Software Design Document

for

FastTrack

Prepared by

Akhil S Nair (KTE22CS009)

Alwin Philip (KTE22CS012)

Edwin Varkey (KTE22CS028)

Jessin Sunny (KTE22CS036)

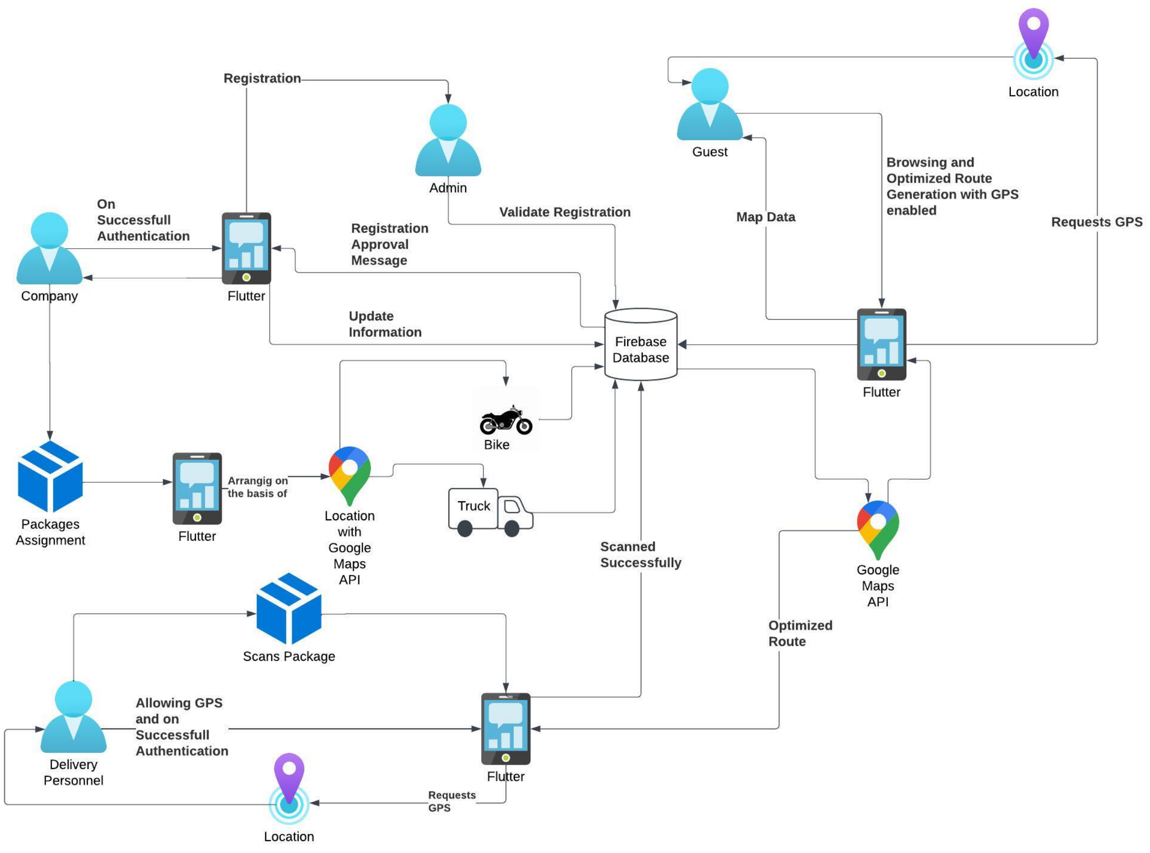
Department of Computer Science and Engineering

