**4.IMPLEMENTATION**

**4.1 TOOLS**

A software or a programming tool is a set of computer programs that are used by the developers to create, maintain, debug, or support other applications and programs. Software development tools are simply tools (generally software themselves) that programmers practice to create other software.

The company has incorporated many of the AWS products into its solution, including Amazon EC2, Elastic Load Balancing, Amazon RDS, Amazon S3, Amazon EBS, and Amazon CloudWatch.

Tools Used by redBus:

1. Android Studio
2. Eclipse IDE
3. Ionic (Hybrid App): WebStorm
4. React-Native (Hybrid App): WebStorm
5. Xamarin
6. Log Management: Logstash, ELK Stack
7. Server: Tomcat
8. Deployment: Docker & Ansible
9. Code Repository: Git

**4.2 TECHNOLOGY**

A general term covering the development methods, programming languages, and tools to support them that may be used in the development of software. It is used in production of goods or services to accomplish objectives. Benefits of technology include saving time, cost efficiency, ease of access to information, better learning techniques, etc.

Technologies used in redBus:

* **Backend:** Java, Scala, Python, Go, Rust, NodeJS
* **Frontend:** ReactJS
* **App:** Kotlin (Android), Swift(iOS)
* **Server-side**: ASP.NET, PHP
* **Client-side:** JavaScript
* **JS library:** jQuery
* **SaaS:** AWS
* **Framework:** Flutter (Android)
* **DB’s:** MySQL, Postgres
* **Data platform:** S3, Druid, RDS

**DESCRIPTION ABOUT TECHNOLOGIES:**

JAVA: Java is a popular programming language, created in 1995. It is owned by Oracle, and more than 3 billion devices run Java. It is used for:

* Mobile applications (especially Android apps)
* Desktop applications
* Web applications
* Web servers and application servers
* Games
* Database connection

SCALA: Scala combines object-oriented and functional programming in one concise, high-level language. Scala's static types help avoid bugs in complex applications, and its JVM and JavaScript runtimes let you build high-performance systems with easy access to huge ecosystems of libraries.

PYTHON: Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

GO: Go is a statically typed, compiled programming language designed at Google by Robert Griesemer, Rob Pike, and Ken Thompson. Go is syntactically like C, but with memory safety, garbage collection, structural typing, and CSP-style concurrency.

NODE JS: Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser.

REACT JS: React is a free and open-source front-end JavaScript library for building user interfaces based on UI components. It is maintained by Meta and a community of individual developers and companies. React can be used as a base in the development of single-page or mobile applications.

KOTLIN: Kotlin is a cross-platform, statically typed, general-purpose programming language with type inference. Kotlin is designed to interoperate fully with Java, and the JVM version of Kotlin's standard library depends on the Java Class Library, but type inference allows its syntax to be more concise.

SWIFT: Swift is a general-purpose, multi-paradigm, compiled programming language developed by Apple Inc. and the open-source community

AWS: Amazon Web Services, Inc. is a subsidiary of Amazon providing on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered pay-as-you-go basis.

MYSQL: MySQL is an open-source relational database management system. Its name is a combination of "My", the name of co-founder Michael Wideness’s daughter, and "SQL", the abbreviation for Structured Query Language.

POSTGRES: PostgreSQL, also known as Postgres, is a free and open-source relational database management system emphasizing extensibility and SQL compliance. It was originally named POSTGRES, referring to its origins as a successor to the Ingres database developed at the University of California, Berkeley.

HTML: HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

Git: Git is software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development.

JAVASCRIPT: JavaScript, often abbreviated as JS, is a programming language that conforms to the ECMAScript specification. JavaScript is high-level, often just-in-time compiled and multi-paradigm. It has dynamic typing, prototype-based object-orientation and first-class functions.

JQUERY: jQuery is a JavaScript library designed to simplify HTML DOM tree traversal and manipulation, as well as event handling, CSS animation, and Ajax.

PHP: PHP is an open-source, interpreted, and object-oriented scripting language that can be executed at the server-side. PHP is well suited for web development.

ASP.NET: Build web apps and services that run on Windows, Linux, and macOS using C#, HTML, CSS, and JavaScript. Get started for free on Windows, Linux, or macOS.

