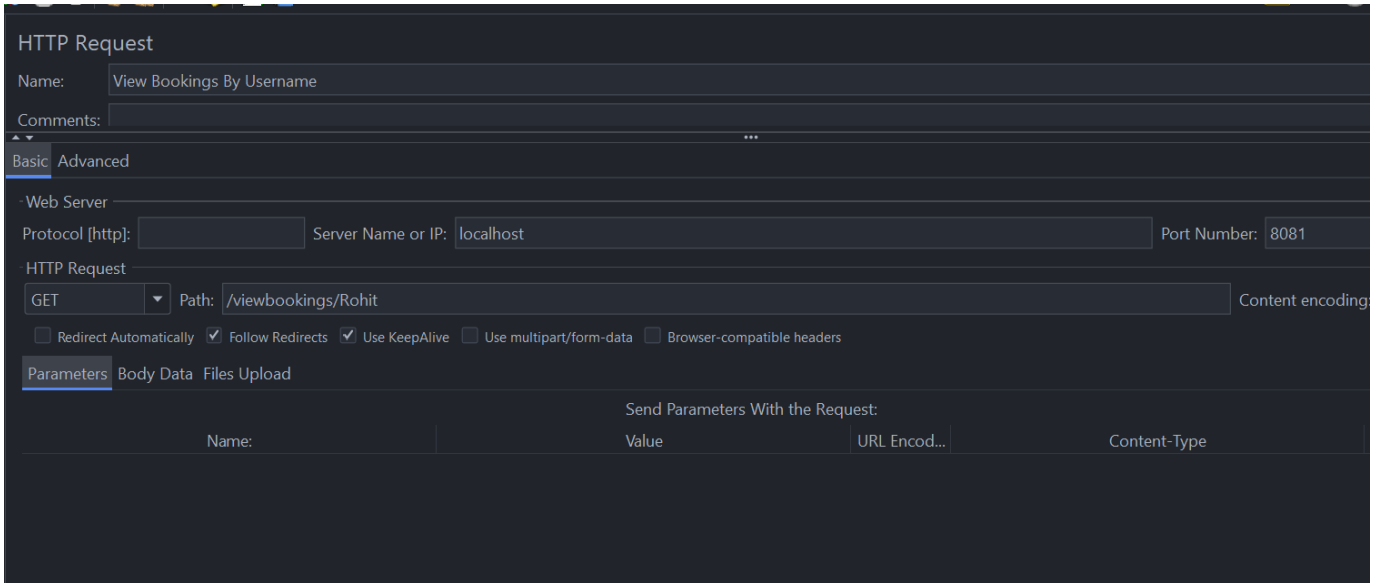
ID	Description	Data	Expected Result	Actual Result	Pass/Fail
1	Get booking data by username	username=Nirmit	Returns info about the bookings done by user	As Expected	Pass
2	Get booking data by username	username=Rohit	Empty List	As Expected	Pass

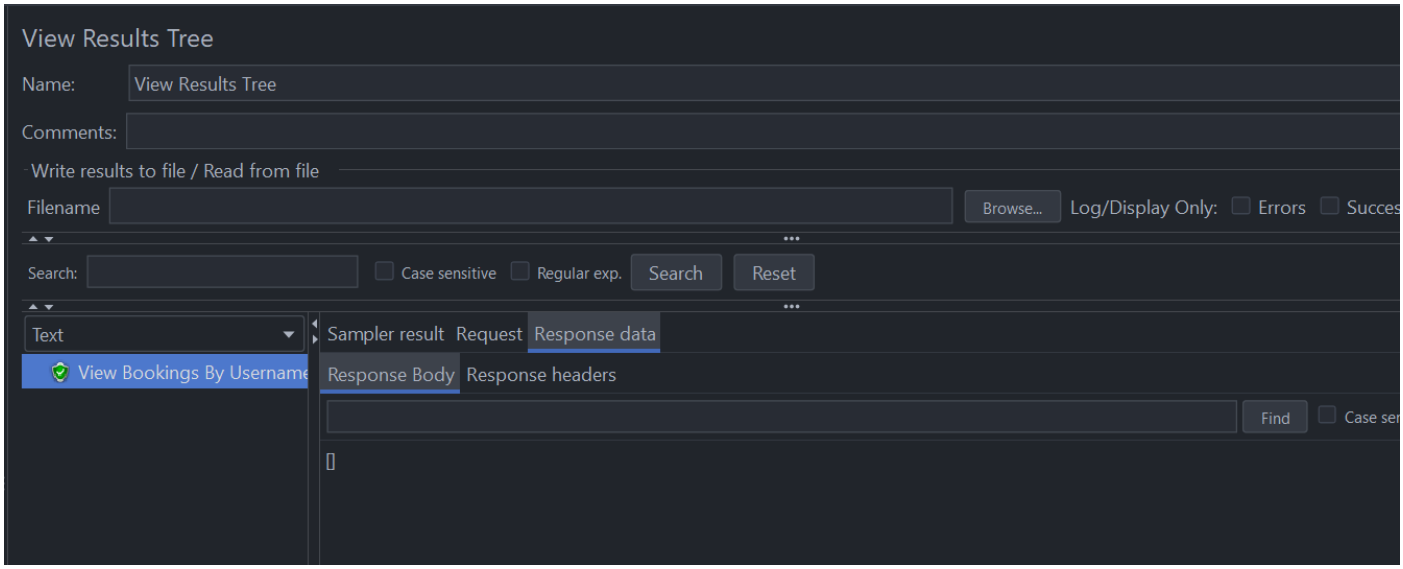
a) Test ID 1:




View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Search:
Case sensitive
Regular exp.
Search
Reset
Text
View Bookings By Username
Sampler result
Request
Response data
Response Body
Response headers
Find
Case sensitive
Regular exp.
[{"username":"Nirmit","name":"Nirmit Surve","date":"28-11-2022","time":"06:00-07:00","email":"2022","phoneno":"9702912526"},{"username":"Nirmit","name":"Smit Surve","date":"28-11-2022","time":"14:00-15:00","email":"2020","phoneno":"8424057411"},{"username":"Nirmit","name":"Nirmit","date":"28-11-2022","time":"15:00-16:00","email":"survenirmit6@gmail.com","phoneno":"8424057411"},{"username":"Nirmit","name":"Ujwala Surve","date":"28-11-2022","time":"18:00-19:00","email":"survenirmit ","phoneno":"522021"},{"username":"Nirmit","name":"Nirmit","date":"25-11-2022","time":"16:00-17:00","email":"survenirmit6@gmail.com","phoneno":"9702912526"},{"username":"Nirmit","name":"","date":"28-11-2022","time":"11:00-12:00","email":"","phoneno":""},{"username":"Nirmit","name":"Nirmit","date":"28-11-2022","time":"20:00-21:00","email":"survenirmit6@gmail.com","phoneno":"9702912526"},{"username":"Nirmit","name":"Nirmit","date":"29-11-2022","time":"16:00-17:00","email":"survenirmit6@gmail.com","phoneno":"9702912526"},{"username":"Nirmit","name":"Nirmit","date":"27-11-2022","time":"16:00-17:00","email":"survenirmit6@gmail.com","phoneno":"9702912526"},{"username":"Nirmit","name":"Nirmit","date":"27-11-2022","time":"16:00-17:00","email":"survenirmit6@gmail.com","phoneno":"9702912526"}]

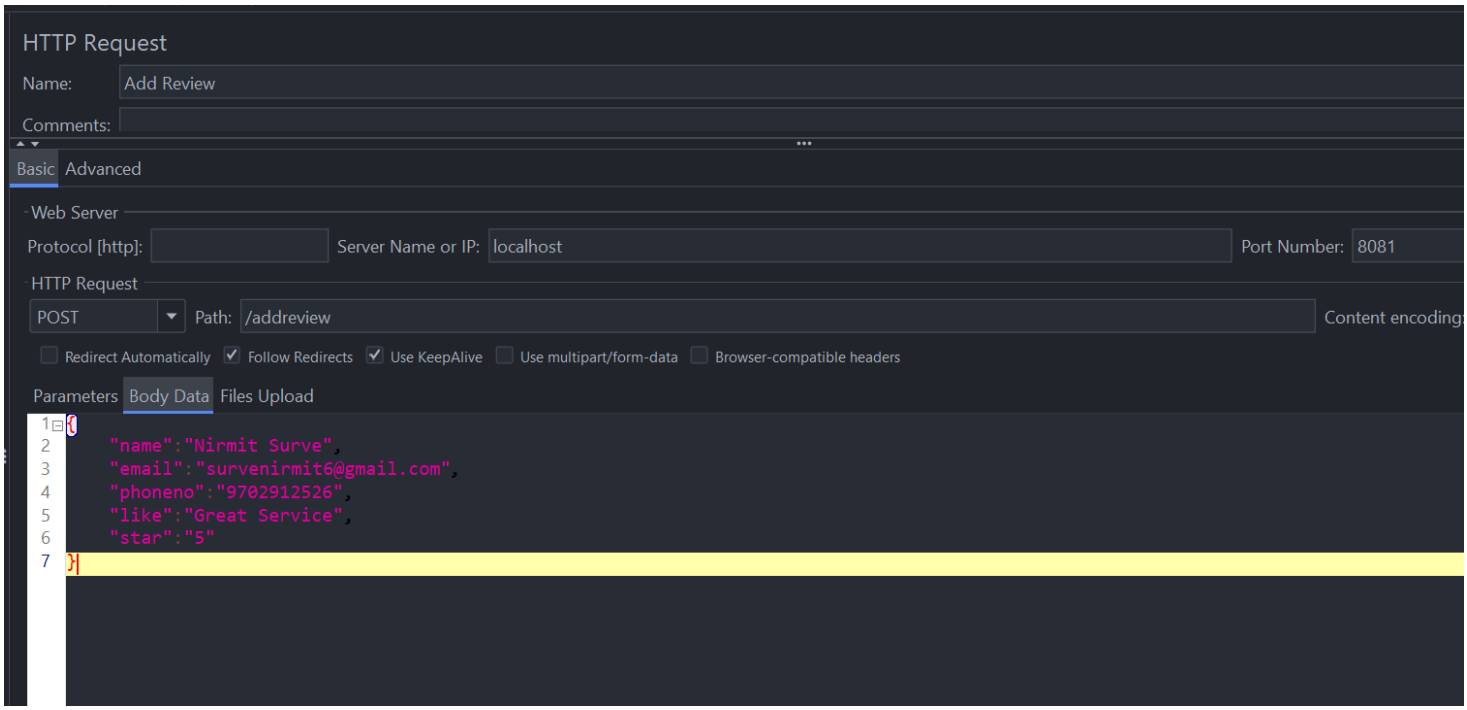
b) Test ID 2:


