Sri Lanka Institute of Information Technology

LinkUp

Current Trends in Software Engineering

2022

Submitted by: Palliyaguruge D.N – IT19120980

Job Application Management

Table of Contents

Contents

Background

Linkup is a social networking platform mainly focused on connecting skilled professionals with employers. In a highly competitive job market, it is essential that employers are given the ability to filter out the talent that is required and suited for the jobs. Also, the applicants must be able to showcase their knowledge and capabilities. Linkup takes the essential foundation of social networking applications and elevates it by adding extensive capabilities for job posting, application management, etc. It simplifies the process of professional networking.
In the application, there is one single user role. Unlike the competing applications where separate user roles are created for posting jobs and applying/exploring for jobs, Link up takes a different approach. In this particular application, the user has a single login and from there, they can either post jobs, edit jobs, manage the job applications that are coming, etc. (job posting functionality) while also being able to explore jobs and apply for those ones that they prefer. Based on that, the users can be divided logically as,

- Employers
- Applicants

Job application management is also another major functionality of this application. This feature allows the users to upload their job applications to the relevant job. Once the application is submitted, users will be able to view the added applications in the application screen. Users have the ability to edit and delete the applications as well. To enhance the user experience we have used some Flutter features like expansions tiles, sliders and so on. The following can be considered as the main features of the job application functionality.

- Apply for jobs
- Edit application
- Delete application

Functionalities

Apply for jobs

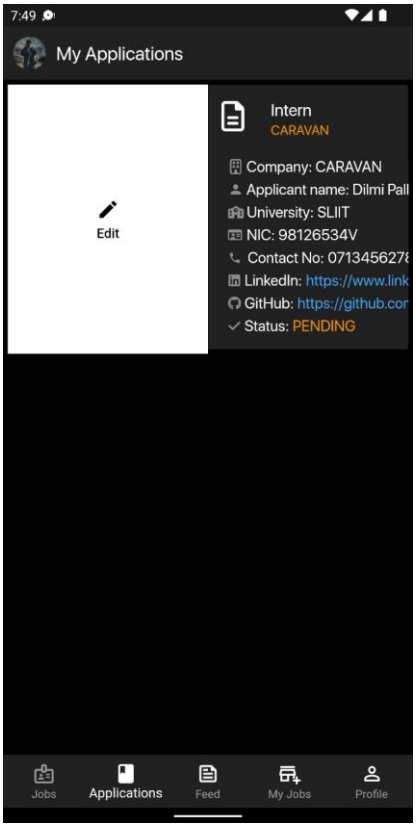
Once the user navigate to the job feed there is an option to apply for jobs. Once that button is clicked users will be redirected to the application form. The form has been properly validated and once the user has provided the relevant details he can submit the application. Once the application is added the data will be sent to the database. When viewing the application the data will be retrieved from the database. We have used expansion tiles to view applications.

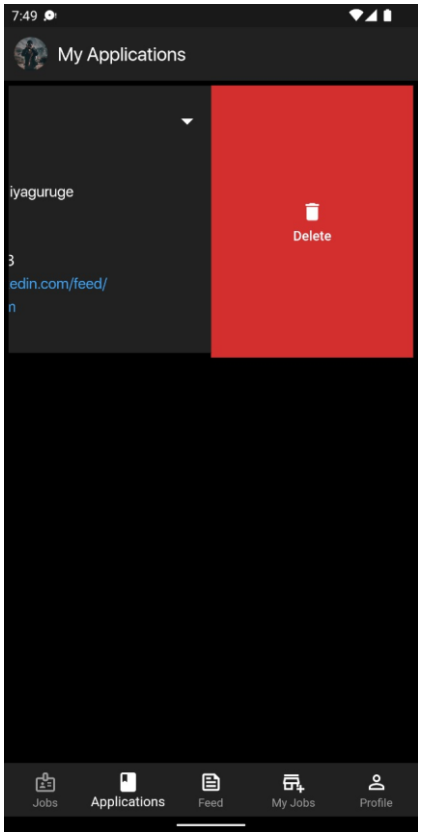
Edit applications

In the job list users can edit a specific application by swiping the application from left to right. Here we have user the ‘Slider’ feature. Once the user selects to edit the application he will be redirect back to the application form and once he submits the details the updated details will be displayed.

Delete applications

To delete the application users have to swipe the application from right to left. Here also we have used ‘Slider’ feature. Once the user deletes the application, the application will be deleted from the database and the list as well.