**4.3 Sample Code**

Sample code of handling payment error in phonepay:

package phonepay.silentpay;

import phonepay.silentpay.C9719d;

import android.content.Intent;

import com.phonepay.identity.auth.device.AuthError;

public class APayError extends Exception {

/\* renamed from: b \*/

public ErrorType f85683b;

/\* renamed from: c \*/

public AuthError f85684c;

public enum ErrorType {

AUTH\_ERROR,

APAY\_ERROR,

LOW\_MEMORY

}

public APayError(ErrorType errorType, String str) {

super(str);

this.f85683b = errorType;

}

/\* renamed from: a \*/

public static AuthError m21749a(Intent intent) {

if (intent.getExtras().containsKey("ERROR\_CAUSE"))

return new AuthError(intent.getExtras().getString("ERROR\_MESSAGE"), (AuthError.ERROR\_TYPE) intent.getExtras().getSerializable("AUTH\_ERROR\_TYPE"));

public static APayError fromIntent(Intent intent) {

if (intent == null)

return null;

try {

if (intent.getExtras() == null)

return null;

for (ErrorType errorType : ErrorType.values()) {

if (intent.getExtras().containsKey(errorType.name())) {

ErrorType errorType2 = ErrorType.AUTH\_ERROR;

if (errorType == errorType2 && intent.getExtras().containsKey("AUTH\_ERROR\_TYPE"))

return new APayError(errorType2, m21749a(intent));

if (intent.getExtras().containsKey("ERROR\_MESSAGE") && intent.getExtras().containsKey("ERROR\_CAUSE")) {

return new APayError(errorType, intent.getExtras().getString("ERROR\_MESSAGE"), (Throwable) intent.getExtras().getSerializable("ERROR\_CAUSE")); }

if (intent.getExtras().containsKey("ERROR\_MESSAGE")) {

return new APayError(errorType, intent.getExtras().getString("ERROR\_MESSAGE")); }

}

}

return null;

} catch (Exception e) {

C9710a.m21753c("APayError", "Error parsing Apay Error", e);

C9722e.m21773b(C9719d.C9720a.AUTHORIZE\_RESPONSE\_PARSING\_FAILED);

return new APayError(ErrorType.APAY\_ERROR, "Error parsing Apay Error", e);

}

}

public AuthError getAuthError() {

return this.f85684c; }

public ErrorType getErrorType() {

return this.f85683b; }

public APayError(ErrorType errorType, String str, Throwable th) {

super(str, th);

this.f85683b = errorType; }

public APayError(ErrorType errorType, AuthError authError) {

this.f85683b = errorType;

this.f85684c = authError; }

}

**5.Testing**

Testing is the process of executing a program to find errors. To make our software perform well it should be error-free. If testing is done successfully, it will remove all the errors from the software.

Principles of Testing: -

1. All the tests should meet the customer requirements.
2. To make our software, testing should be performed by a third party.
3. Exhaustive testing is not possible. As we need the optimal amount of testing based on the risk assessment of the application.
4. All the tests to be conducted should be planned before implementing it
5. It follows the Pareto rule (80/20 rule) which states that 80% of errors come from 20% of program components.
6. Start testing with small parts and extend it to large parts.

**5.1 TYPES OF TESTING**

**5.1.1 Unit Testing:** It is the first level of functional testing in order to test any software. In this, the test engineer will test the module of an application independently or test all the module functionality is called unit testing. The primary objective of executing the unit testing is to confirm the unit components with their performance. Here, a unit is defined as a single testable function of a software or an application. And it is verified throughout the specified application development phase. To make ticket booking reality, we designed some major building blocks. Let us look under the hood:

* Home Page
* Profile Page (My Account)
* Booking & Booking History
* Agency/Bus info
* Seat Selection
* Payment
* Customer Service

**5.1.2 Integration Testing:** Once we are successfully implementing the unit testing, we will go integration testing. It is the second level of functional testing, where we test the data flow between dependent modules or interface between two features is called integration testing. If these modules are working fine, then we can add one more module and test again. And we can continue with the same process to get better results. we have integrated the above modules and checked for their performance:

1. Home

* Test for correct text and images
* Links to important pages (catalogue, account login, offers etc) to be visible and

Functional

2. Profile page

* Users should be able to search for relevant items and to be directed to the exact page.
* All agency services should be listed clearly
* Any ability to sort booking by agency name, price, bus type, route (source & destination).

3. Seat Selection

* Selected seats and all total prize should be displayed
* Remove, add, or modify the seated seats
* Apply offers and proceed to pay.

4. Payment

* Check different payment options
* Final amount is displayed
* Make sure that the delivery of the products is done efficiently

5. Customer service

* Customer service will review the issue raised by the customer.
* Feedback is taken by customer service from customers.

**5.1.3 Validation Testing:** The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. It ensures that the product meets the client's needs. It can also be defined as to demonstrate that the product fulfils its intended use when deployed in an appropriate environment.