HTTP Request
Name:
View Bookings By Username
Comments:
Basic
Advanced
Web Server
Protocol [http]:
Server Name or IP:
localhost
Port Number:
8081
HTTP Request
GET
Path:
/viewbookings/Rohit
Content encoding:
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters
Body Data
Files Upload
Send Parameters With the Request:
Name:
Value
URL Encod...
Content-Type

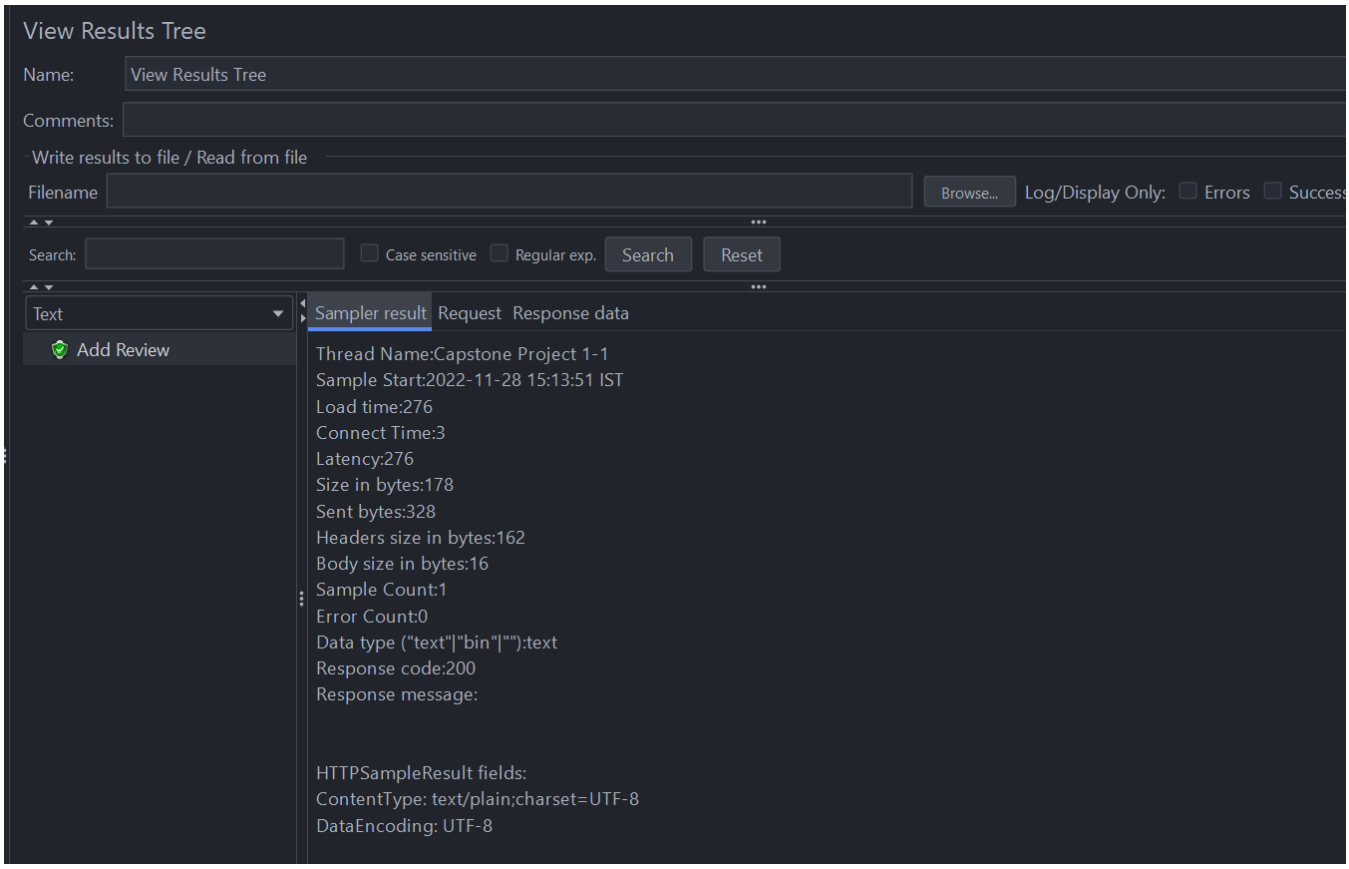

View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Succes
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Sampler result
Request
Response data
View Bookings By Username
Response Body
Response headers
Find
Case ser

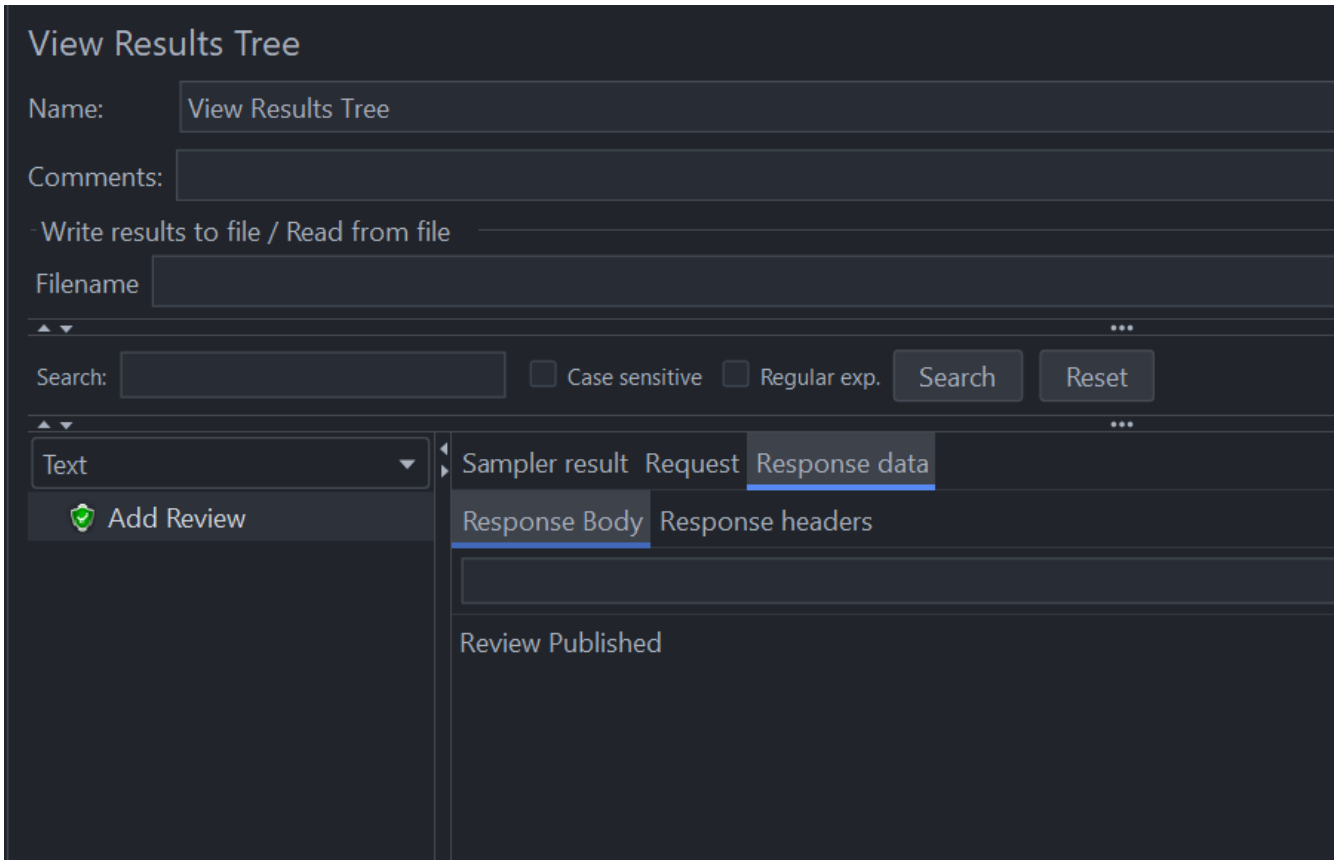
7) Add review:

This method has no fail case.



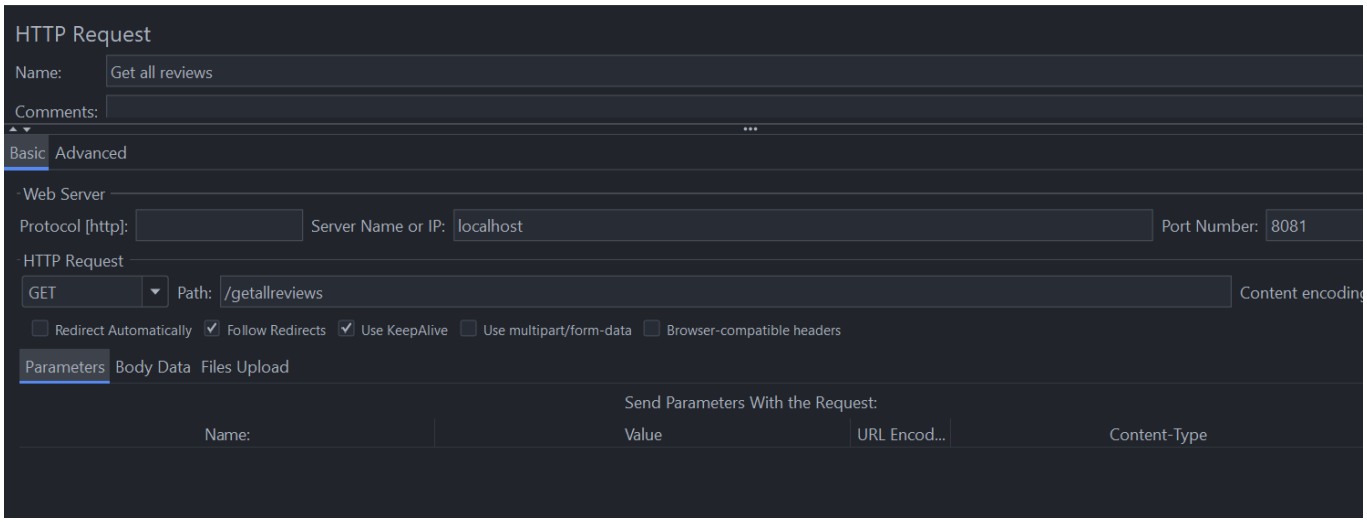
When we run the request we can see the result in the results tree