Rajiv Gandhi Institute of Technology,Kottayam

Contents

1 System Design

1.1 System Architecture

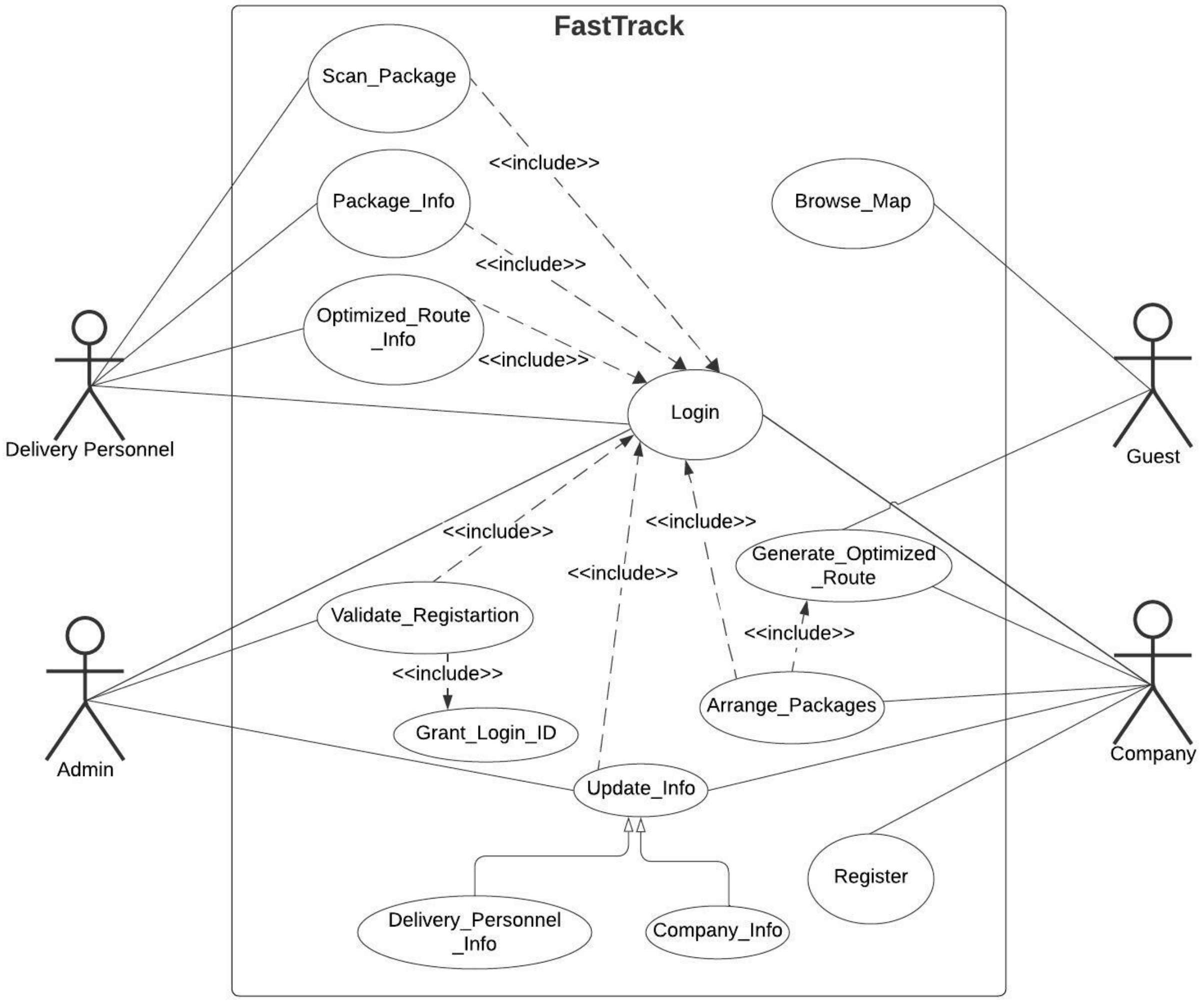


The figure represents the system architecture for FastTrack, showcasing the interaction between various stakeholders (Admin, Company, Delivery Personnel, and Guest) and technological components.

It uses Flutter for the front-end application, connected to a Firebase Database for centralized data management. The system integrates the Google Maps API for route optimization and location tracking. Key features include company registration and validation by the admin, package assignment and scanning by delivery personnel, and GPS-enabled optimized route generation for efficient deliveries. The architecture also supports different delivery modes like bikes and trucks and allows guests to browse routes or map data.

This system aims to streamline logistics, enhance delivery efficiency, and provide real-time updates for all users involved.