1. Home page

* case 1: Home page is displayed successfully.
* case 2: Home page not displayed, return the project to developer's team.

2. Profile page

* case 1: Profile details page is displayed successfully.
* case 2: Profile details page not displayed, return the project developer's team

3.Seat Selection

* case 1: Seat will be selected successfully & prize will be displayed.
* case 2: Seats will not be selected i.e. presence of some bug in the code, return the project developer's team.

4.Payment

* case 1: Payment mode and Payment Transactions are successful.
* case 2: Transactions are not properly done by the application, return the project

developer's team.

5.Customer service

* case 1: Reviewing problems, responding to complaints and feedback are successful.
* case 2: Feedback issues, direct contact with user is not possible, return the project developer's team.

**5.1.4 System Testing System:** It includes testing of a fully integrated software system. Generally, a computer system is made with the integration of software. The software is developed in units and then interfaced with other software and hardware to create a complete computer system. System testing is a series of different types of tests with the purpose to exercise and examine the full working of an integrated software computer system against requirements. To check the end-to-end flow of an application or the software as a user is known as System testing. In this, we navigate all the necessary modules of an application and check if the end features or the end business works fine, and test the product as a whole system.

Usability: Online Customers are goal-oriented. They exit e-commerce websites/applications immediately if they cannot find their needed agencies, best route, prizes/offers.

Security: The application must comply with the regulations and protect online customers from information theft, identity theft etc.

Performance: The number of online transactions fluctuates daily and it increases dramatically

during the holiday/festival season.

Hence, the following challenges were taken by our website:

* + Compliance with security guidelines to safeguard customer data and identity
  + Compliance with accessibility standards to support multilingual markets and business regions
  + End to end testing and test management for large ecommerce transformation programs
  + Scalability and reliability of applications

**5.2 Test Cases**

Test Case acts as the starting point for the test execution, and after applying a set of input values, the application has a definitive outcome and leaves the system at some end point or also known as execution postcondition.

The parameters of the test case are: Test Case ID, Test Scenario, Test Case Description, Test Steps, Prerequisite, Test Data, Expected Result, Test Parameters, Actual Result, Environment Information, Comments.

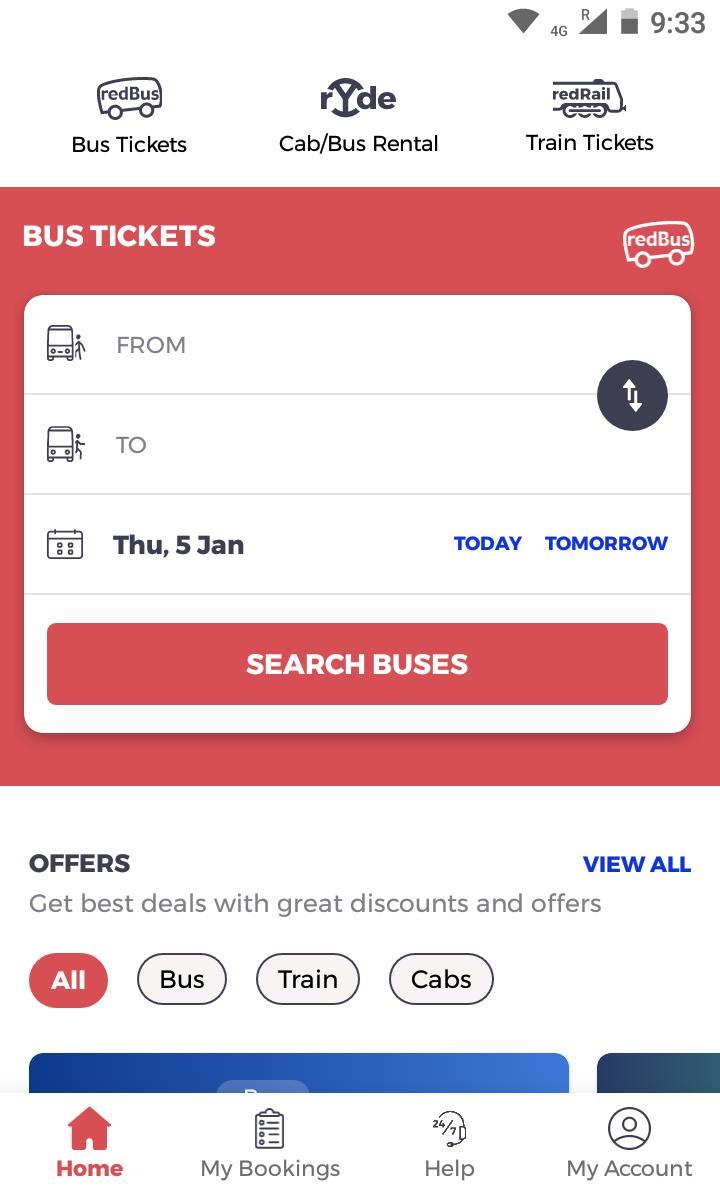
| **Test case ID:** Test\_01 |
| --- |
| **Test priority (Low/Medium/High):** Medium |
| **Medium Module name:** login |
| **Test title:** verify login with valid username and password |
| **Precondition:** User has invalid username and password |