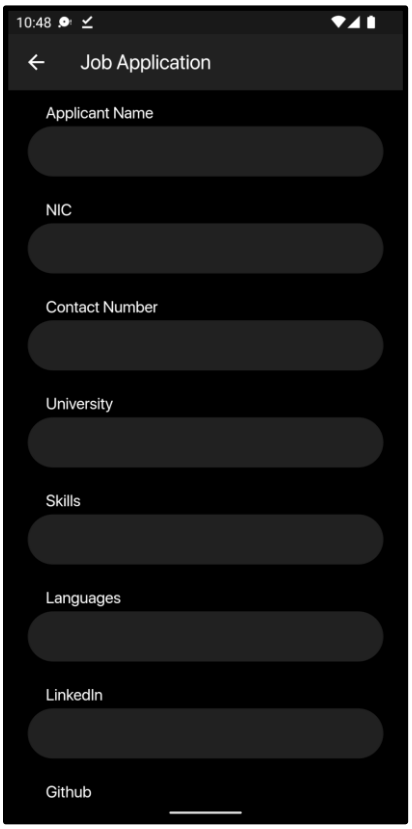


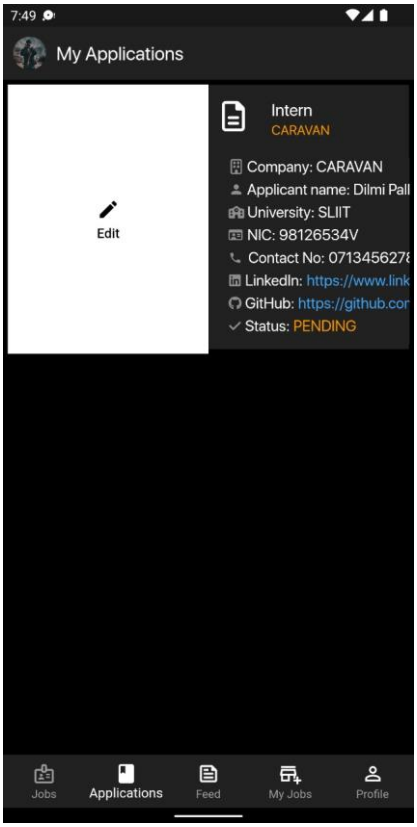
Flutter features

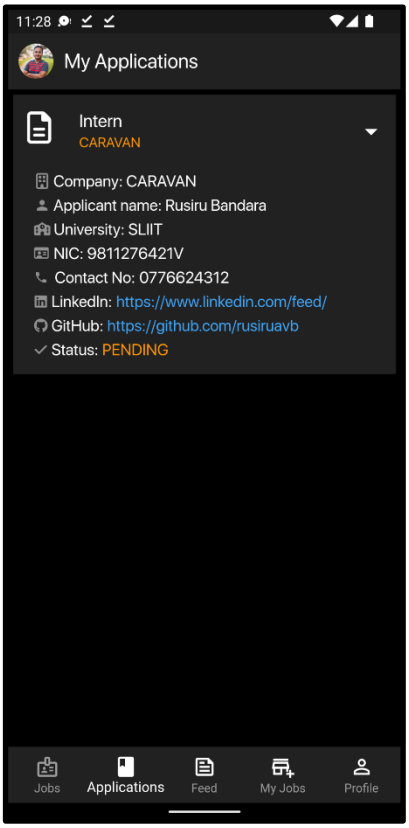
The following special features in the Flutter framework and 3rd party libraries in pub dev were used. The following list provides the references corresponding to each feature and where they were used.

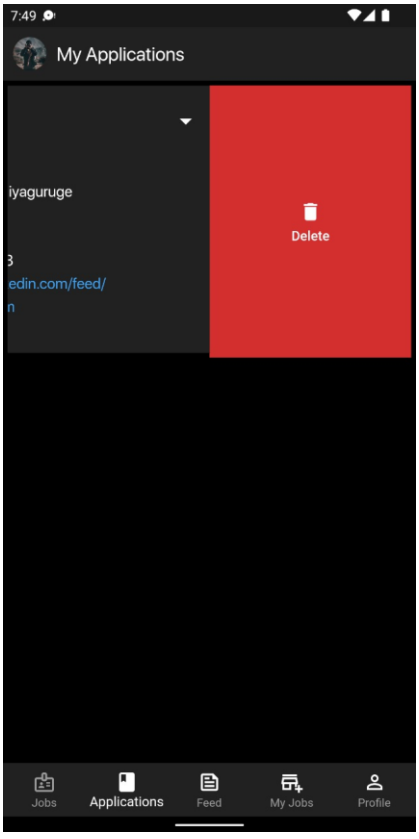
- Accordion
    - Instances used: View application
    - References :
        - https://pub.dev/packages/accordion/example
- Slider
    - Instances used : Application delete and edit
    - Reference :
        - https://api.flutter.dev/flutter/material/Slider-class.html
- Pull to refresh
    - Instances used: Profile , to refresh experiences and other modules
    - References:
        - https://pub.dev/packages/pull_to_refresh