1.2 USE CASE DIAGRAM

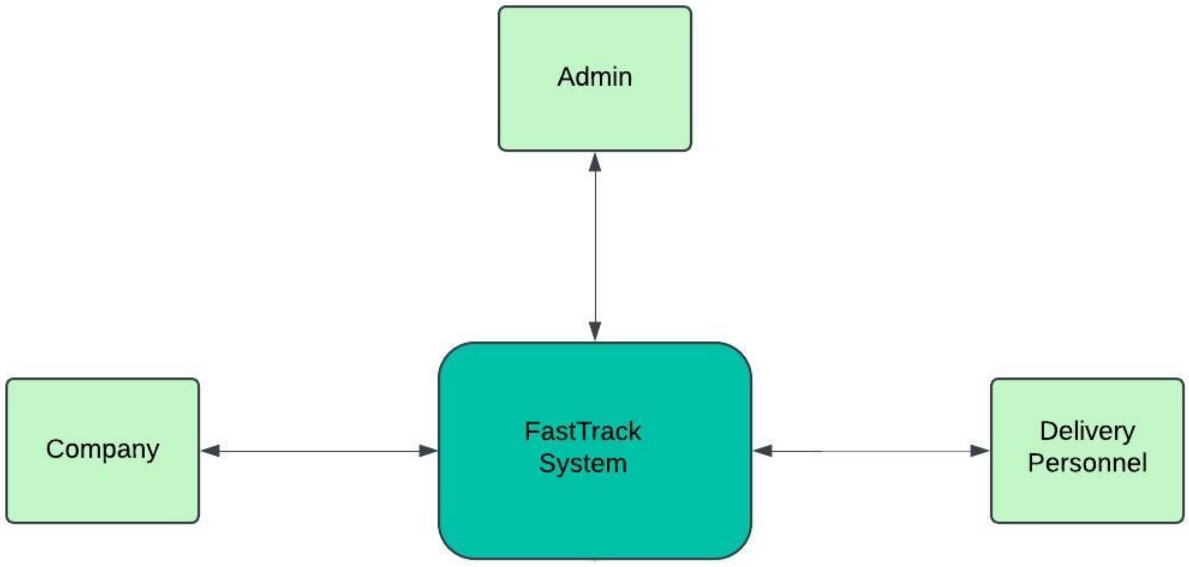


1.3 User Specific Functionalities

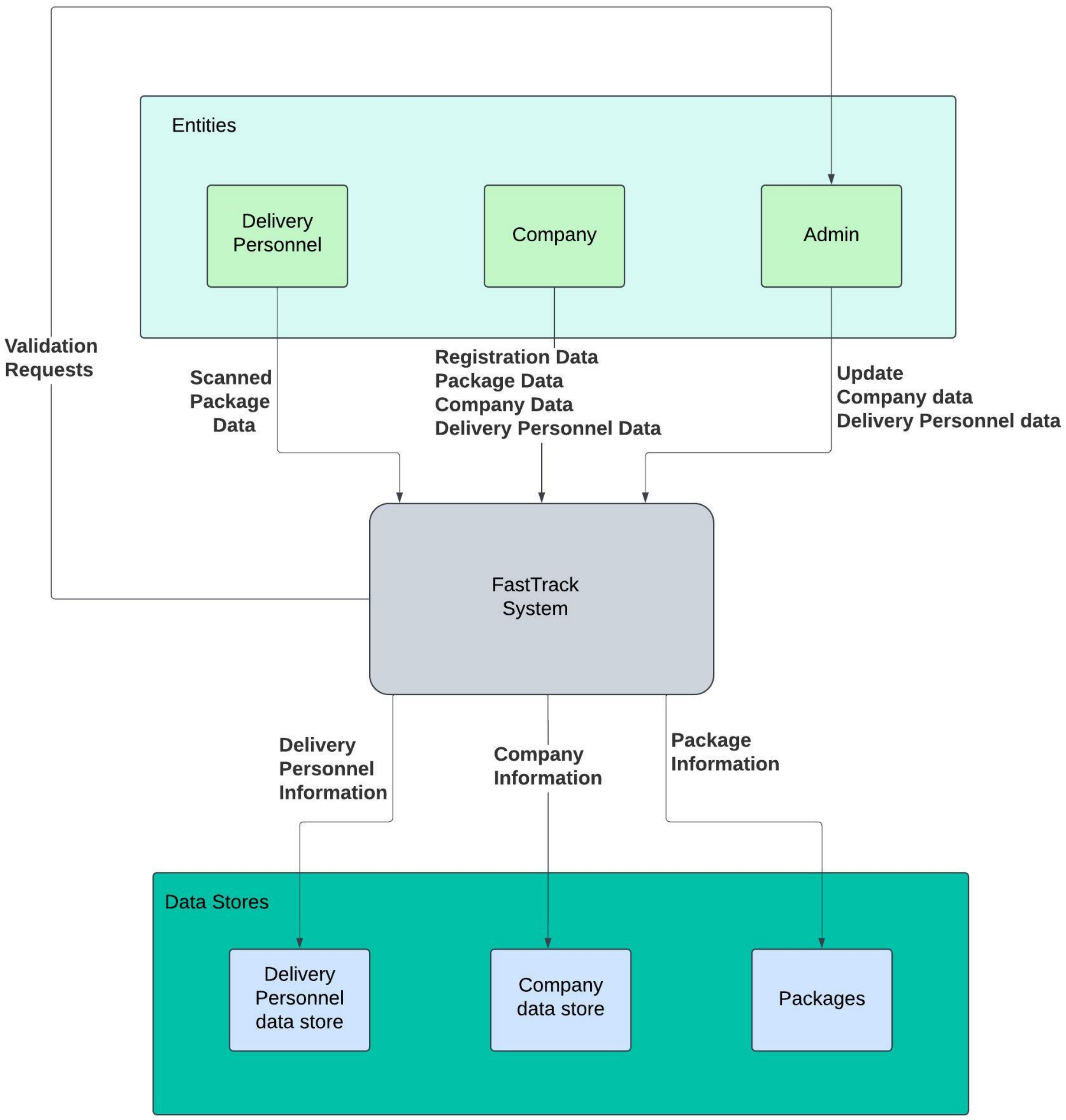
- Delivery Personnel
    1. Login
    2. View package information
    3. Scan packages
    4. Get the optimized route
- Company
    1. Register
    2. Login
    3. Updates company and delivery personnel information
    4. Arrange the arrived packages
    5. Generate optimized route for delivery
    6. Locate each delivery personnel current location
- Guest
    1. Explore the map
    2. Add destinations
    3. Calculate optimized route
- Admin
    1. Login
    2. Validate Registration for Company
    3. Grant company and delivery personnel login ID
    4. Updates company and delivery personnel information