| **Test Scenario ID** | **Test Scenario** | **Test Case ID** | **Test Steps** | **Input** | **Expected Result** | **Actual Result** | **Result/**  **Status** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Register** | | | | | | | |
| TS-1 | Check register application | TC-1.1 | Click Sign in | Valid Phone & OTP | Registered page will be displayed | As Expected | Pass/Successful |
| Tc-1.2 | Click Sign in | In Valid Phone & OTP | Registered page is not displayed | Show  try  again | Fail/Unsuccessful |
| **Login** | | | | | | | |
| TS-2 | Check Login | Tc-2.1 | Click Login | Registered Phone & Valid OTP | Login Access will be provided | As expected | Pass/Successful |
| Tc-2.1 | Click Login | Registered Phone & In-Valid OTP | Will not  verify us  OTP | As  Expected | Fail/Unsuccessful |
| Tc-2.1 | Click Login | Unregistered Phone no | Will not verify the Phone no | As Expected | Fail/ Unsuccessful |
| Tc-2.1 | Click Login | Unregistered Phone & In-Valid OTP | Will not verify the Phone no | As Expected | Fail/ Unsuccessful |

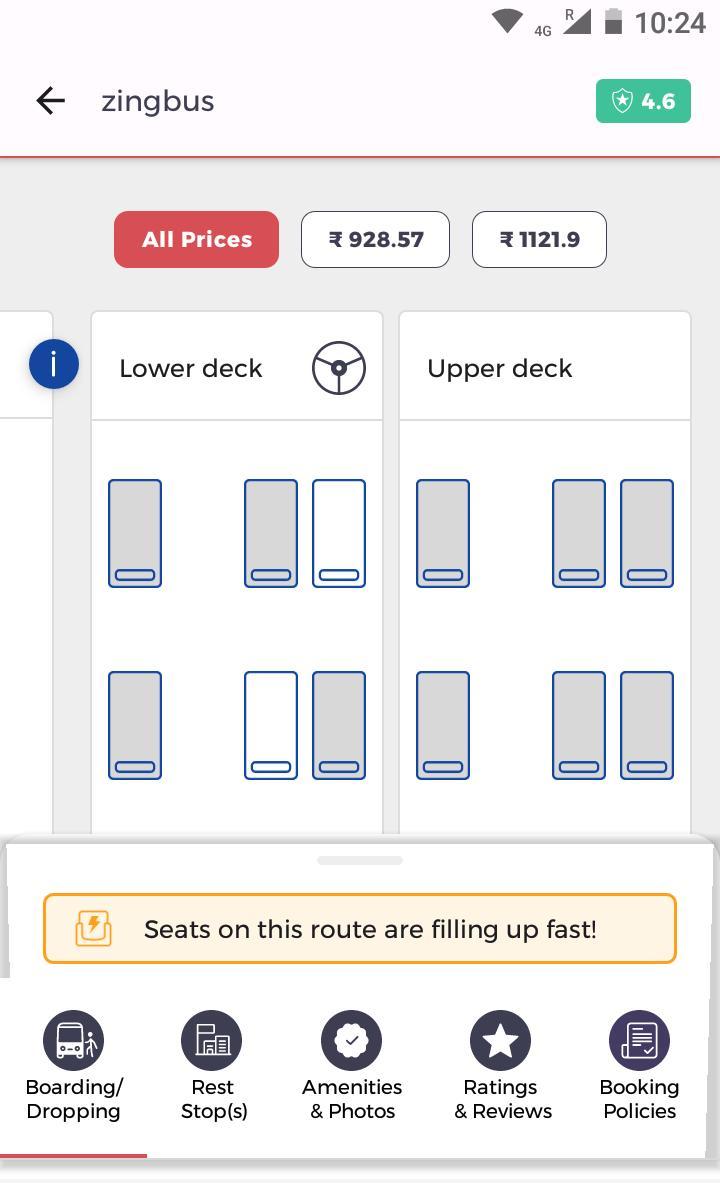
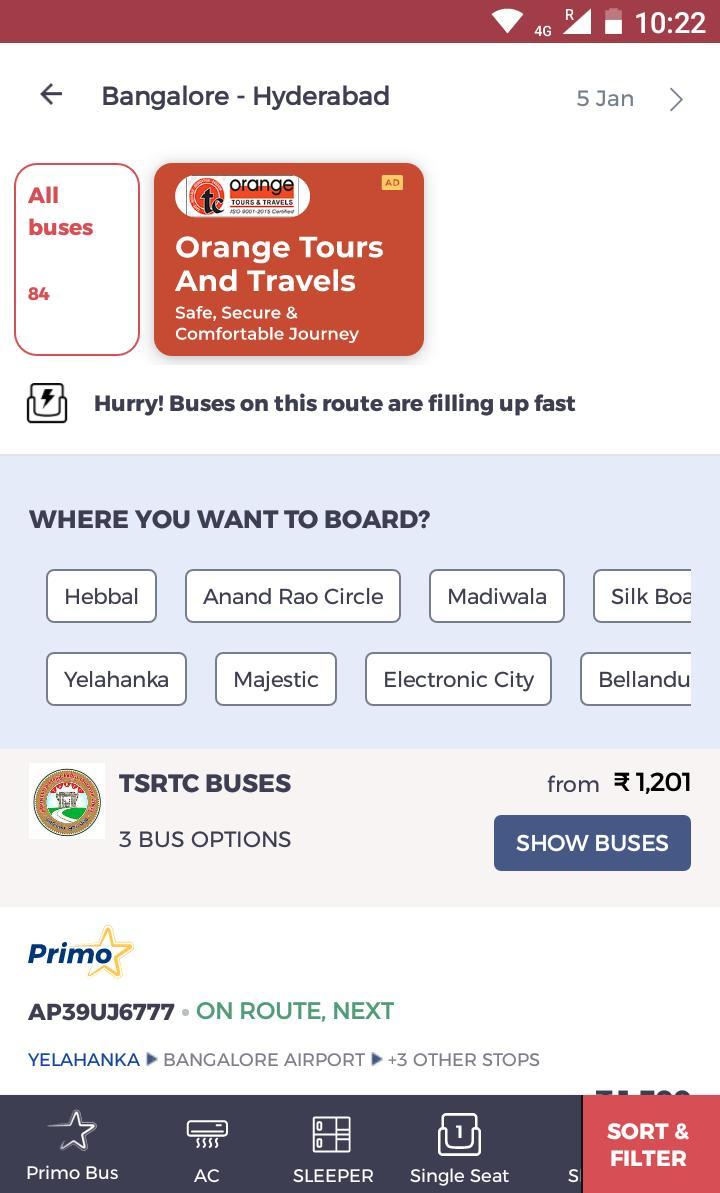
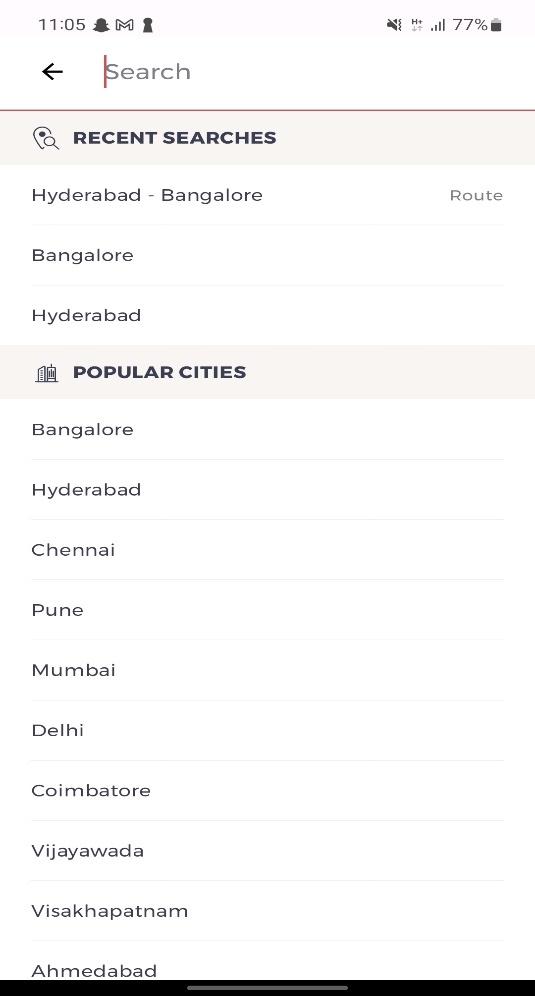
**5.3 User Interfaces**