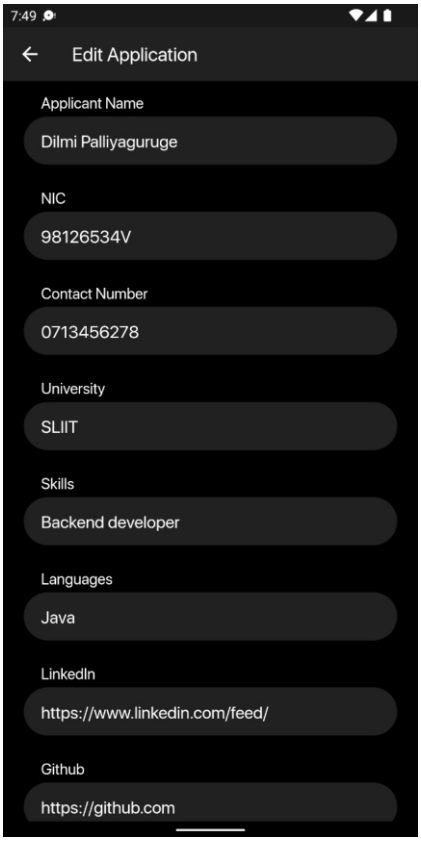
User interfaces











Screens

Application form

```
import "package:flutter/material.dart";
import 'package:fluttertoast/fluttertoast.dart';
import 'package:linkup/components/rounded_button.dart';
import 'package:linkup/components/rounded_text_field.dart';
import 'package:linkup/constants.dart';
import 'package:linkup/providers/application_provider.dart';
import 'package:provider/provider.dart';

class ApplicationFormScreen extends StatefulWidget {
  final String jobId;

  const ApplicationFormScreen({
    Key key,
    this.jobId,
  }) : super(key: key);

  @override
  _ApplicationFormScreenState createState() => _ApplicationFormScreenState();
}

class _ApplicationFormScreenState extends State<ApplicationFormScreen> {
  GlobalKey<FormState> sendApplication = GlobalKey();
  ApplicationProvider _applicationProvider;
  String applicantName = "";
  String nic = "";
  String contactNumber = "";
  String university = "";
  String skills = "";
  String languages = "";
  String linkedIn = "";
  String github = "";

  @override
  void initState() {
    super.initState();
    _applicationProvider = context.read<ApplicationProvider>();
  }

  void submit() {
```

```
  if (sendApplication.currentState.validate()) {
    _applicationProvider.createApplication(
      context,
      widget.jobId,
      applicantName,
      nic,
      contactNumber,
      university,
      skills,
      languages,
      linkedIn,
      github,
    );
  } else {
    Fluttertoast.showToast(msg: 'Please check the input fields');
  }
}

@override
Widget build(BuildContext context) {
  final size = MediaQuery.of(context).size;
  final orientation = MediaQuery.of(context).orientation;

  return Scaffold(
    appBar: AppBar(
      title: const Text(
        "Job Application",
        style: TextStyle(fontFamily: fontFamilySFPro),
      ),
      backgroundColor: colorDarkMidGround,
      elevation: 0.0,
    ),
    backgroundColor: colorDarkBackground,
    body: SingleChildScrollView(
      child: Align(
        alignment: Alignment.topCenter,
        child: Container(
          width: orientation == Orientation.landscape
              ? size.width * 0.8
              : size.width,
          padding: const EdgeInsets.all(0.0),
          child: Padding(
            padding: const EdgeInsets.all(5.0),
```

```
child: Form(
  key: sendApplication,
  child: Column(
    children: [
      SizedBox(
        height: size.height * 0.01,
      ),
      RoundedTextField(
        text: "Applicant Name",
        onChange: (value) {
          setState(() {
            applicantName = value;
          });
        },
        backgroundColor: colorDarkBackground,
        textAreaColor: colorDarkMidGround,
        value: applicantName,
        isRequired: true,
      ),
      SizedBox(
        height: size.height * 0.03,
      ),
      RoundedTextField(
        text: "NIC",
        onChange: (value) {
          setState(() {
            nic = value;
          });
        },
        backgroundColor: colorDarkBackground,
        textAreaColor: colorDarkMidGround,
        value: nic,
        isRequired: true,
      ),
      SizedBox(
        height: size.height * 0.03,
      ),
      RoundedTextField(
        text: "Contact Number",
        type: "phone",
        onChange: (value) {
          setState(() {
            contactNumber = value;
```

```
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: contactNumber,
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
RoundedTextField(
  text: "University",
  onChange: (value) {
    setState(() {
      university = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: university,
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
RoundedTextField(
  text: "Skills",
  onChange: (value) {
    setState(() {
      skills = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: skills,
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
RoundedTextField(
  text: "Languages",
  onChange: (value) {
```

```
    setState(() {
      languages = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: languages,
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
RoundedTextField(
  text: "LinkedIn",
  onChange: (value) {
    setState(() {
      linkedIn = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: linkedIn,
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
RoundedTextField(
  text: "Github",
  onChange: (value) {
    setState(() {
      github = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: github,
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
MediaQuery.of(context).orientation == Orientation.landscape
```

```
                    ? SizedBox(
                        height: size.height * 0.06,
                      )
                    : SizedBox(
                        height: size.height * 0.01,
                      ),
                Padding(
                  padding: const EdgeInsets.only(
                    left: 15,
                    right: 15,
                  ),
                  child: RoundedButton(
                    color: colorTextPrimary,
                    fontSize: 16,
                    height: 45,
                    width: size.width * 0.89,
                    text: "Apply",
                    textColor: colorDarkBackground,
                    onPressed: () {
                      submit();
                    },
                  ),
                ),
                 SizedBox(
                  height: size.height * 0.03,
                ),
              ],
            ),
          ),
        ),
      ),
    ),
  );
 }
}
```