1.4 Data Flow Diagram

1.4.1 0-Level DFD

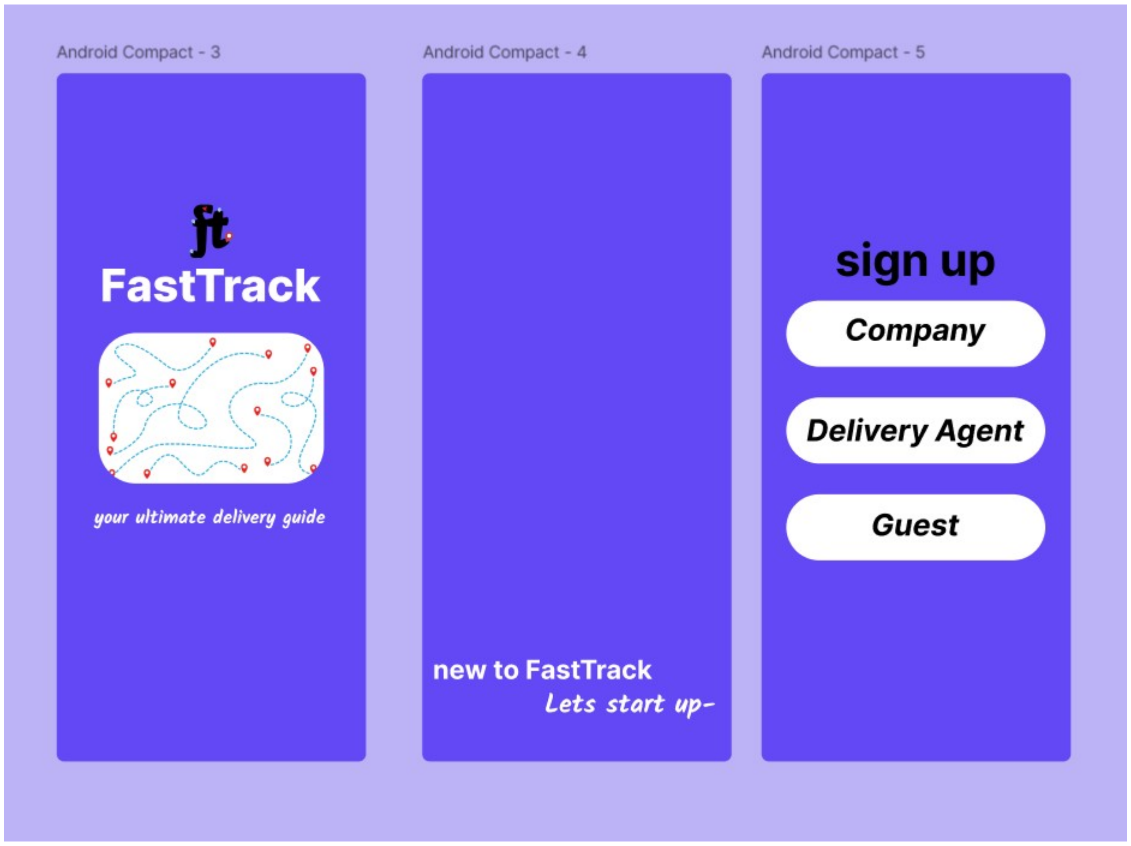


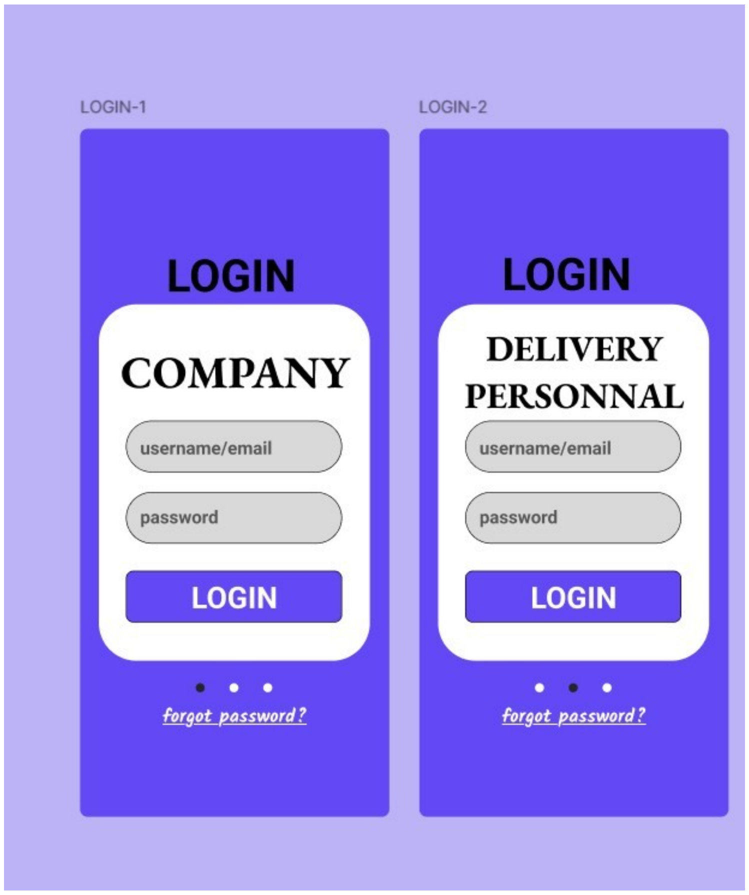
1.4.2 1-Level DFD

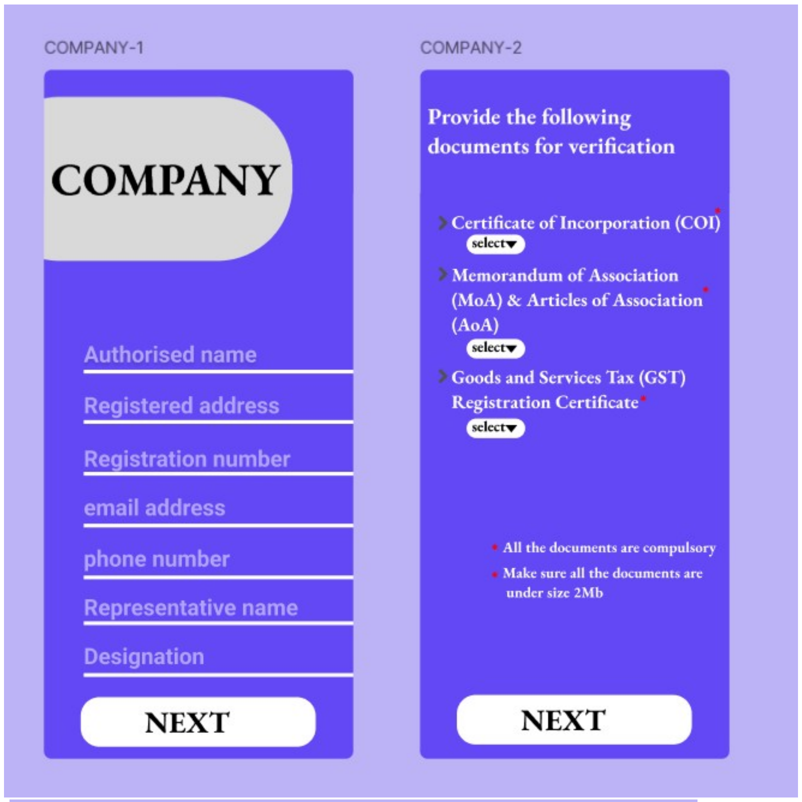


2 Detail Design

2.1 UI Design




LOGIN-1
LOGIN
COMPANY
username/email
password
LOGIN
forgot password?
LOGIN-2
LOGIN
DELIVERY PERSONNAL
username/email
password
LOGIN
forgot password?