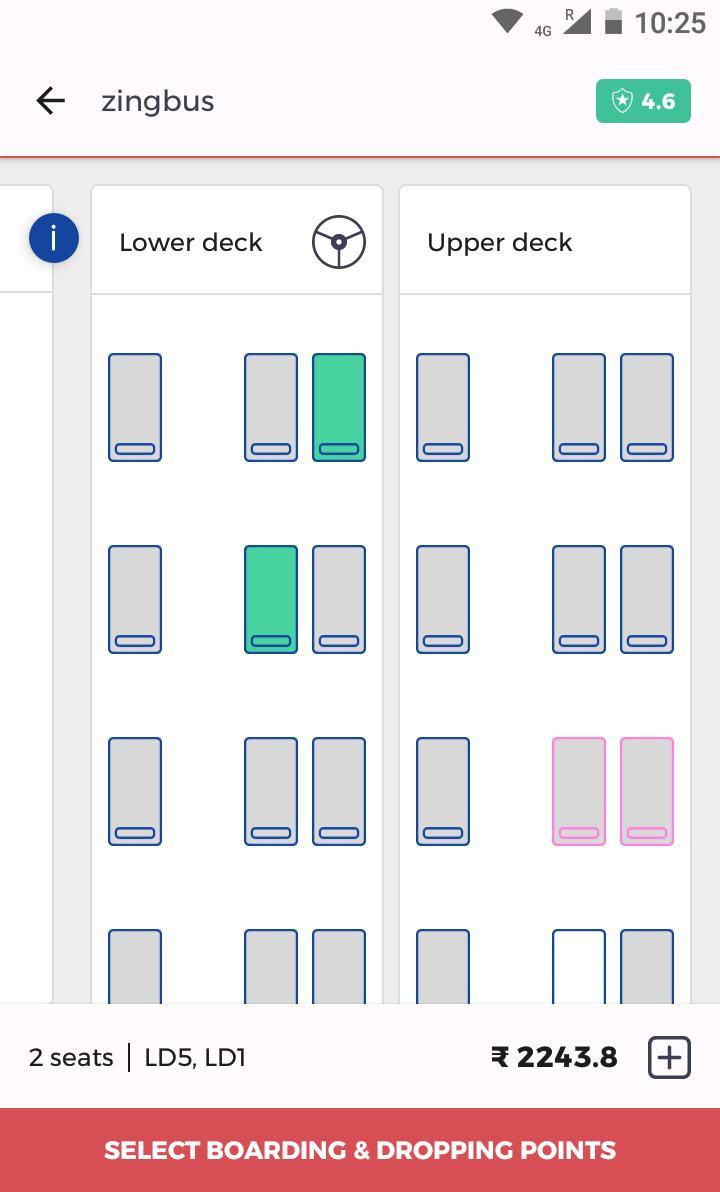
The user interface (UI) is the point of human-computer interaction and communication in a device. This can include display screens, keyboards, a mouse and the appearance of a desktop. It is also the way through which a user interacts with an application or a website. User interface is the front-end application view to which user interacts in order to use the software.