View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Success
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Add Review
Sampler result
Request
Response data
Thread Name:Capstone Project 1-1
Sample Start:2022-11-28 15:13:51 IST
Load time:276
Connect Time:3
Latency:276
Size in bytes:178
Sent bytes:328
Headers size in bytes:162
Body size in bytes:16
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:
HTTPSampleResult fields:
ContentType: text/plain;charset=UTF-8
DataEncoding: UTF-8

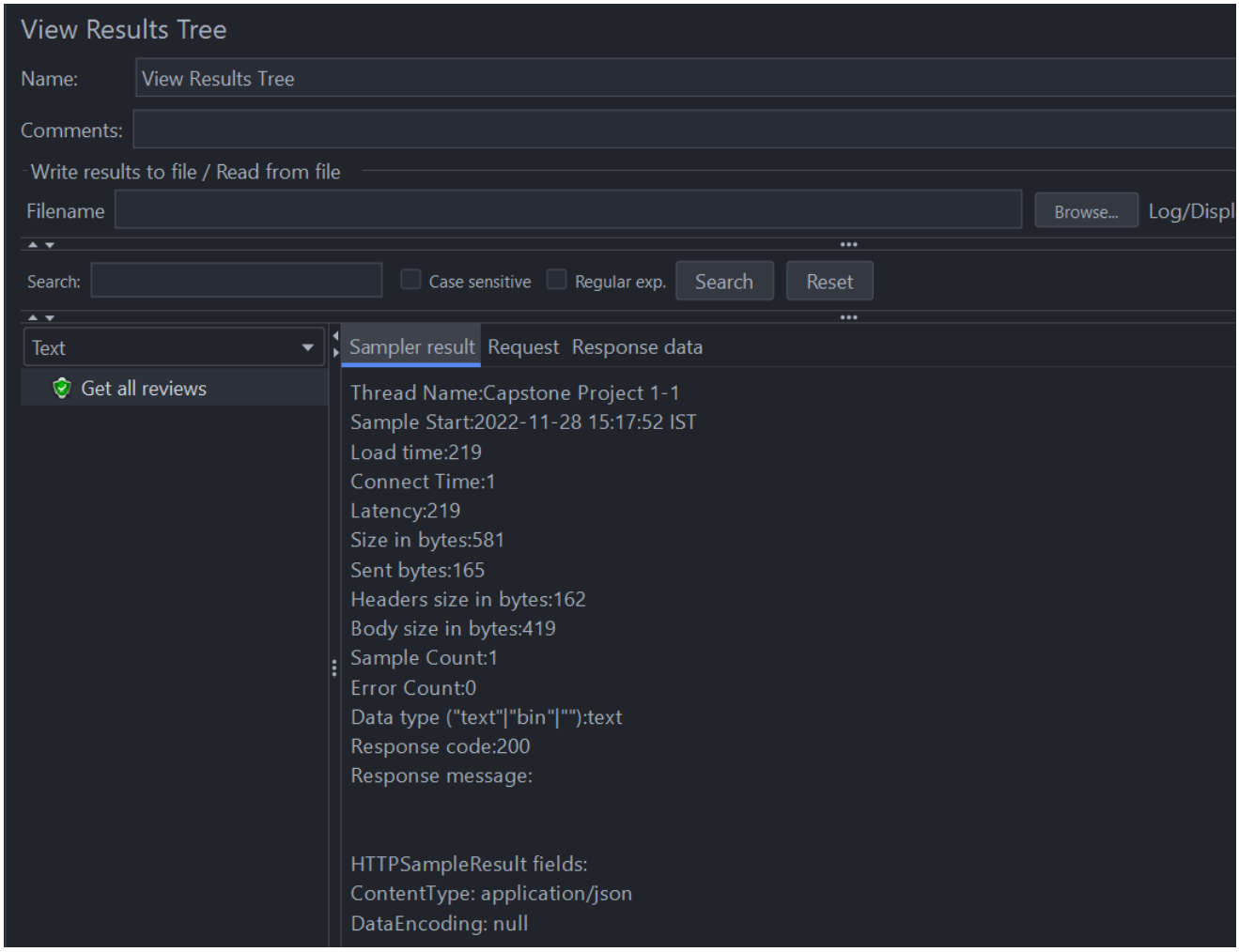


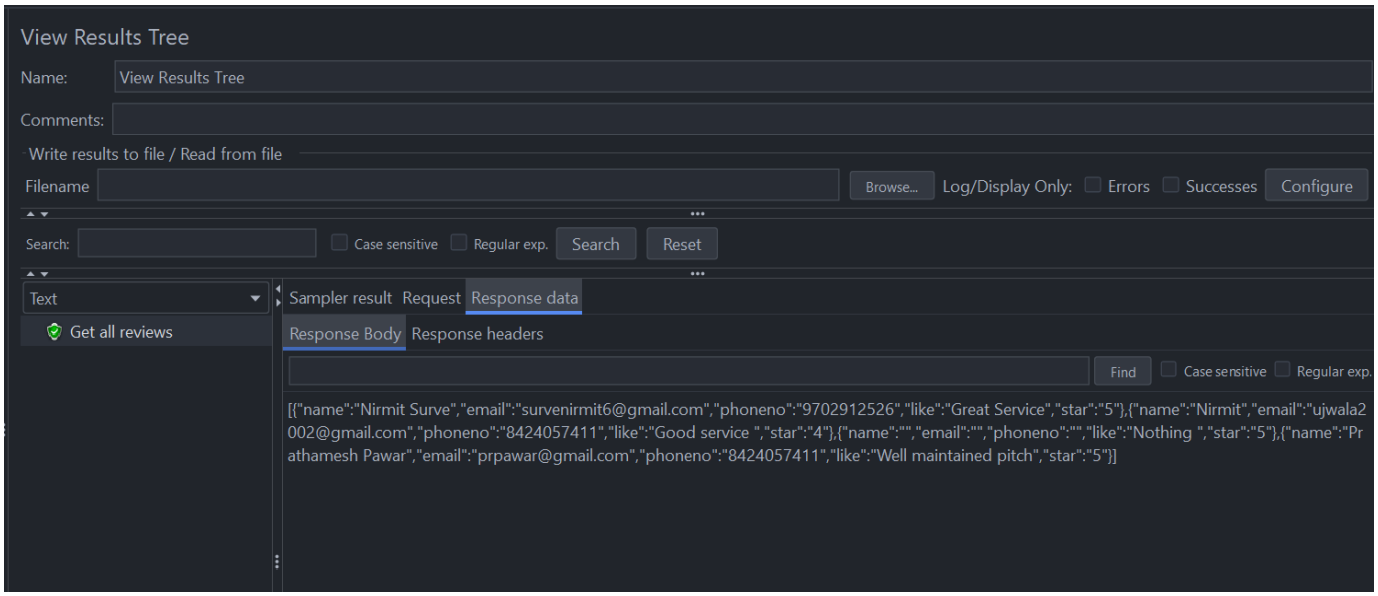
8) Get all reviews:

This method has no fail case



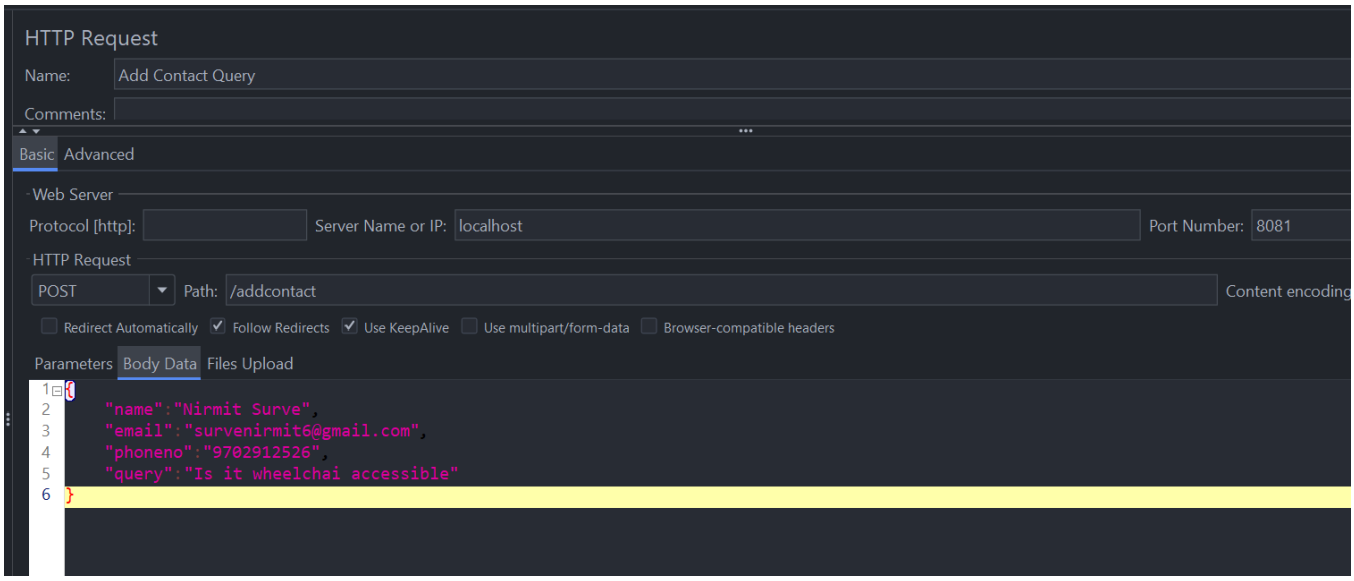
When we run this request we can see the results in the result trees