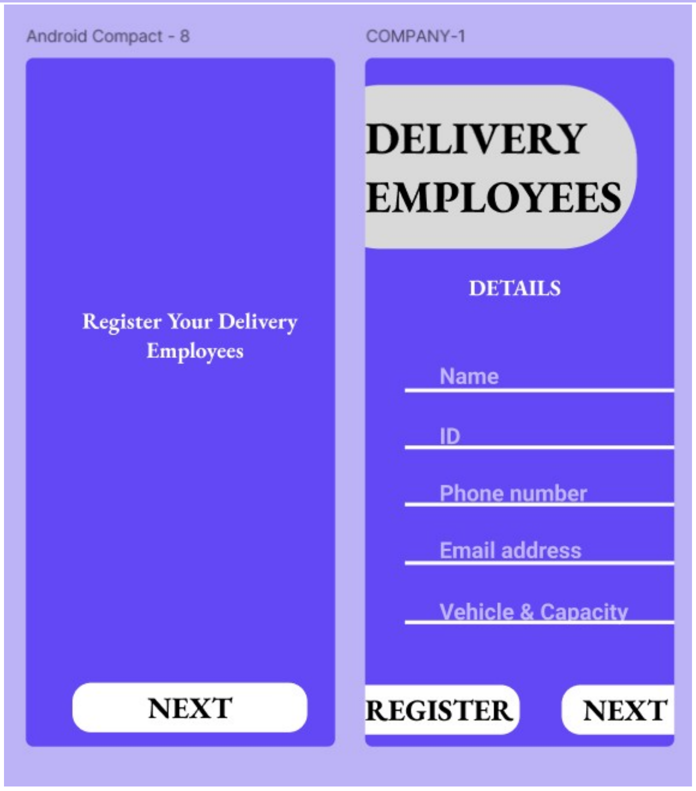


Provide the following documents for verification

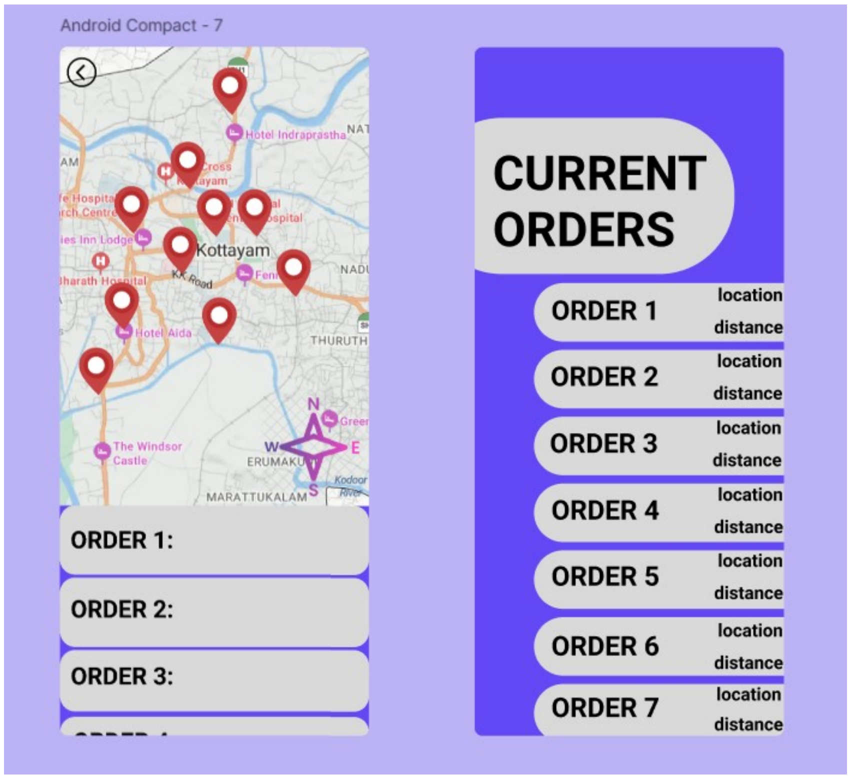
- Certificate of Incorporation (COI)*
- Memorandum of Association (MoA) & Articles of Association (AoA)*
- Goods and Services Tax (GST) Registration Certificate*