Application screen

```
import 'package:flutter/material.dart';
import 'package:flutter_slidable/flutter_slidable.dart';
import 'package:linkup/constants.dart';
```

```dart
import 'package:linkup/models/application_model.dart';
import 'package:linkup/providers/application_provider.dart';
import 'package:linkup/screens/applications/application_tile.dart';
import 'package:linkup/screens/applications/edit_application.dart';
import 'package:provider/provider.dart';

class ApplicationScreen extends StatefulWidget {
  const ApplicationScreen({Key key}) : super(key: key);

  @override
  _ApplicationScreenState createState() => _ApplicationScreenState();
}

class _ApplicationScreenState extends State<ApplicationScreen> {
  Future<List<Application>> _applications;
  ApplicationProvider _applicationProvider;
  RouteSettings settings;
  bool _isExpanded;
  int _index;

  @override
  void initState() {
    super.initState();
    _applicationProvider = context.read<ApplicationProvider>();
    _applications = _applicationProvider.getUserApplications(context);
    _isExpanded = false;
    _index = 0;
  }

  @override
  Widget build(BuildContext context) {
    final size = MediaQuery.of(context).size;
    final orientation = MediaQuery.of(context).orientation;

    return Scaffold(
      backgroundColor: colorDarkBackground,
      body: SingleChildScrollView(
        child: Align(
          alignment: Alignment.topCenter,
          child: Container(
            width: orientation == Orientation.landscape
                ? size.width * 0.75
                : size.width,
```

```
          height: size.height,
          padding: const EdgeInsets.only(bottom: 5),
          child: Padding(
            padding: const EdgeInsets.all(5),
            child: Align(
              alignment: Alignment.topCenter,
              child: FutureBuilder<List<Application>>(
                future: _applications,
                builder: (context, snapshot) {
                  return RefreshIndicator(
                    child: _listView(snapshot),
                    onRefresh: _pullRefresh,
                  );
                },
              ),
            ),
          ),
        ),
      ),
    ),
  );
}

Widget _listView(AsyncSnapshot snapshot) {
  if (snapshot.hasData) {
    if (snapshot.data.length > 0) {
      return ListView.builder(
        itemCount: snapshot.data.length,
        itemBuilder: ((context, index) {
          return Slidable(
            key: Key(snapshot.data[index].id),
            closeOnScroll: true,
            endActionPane: ActionPane(
              dismissible: DismissiblePane(
                onDismissed: () {
                  _applicationProvider.deleteApplication(
                    context,
                    snapshot.data[index].id,
                  );
                },
              ),
              children: [
                SlidableAction(
```

```
        onPressed: (context) {
          print("Delete");
        },
        backgroundColor: colorError,
        foregroundColor: colorTextPrimary,
        spacing: 5,
        icon: Icons.delete,
        label: 'Delete',
      ),
    ],
    motion: const DrawerMotion(),
  ),
  startActionPane: ActionPane(
    motion: const DrawerMotion(),
    children: [
      SlidableAction(
        onPressed: (context) {
          Navigator.push(
            context,
            PageRouteBuilder(
              pageBuilder: (_, __, ___) => EditApplicationScreen(
                applicantName: snapshot.data[index].applicantName,
                contactNumber: snapshot.data[index].contactNumber,
                github: snapshot.data[index].github,
                linkedIn: snapshot.data[index].linkedIn,
                languages: snapshot.data[index].languages,
                nic: snapshot.data[index].nic,
                skills: snapshot.data[index].skills,
                university: snapshot.data[index].university,
                applicationId: snapshot.data[index].id,