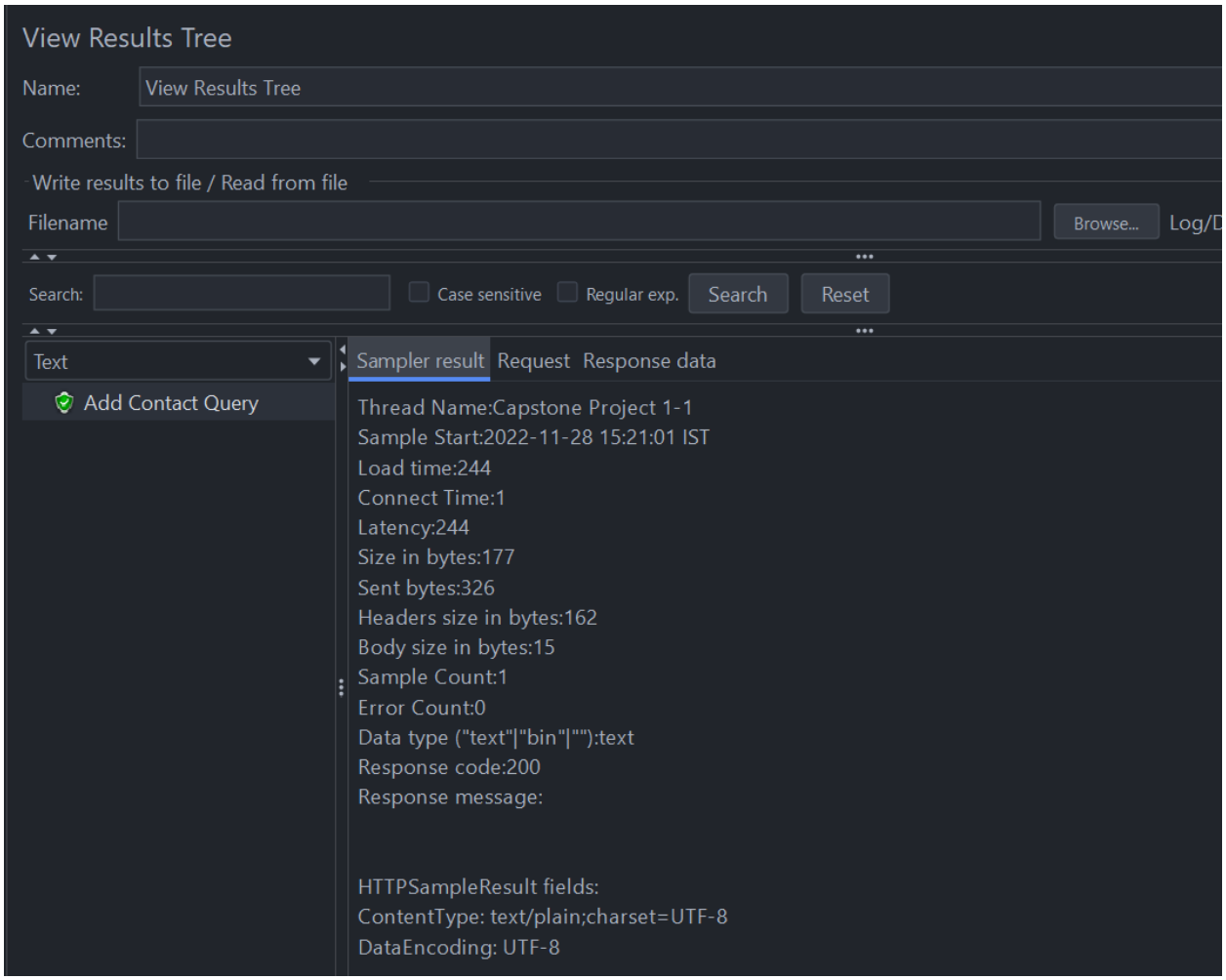


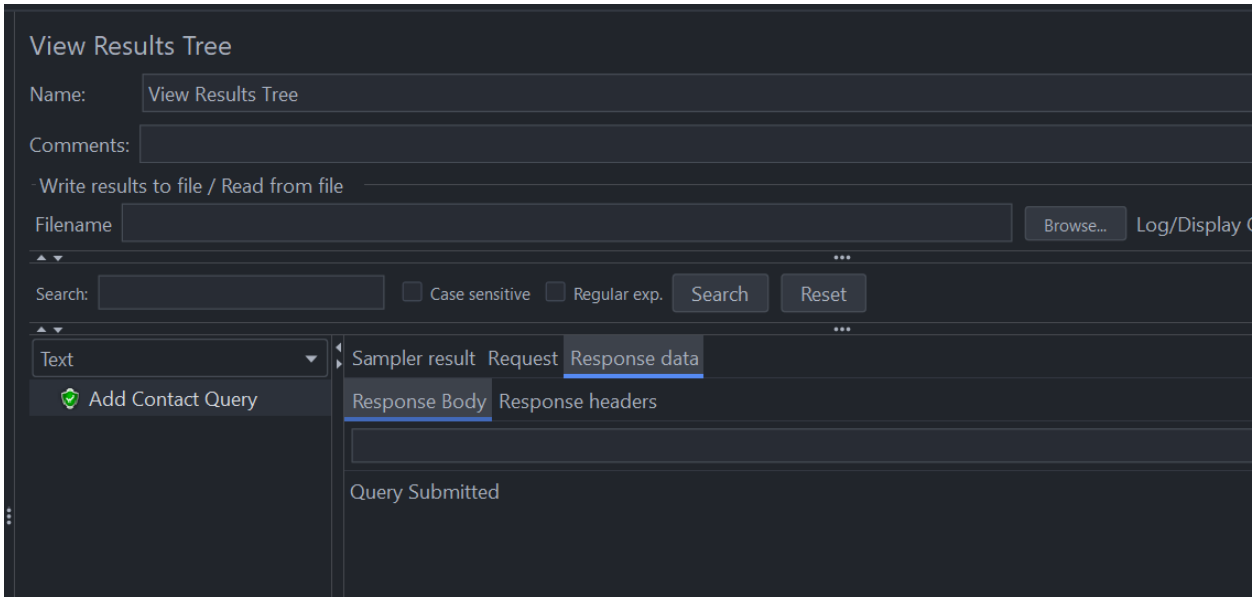
9) Add Contact Query

This method has no fail case



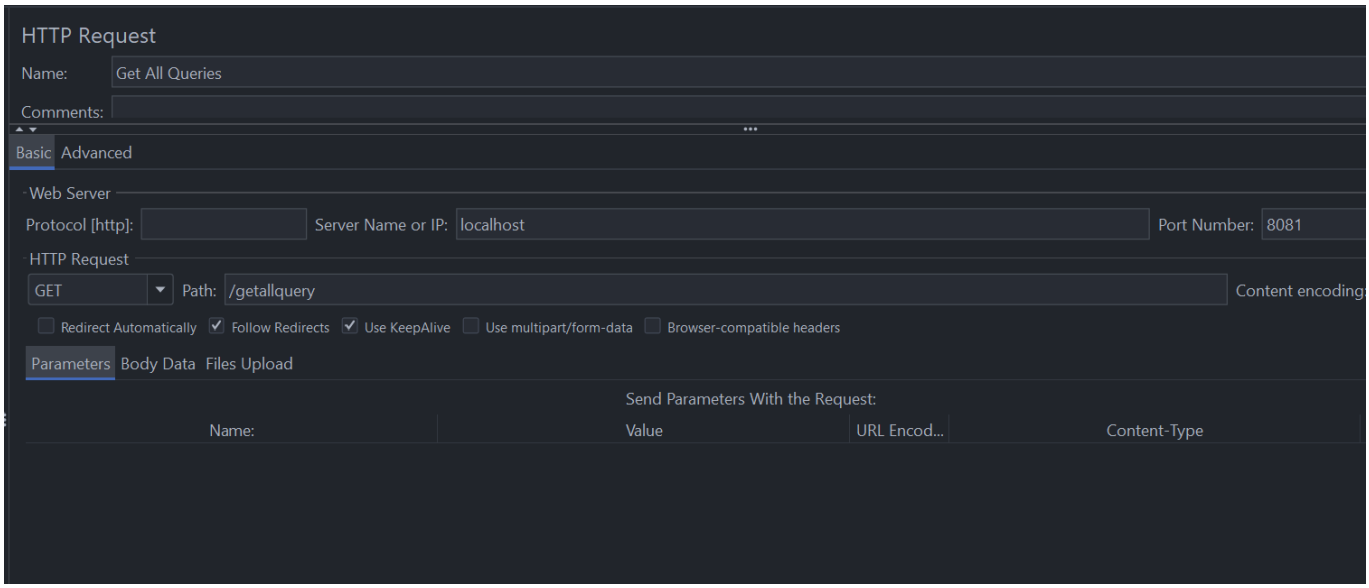
When we run this request we can see the results in the result trees


View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/D
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Add Contact Query
Sampler result
Request
Response data
Thread Name:Capstone Project 1-1
Sample Start:2022-11-28 15:21:01 IST
Load time:244
Connect Time:1
Latency:244
Size in bytes:177
Sent bytes:326
Headers size in bytes:162
Body size in bytes:15
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:
HTTPSampleResult fields:
ContentType: text/plain;charset=UTF-8
DataEncoding: UTF-8

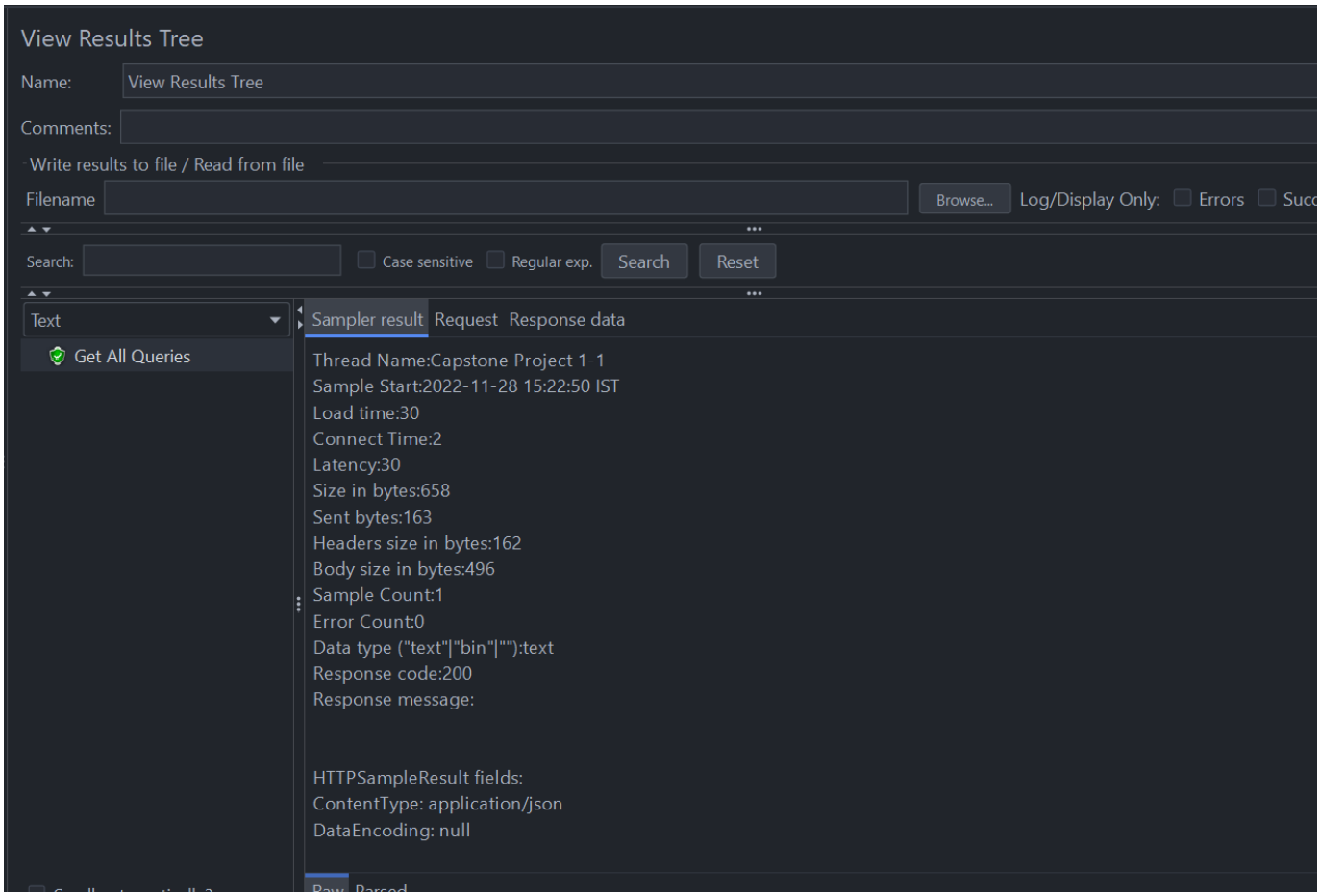

View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Add Contact Query
Sampler result
Request
Response data
Response Body
Response headers
Query Submitted