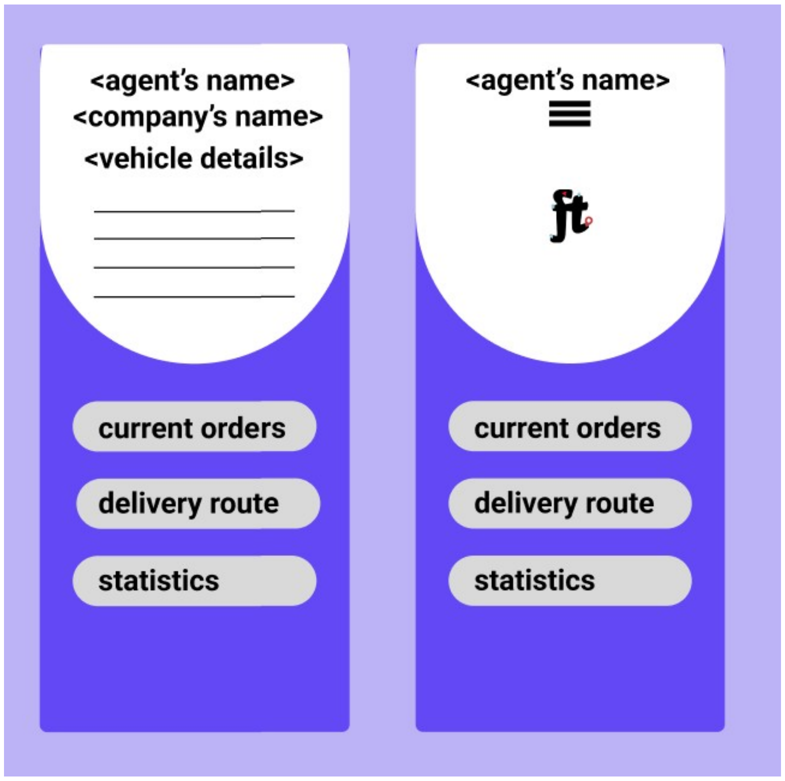
* All the documents are compulsory

* Make sure all the documents are under size 2Mb

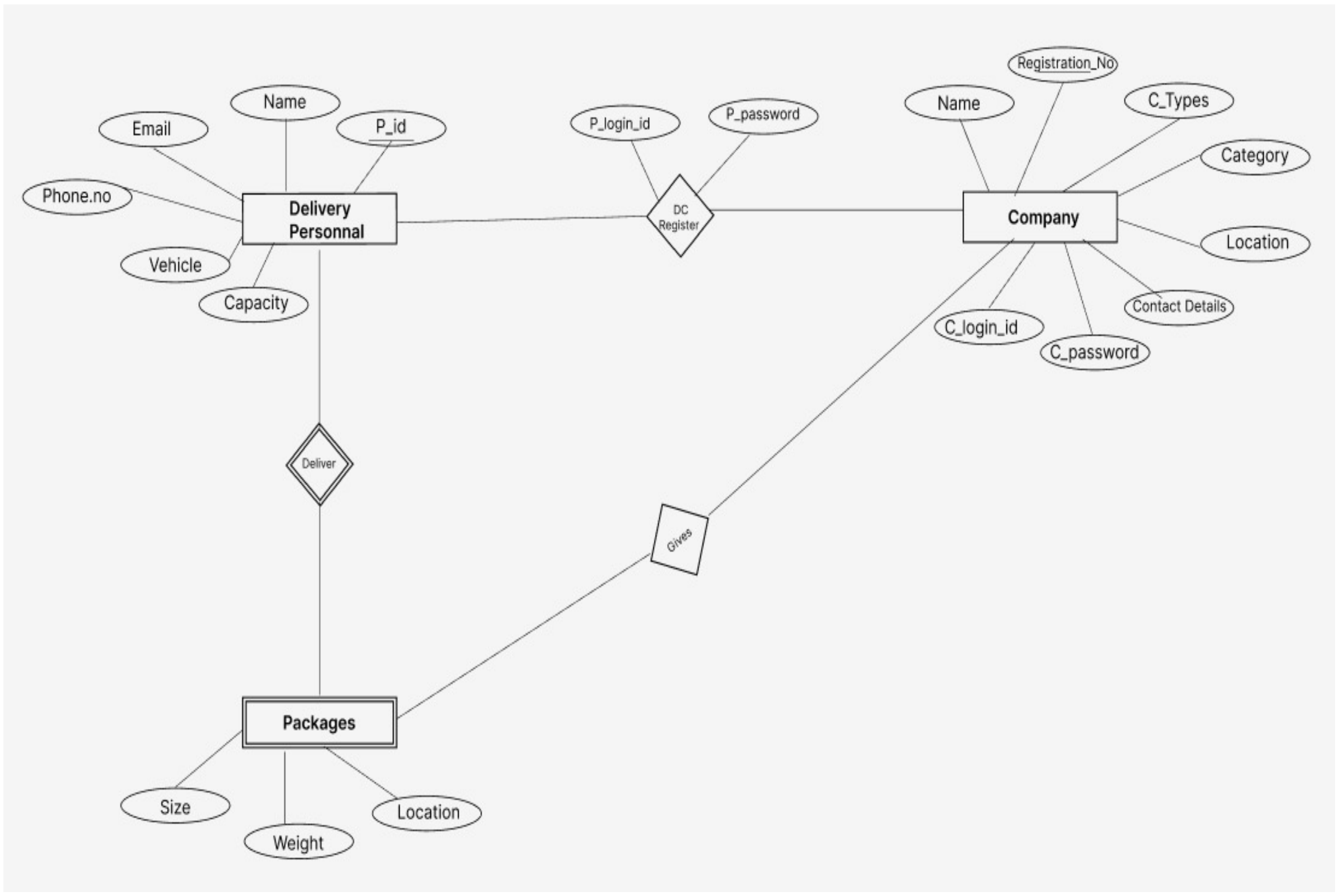


Register Your Delivery Employees


Android Compact - 7
Kottayam
ORDER 1:
ORDER 2:
ORDER 3:
CURRENT ORDERS
ORDER 1 location distance
ORDER 2 location distance
ORDER 3 location distance
ORDER 4 location distance
ORDER 5 location distance
ORDER 6 location distance
ORDER 7 location distance


<agent's name>
<company's name>
<vehicle details>
current orders
delivery route
statistics
<agent's name>
current orders
delivery route
statistics

2.2 ER Diagram



2.3 Algorithm Design

Algorithm 1: Delivery Personnel or Company or Admin Authentication Algorithm

```
Input  : LoginID and Password
Output: Authentication status (Success/Failure message with token if successful)
1  Input the LoginID and Password from the user.
2  Search for the LoginID in the user database.
3  if the user with the LoginID is found then
4      Validate the Password.
5      if the Password matches then
6          Generate an Authentication Token.
7          Grant access.
8          Display ”Authentication Successful”.
9      end
10     else
11         Display ”Invalid Password,Try Again!”.
12     end
13 end
14 else
15     Display ”User Not Found”.
16 end
```

Algorithm 2: Company Registration

```
Input  : Company Name, Registration Number, Type of Company, Business Category, Address,
         Contact Details, and Verification Documents
Output: Application Number and Validation Status
1  Input Company Name, Registration Number, Type of Company, Business Category, Address and
     Contact Details.
2  Check if the registration number matches the official format using a regular expression.
3  if valid then
4      Check if contact details are valid.
5      if valid then
6          Input Certificate of Incorporation, PAN Card and Proof of Address Documents.
7          if Documents uploaded is NULL then
8              Display ”Please upload the required documents”.
9          end
10         else
11             Generate an application tracking number.
12             Display ”Validation Status: Pending Approval”.
13         end
14     end
15     else
16         Display ”Invalid Contact Details”.
17     end
18 end
19 else
20     Display ”Invalid Registration Number”.
21 end
```