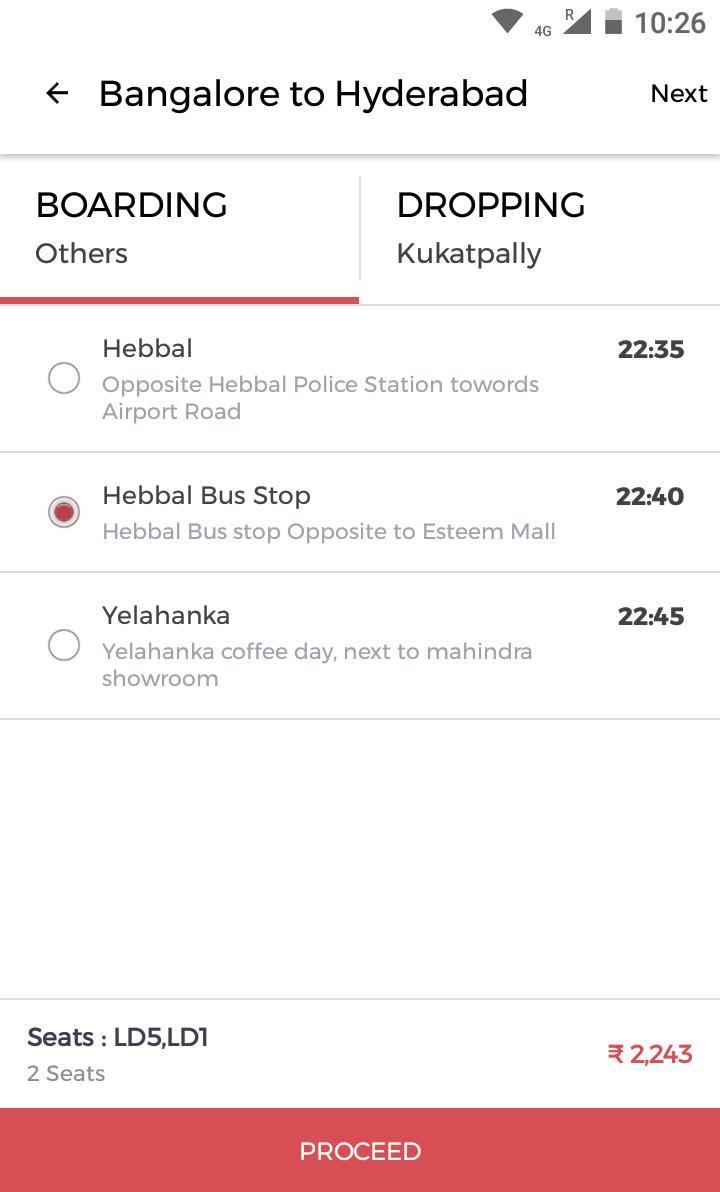
following is some of the sample user interfaces which we are going to use in Red Bus application:

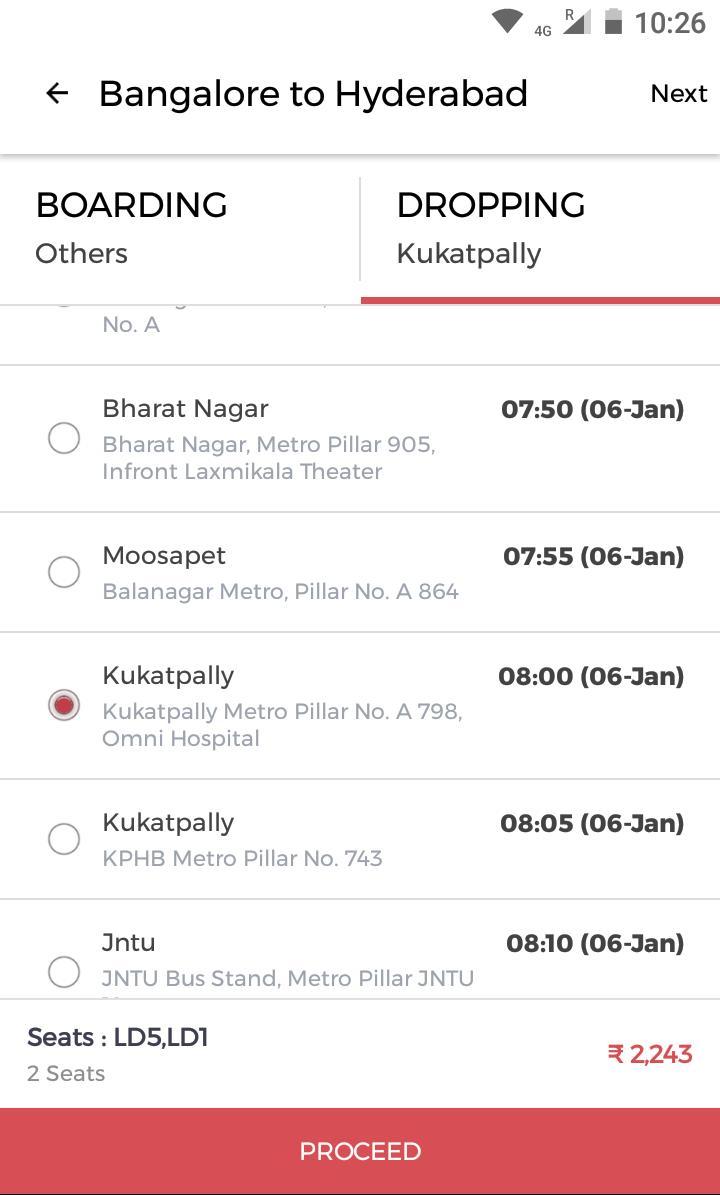


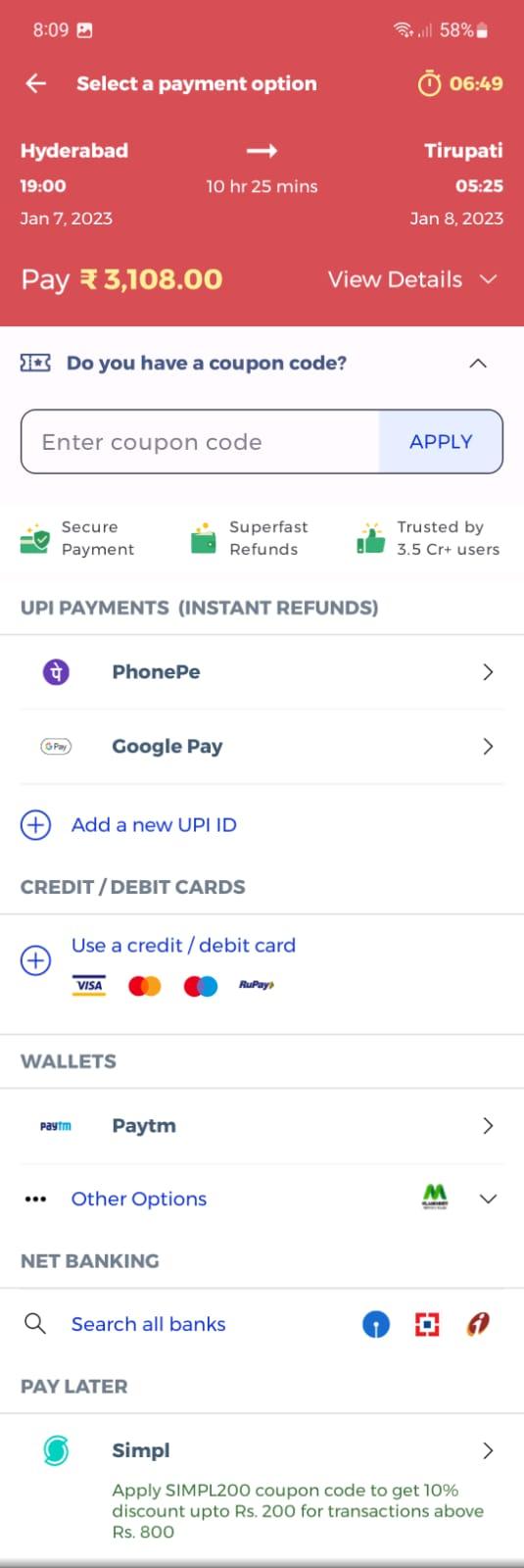
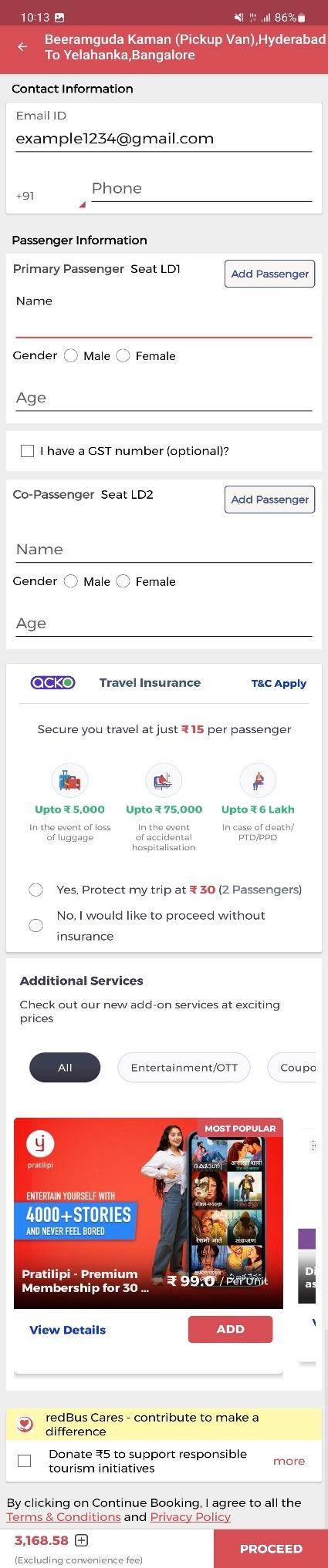








***5.3.7 Boarding place selection***



***5.3.9 Passenger info 5.3.10 Payment options***