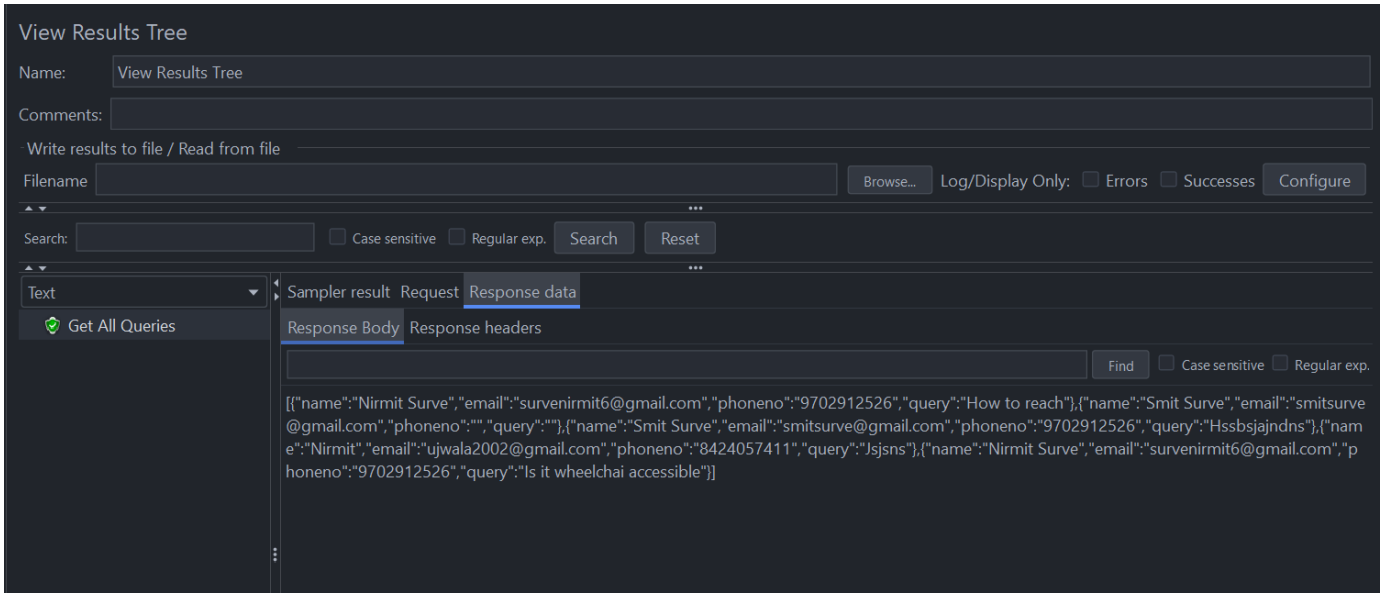
10) Get all User Queries

This method has no fail case



When we run this request we can see the results in the result trees

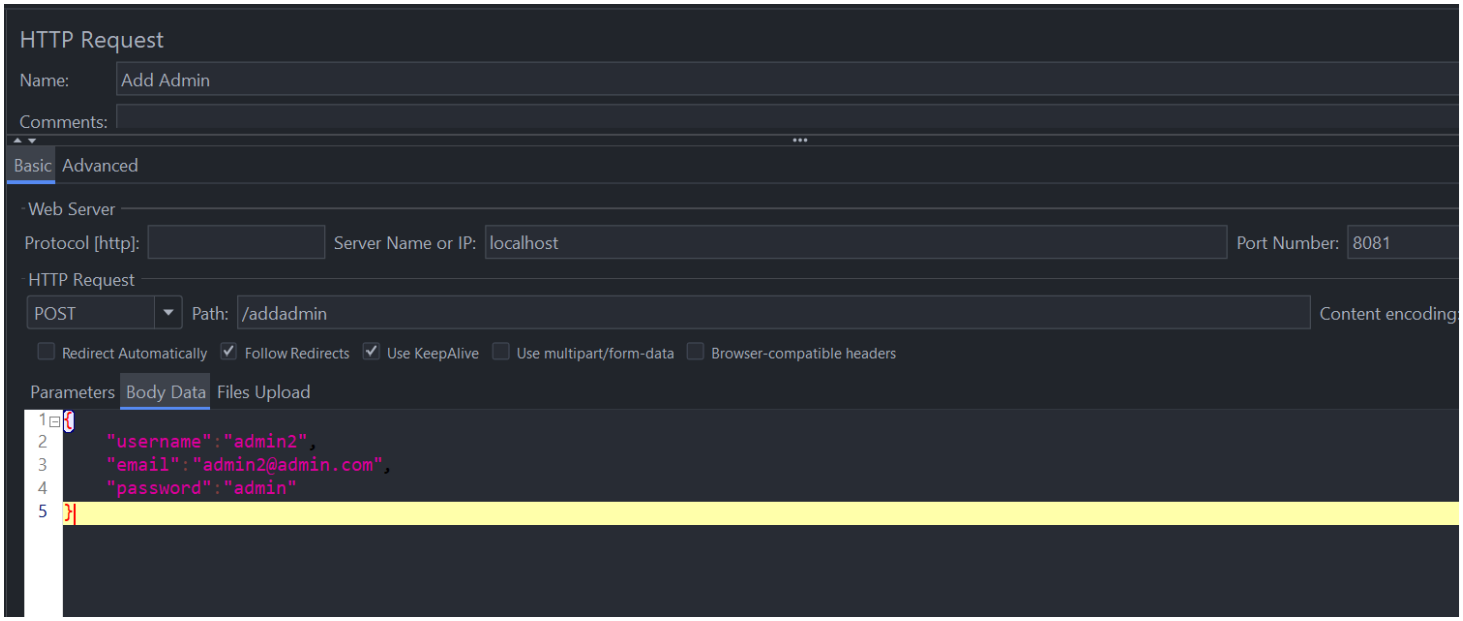

View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Get All Queries
Sampler result
Request
Response data
Thread Name:Capstone Project 1-1
Sample Start:2022-11-28 15:22:50 IST
Load time:30
Connect Time:2
Latency:30
Size in bytes:658
Sent bytes:163
Headers size in bytes:162
Body size in bytes:496
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:
HTTPSampleResult fields:
ContentType: application/json
DataEncoding: null

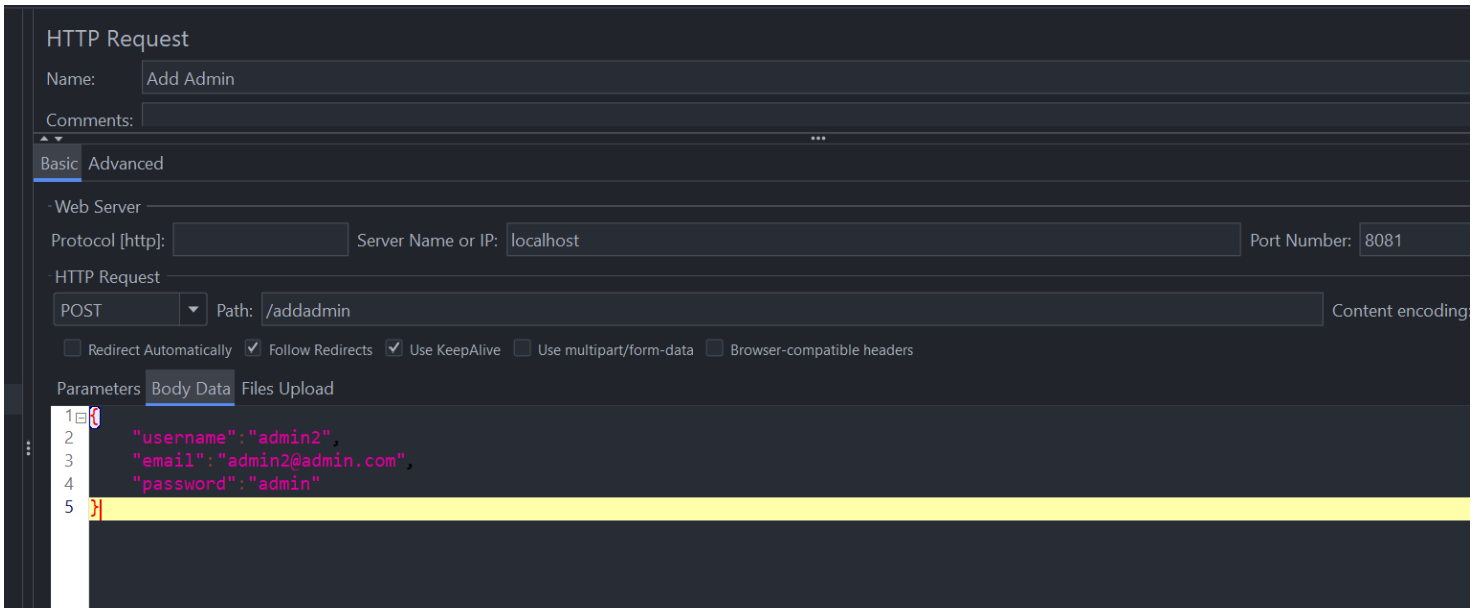

View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Get All Queries
Sampler result
Request
Response data
Response Body
Response headers
Find
Case sensitive
Regular exp.
[{"name":"Nirmit Surve","email":"survenirmit6@gmail.com","phoneno":"9702912526","query":"How to reach"},{"name":"Smit Surve","email":"smitsurve@gmail.com","phoneno":"","query":""},{"name":"Smit Surve","email":"smitsurve@gmail.com","phoneno":"9702912526","query":"Hssbsjajndns"},{"name":"Nirmit","email":"ujwala2002@gmail.com","phoneno":"8424057411","query":"Jsjsns"},{"name":"Nirmit Surve","email":"survenirmit6@gmail.com","phoneno":"9702912526","query":"Is it wheelchai accessible"}]