Algorithm 3: Update Information

```
Input  : User Role: Admin, Company, or Delivery Personnel
         Update options: Address, Contact Information, Business Category
Output: Update Status (Success or Failure)
1  Input User Role: Admin, Company, or Delivery Personnel.
2  Verify User Login using Algorithm 1.
3  Capture the fields the user wants to update.
4  Access the corresponding database table.
5  Update the relevant fields with the new details.
6  if the database update fails then
7      Display "Error updating information. Please try again later".
8      Set Update Status as "Failure".
9      else
10         Display "Information updated successfully".
11         Set Update Status as "Success".
12     end
13 end
```

Algorithm 4: Validate Registration

```
Input  : Certificate of Incooperation, PAN Number and Proof of Address
Output: Validation status (Valid/Invalid)
1  Input Certificate of Incooperation, PAN Number and Proof of Address.
2  Ensure the Certificate of Incorporation is uploaded.
3  Verify that the PAN Card matches the company name format.
4  Check if the Proof of Address is valid.
5  if All validation is done then
6      set Validation Status as "Valid".
7      sent LoginID and temporary password to the company's email address.
8      else
9          set Validation Status as "Invalid".
10         sent a message "Validation Failed due to incorrect document submission" to the company's email
            address.
11     end
12 end
```

Algorithm 5: Scan Package

```
Input  : Package ID (from the scanned QR code)
Output: Scanning Status (Success or Failure)
1  Verify Login using Algorithm 1.
2  Extract the Package ID from the scanned data.
3  Check if the Package ID exists in the system's database.
4  if not found then
5      Display "Invalid Package ID. Please scan a valid package".
6      Set Scanning Status as "Failure".
7  end
8  else
9      Update the package record in the database
10     Set Scanning Status as "Success."
11     Display "Package successfully scanned and assigned for delivery."
12 end
```

Algorithm 6: Retrieve Optimized Route After Scanning Packages

```
Input  : Delivery Personnel ID
         Scanned Package IDs
Output:  Route Map (Optimized delivery route) or Error Message
1  Verify Login using Algorithm 1.
2  Ensure all assigned Package IDs are scanned.
3  if not all assigned packages are scanned then
4      Display "Not all assigned packages are scanned."
5  end
6  else
7      Retrieve the Optimized Route generated by the company for this specific set of packages.
8      if no route exists then
9          Display "Optimized route not available."
10     end
11     else
12         Display the Optimized Route Map on the delivery personnel's interface.
13     end
14 end
```

Algorithm 7: Route Optimization with Google Maps API

```
Input  : Delivery locations (latitude, longitude)
Output: Optimized delivery route and total distance
1  Input Delivery Locations as latitude and longitude pairs.
2  Send the locations to Google Maps Distance Matrix API.
3  Receive pairwise distances and travel times between all locations.
4  Set the first location as the starting point.
5  Initialize an empty route list and add the starting location.
6  Optimize Route Using Greedy Algorithm.
7  while there are unvisited locations do
8      Find the nearest unvisited location using the distance matrix
9      Add the nearest location to route
10     Mark the location as visited
11 end
12 Add the starting location to the end of the route to complete the cycle.
13 For calculating the total distance, sum the distances between consecutive locations in the route using the
     distance matrix.
14 Display the optimized route sequence and integrate with Google Maps for visualization.
15 Print the total distance.
```

Algorithm 8: Package Assignment Algorithm

Input : A list of packages (with attributes: destination, weight, size, priority, deadline) and a list of delivery boys (with attributes: max_weight, max_volume, max_hours).

Output: Assignment of packages to delivery boys, ensuring load balancing and prioritization.

```
1  Create a dictionary ‘destination_groups‘ to store packages grouped by their destination.
2  foreach package in the list of packages do
3      Add the package to the corresponding group in ‘destination_groups‘.
4  end
5  foreach destination in ‘destination_groups‘ do
6      Sort the packages in the group by priority (ascending) and deadline (ascending).
7  end
8  Initialize an empty list ‘sorted_packages‘.
9  foreach group in ‘destination_groups‘ do
10     Append all packages from the group to ‘sorted_packages‘.
11 end
12 Initialize delivery_boys with current_weight and current_volume set to 0.
13 foreach package in sorted_packages do
14     Sort delivery_boys by (current_weight + current_volume) in ascending order.
15     foreach boy in delivery_boys do
16         if (boy.current_weight + package.weight $\leq$ boy.max_weight) and
17         (boy.current_volume + package.size $\leq$ boy.max_volume) then
18             Append package to boy.assigned_packages.
19             Update boy.current_weight $\leftarrow$ boy.current_weight + package.weight.
20             Update boy.current_volume $\leftarrow$ boy.current_volume + package.size.
21             break /* Package assigned, move to the next package.              */
22         end
23     end
24 end
25 foreach boy in delivery_boys do
26     Print the number of packages assigned to the boy.
27     foreach package assigned to the boy do
28         Print the package’s destination, weight, and size.
29     end
30 end
```