11) Add Admin

ID	Description	Data	Expected Result	Actual Result	Pass/Fail
1	Signup admin	username=admin2 email=admin2@admin.com password=admin	Adds admin into database	As Expected	Pass
2	Signup admin	username=admin email=admin2@admin.com password=admin	Username already exists	As Expected	Pass

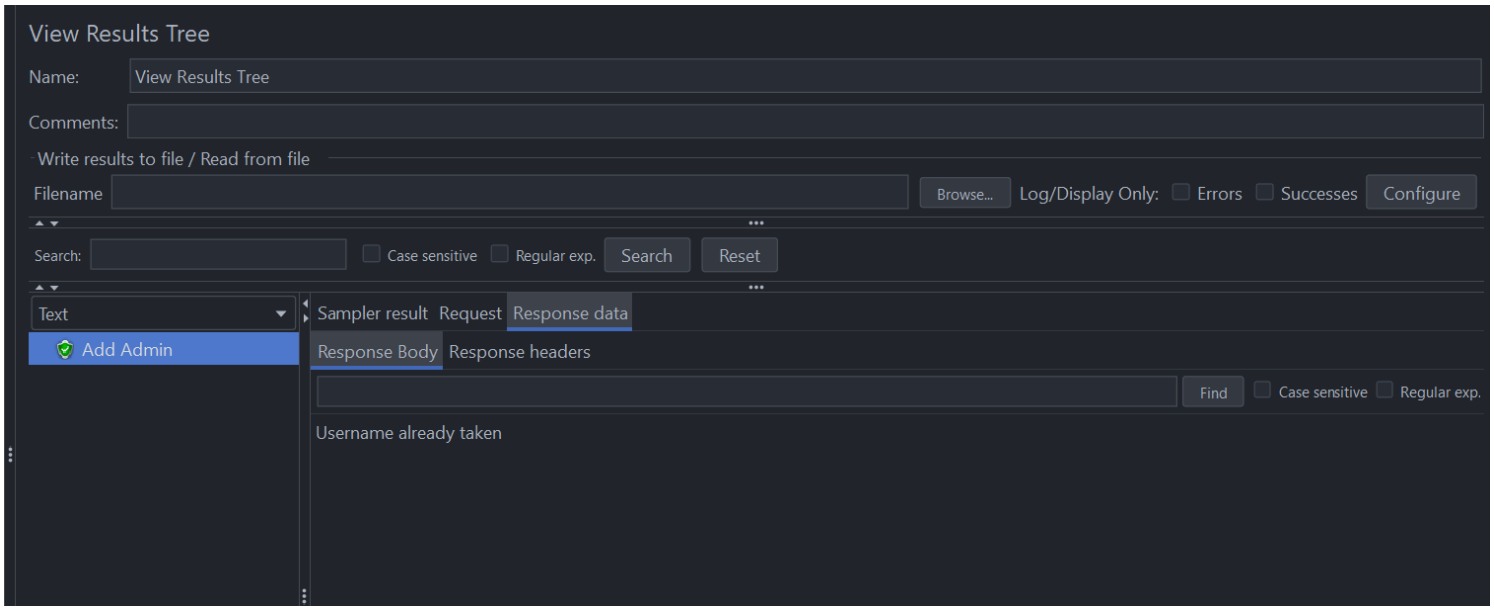
a) Test ID 1:




HTTP Request
Name: Add Admin
Comments:
Basic Advanced
Web Server
Protocol [http]:
Server Name or IP: localhost
Port Number: 8081
HTTP Request
POST
Path: /addadmin
Content encoding:
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters Body Data Files Upload
{
"username":"admin2",
"email":"admin2@admin.com",
"password":"admin"
}

b) Test ID 2:

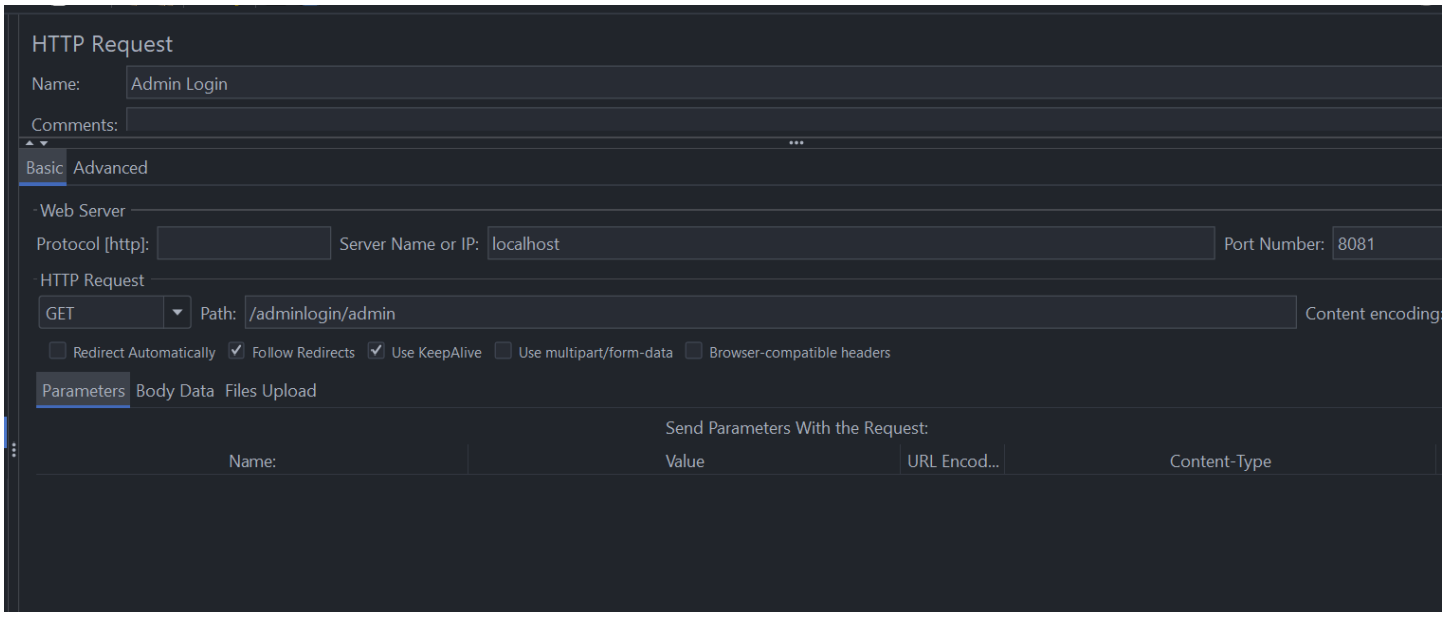

HTTP Request
Name: Add Admin
Comments:
Basic Advanced
Web Server
Protocol [http]:
Server Name or IP: localhost
Port Number: 8081
HTTP Request
POST
Path: /addadmin
Content encoding
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters Body Data Files Upload
{
"username":"admin",
"email":"admin2@admin.com",
"password":"admin"
}

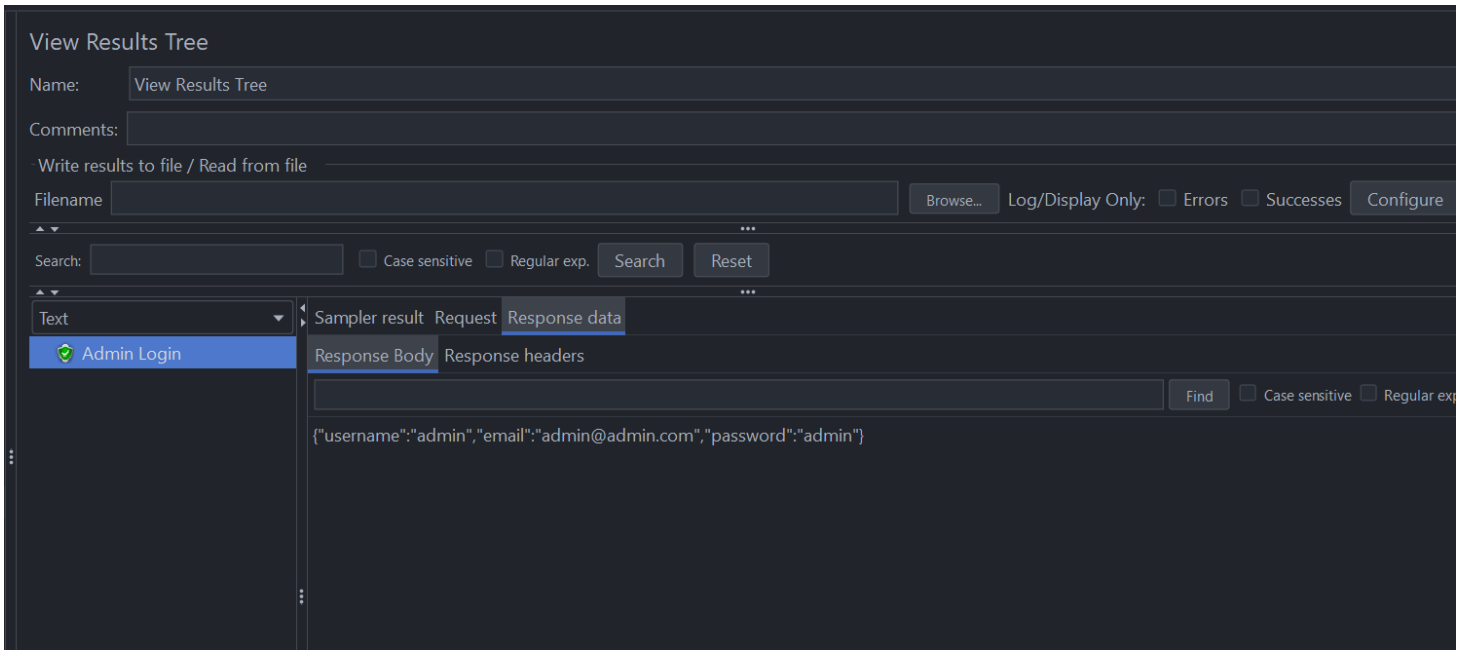


12) Admin Login

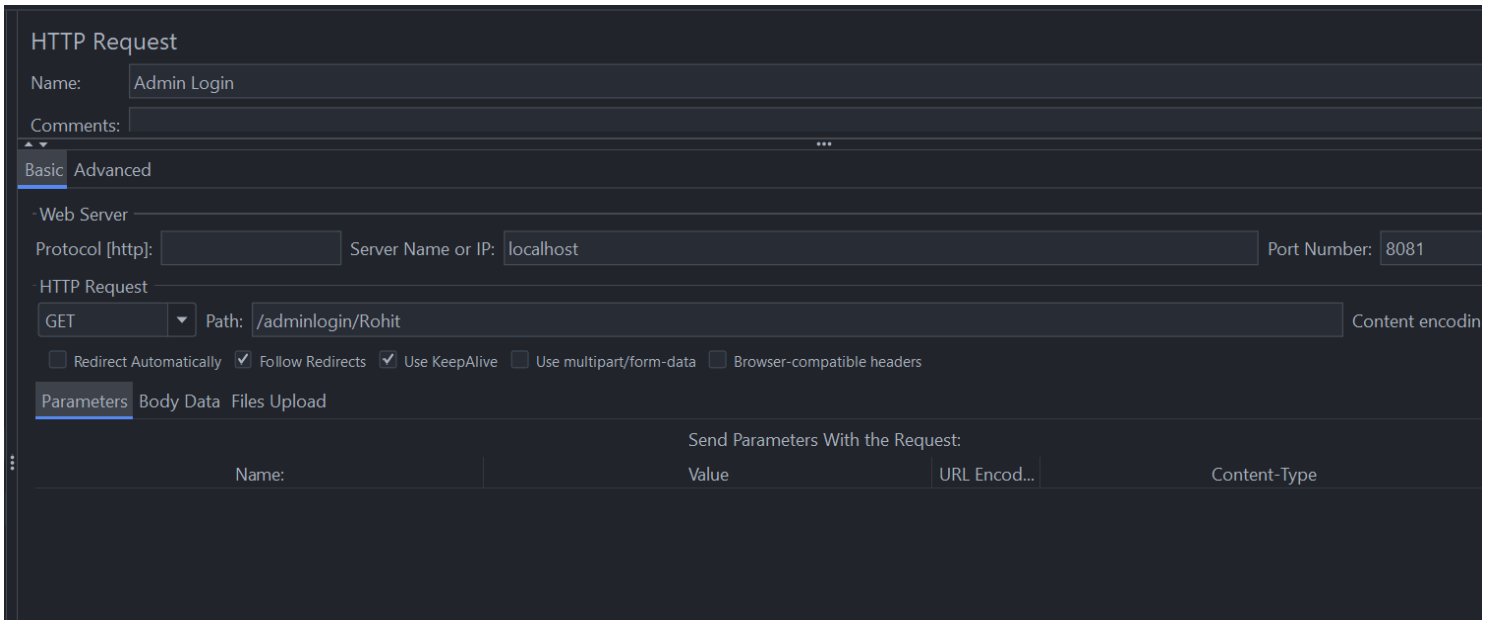
ID	Description	Data	Expected Result	Actual Result	Pass/Fail
1	Get Login Data By Username	username=admin	Returns info about the account	As Expected	Pass
2	Get Login Data By Username	username=Rohit	No response	As Expected	Pass

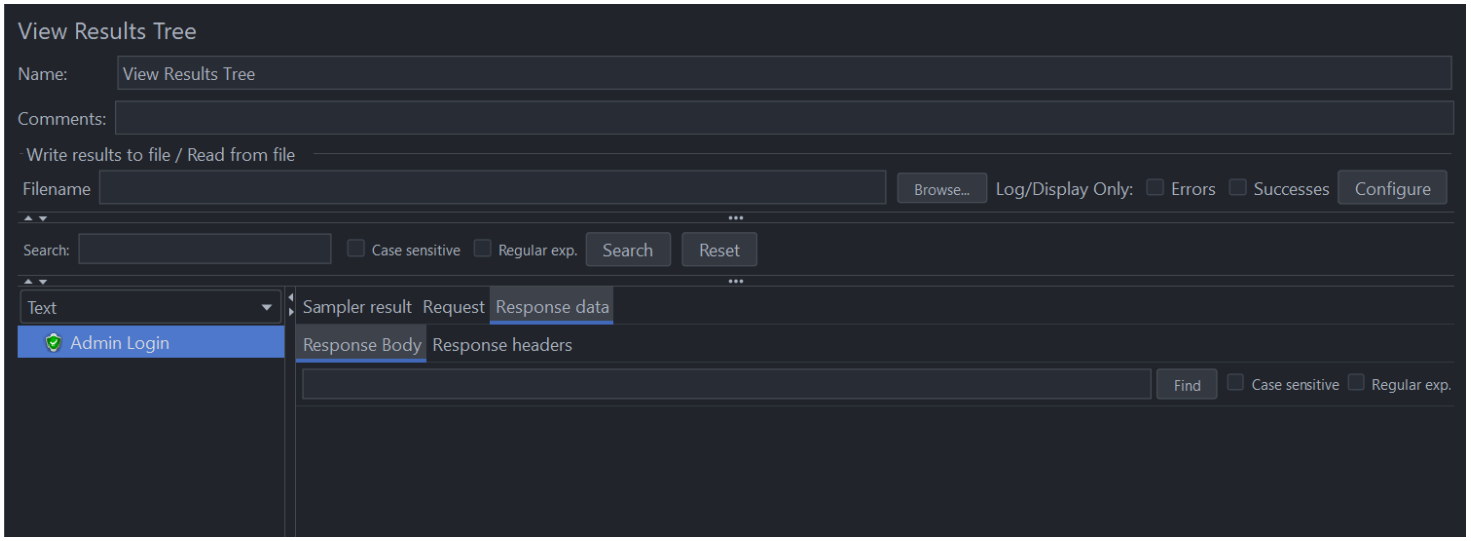
a) Test ID 1:


HTTP Request
Name: Admin Login
Comments:
Basic Advanced
Web Server
Protocol [http]:
Server Name or IP: localhost
Port Number: 8081
HTTP Request
GET
Path: /adminlogin/admin
Content encoding
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters Body Data Files Upload
Send Parameters With the Request:
Name:
Value
URL Encod...
Content-Type


View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Admin Login
Sampler result
Request
Response data
Response Body
Response headers
Find
Case sensitive
{"username":"admin","email":"admin@admin.com","password":"admin"}

b) Test ID 2:

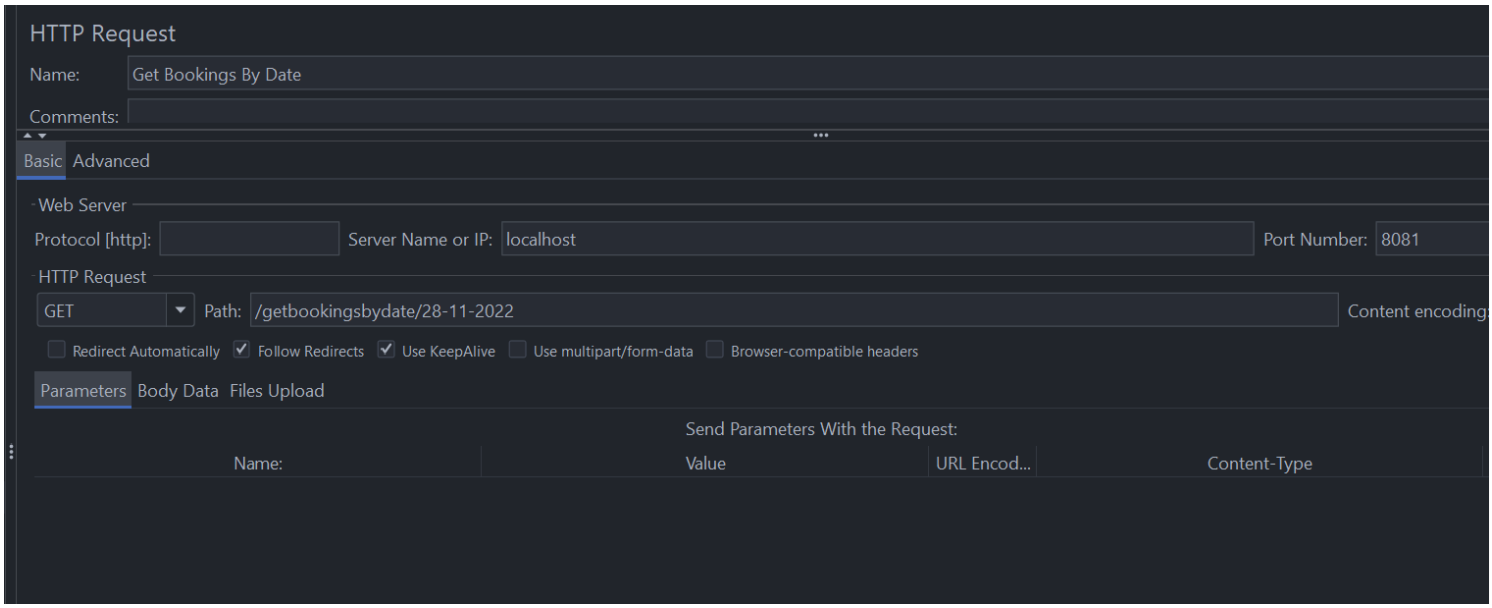

HTTP Request
Name: Admin Login
Comments:
Basic Advanced
Web Server
Protocol [http]:
Server Name or IP: localhost
Port Number: 8081
HTTP Request
GET
Path: /adminlogin/Rohit
Content encodin
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters Body Data Files Upload
Send Parameters With the Request:
Name:
Value
URL Encod...
Content-Type

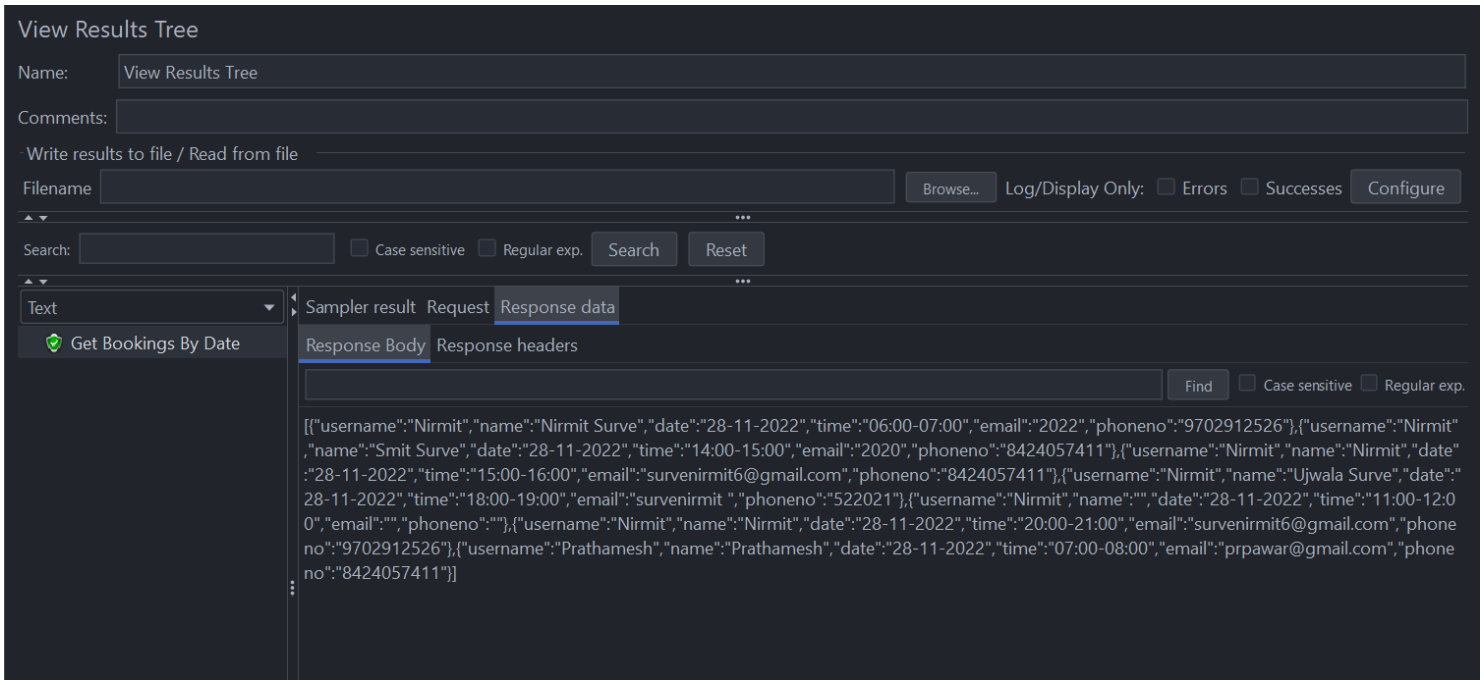

View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Sampler result
Request
Response data
Admin Login
Response Body
Response headers
Find
Case sensitive
Regular exp.

13) Get Bookings By date

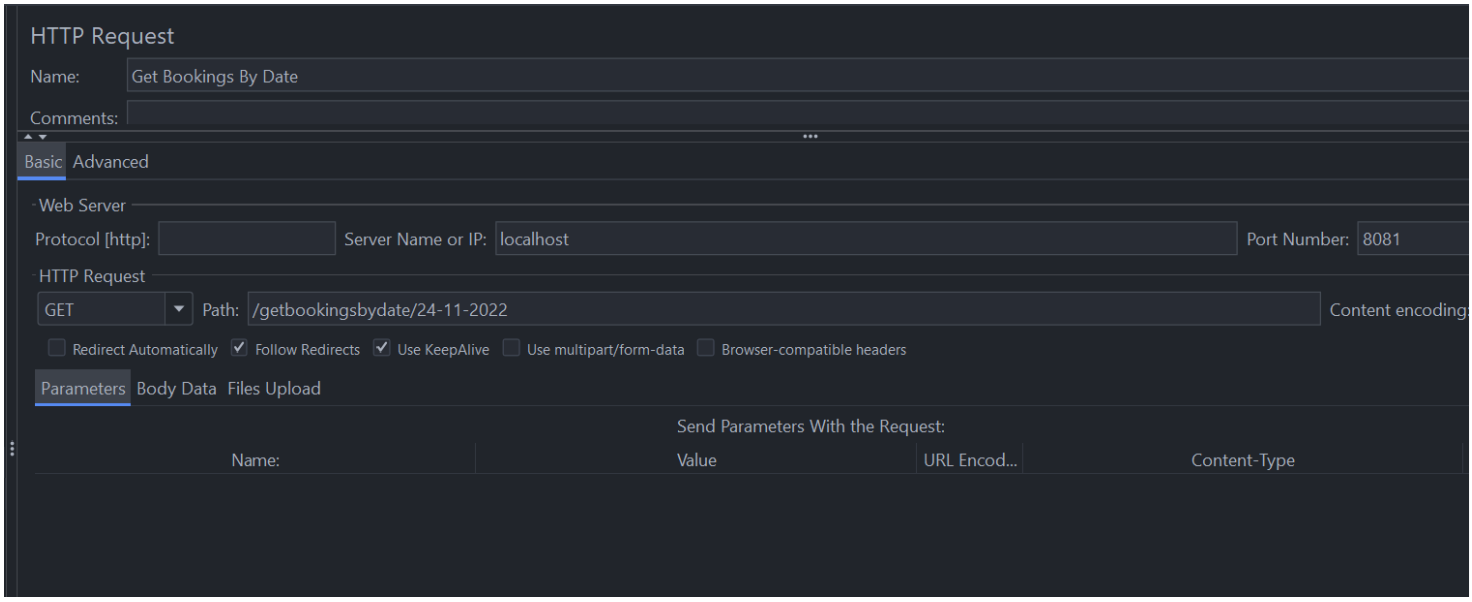
ID	Description	Data	Expected Result	Actual Result	Pass/Fail
1	Get booking info for a date	Date=28-11-2022	Returns info about the booking	As Expected	Pass
2	Get booking info for a date	Date=24-11-2022	Returns Null	As Expected	Pass

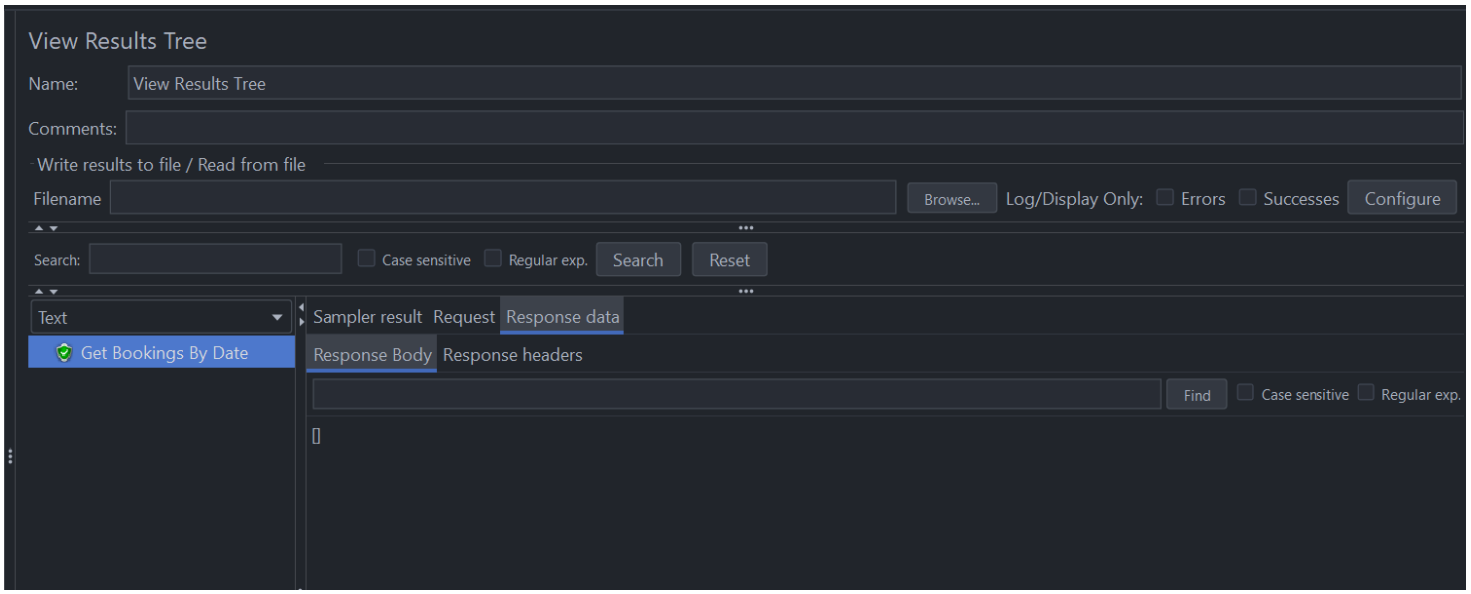
a) Test ID 1:




View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Get Bookings By Date
Sampler result
Request
Response data
Response Body
Response headers
Find
Case sensitive
Regular exp.
[{"username":"Nirmit","name":"Nirmit Surve","date":"28-11-2022","time":"06:00-07:00","email":"2022","phoneno":"9702912526"},{"username":"Nirmit","name":"Smit Surve","date":"28-11-2022","time":"14:00-15:00","email":"2020","phoneno":"8424057411"},{"username":"Nirmit","name":"Nirmit","date":"28-11-2022","time":"15:00-16:00","email":"survenirmit6@gmail.com","phoneno":"8424057411"},{"username":"Nirmit","name":"Ujwala Surve","date":"28-11-2022","time":"18:00-19:00","email":"survenirmit ","phoneno":"522021"},{"username":"Nirmit","name":"","date":"28-11-2022","time":"11:00-12:00","email":"","phoneno":""},{"username":"Nirmit","name":"Nirmit","date":"28-11-2022","time":"20:00-21:00","email":"survenirmit6@gmail.com","phoneno":"9702912526"},{"username":"Prathamesh","name":"Prathamesh","date":"28-11-2022","time":"07:00-08:00","email":"prpawar@gmail.com","phoneno":"8424057411"}]

b) Test ID 2:


HTTP Request
Name:
Get Bookings By Date
Comments:
Basic
Advanced
Web Server
Protocol [http]:
Server Name or IP:
localhost
Port Number:
8081
HTTP Request
GET
Path:
/getbookingsbydate/24-11-2022
Content encoding:
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data
Browser-compatible headers
Parameters
Body Data
Files Upload
Send Parameters With the Request:
Name:
Value
URL Encod...
Content-Type


View Results Tree
Name:
View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Search:
Case sensitive
Regular exp.
Search
Reset
Text
Sampler result
Request
Response data
Get Bookings By Date
Response Body
Response headers
Find
Case sensitive
Regular exp.

Conclusion:

The user convenience market is currently the most important market now. Every little thing or service has its own app now. Nowadays people use apps more than they talk. Currently, there are very few apps or websites that provide turf booking without contacting the turf owner by phone.So, in order to solve this problem I have created a Turf Booking Application which provides the user to easily book or schedule a slot.Using this app the user can book a slot ,view his/her bookings, rate the service provided by the location,etc. For the turf owners I have created an app for admins that can view the bookings by date and view user queries and reply to them regarding the same.