              ),
              settings: settings,
              transitionsBuilder:
                  (context, animation, secondaryAnimation, child) =>
                      SlideTransition(
                child: child,
                position: Tween<Offset>(
                  begin: const Offset(1.0, 0.0),
                  end: Offset.zero,
                ).animate(animation),
              ),
            ),
          );
```

```
            },
            backgroundColor: colorTextPrimary,
            foregroundColor: colorDarkBackground,
            spacing: 5,
            icon: Icons.edit,
            label: 'Edit',
          ),
        ],
      ),
      child: ApplicationTile(
        index: index,
        isExpanded: _isExpanded,
        iterateIndex: index,
        title: snapshot.data[index].companyName,
        subTitle: snapshot.data[index].position,
        application: Application(
          applicantName: snapshot.data[index].applicantName,
          companyName: snapshot.data[index].companyName,
          contactNumber: snapshot.data[index].contactNumber,
          github: snapshot.data[index].github,
          linkedIn: snapshot.data[index].linkedIn,
          languages: snapshot.data[index].languages,
          nic: snapshot.data[index].nic,
          position: snapshot.data[index].position,
          skills: snapshot.data[index].skills,
          status: snapshot.data[index].status,
          university: snapshot.data[index].university,
          id: snapshot.data[index].id,
        ),
      ),
    );
  }),
 );
} else {
 return Padding(
  padding: const EdgeInsets.only(top: 15),
  child: Column(
   children: [
    Image.asset('assets/images/application.png'),
    const SizedBox(
     height: 15,
    ),
    const Text(
```

```
              'You have not apply to any job yet',
              style: TextStyle(
                fontFamily: fontFamilySFPro,
                fontSize: 18,
                color: colorTextPrimary,
              ),
            ),
          ],
        ),
      );
    }
  } else if (snapshot.hasError) {
    return const Text(
      'Error with fetch applications',
      style: TextStyle(
        fontFamily: fontFamilySFPro,
        fontSize: 16,
        color: colorErrorLight,
      ),
    );
  }
  return const Padding(
    padding: EdgeInsets.only(top: 20),
    child: SizedBox(
      width: 30,
      height: 30,
      child: CircularProgressIndicator(
        color: colorTextPrimary,
        strokeWidth: 2,
      ),
    ),
  );
 }

 Future<void> _pullRefresh() async {
  await Future.delayed(const Duration(seconds: 1));
  setState(() {
    _applications = _applicationProvider.getUserApplications(context);
  });
 }
}
```

Application tile

```
import "package:flutter/material.dart";
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'package:linkup/constants.dart';
import 'package:linkup/models/application_model.dart';

class ApplicationTile extends StatelessWidget {
  final int index;
  final int iterateIndex;
  final bool isExpanded;
  final String title;
  final String subTitle;
  final Application application;

  final Function(bool) onExpansionChanged;

  const ApplicationTile({
    Key key,
    this.application,
    this.index,
    this.iterateIndex,
    this.isExpanded,
    this.title,
    this.subTitle,
    this.onExpansionChanged,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.only(bottom: 5),
      child: ExpansionTile(
        tilePadding: const EdgeInsets.only(left: 10, right: 10),
        initiallyExpanded: false,
        collapsedBackgroundColor: colorDarkMidGround,
        backgroundColor: colorDarkMidGround,
        leading: const Icon(
          FontAwesomeIcons.fileText,
          color: colorTextPrimary,
          size: 32,
        ),
        title: Text(
          subTitle,
```

```
        style: const TextStyle(
          fontSize: 18,
          fontFamily: fontFamilySFPro,
          color: colorTextPrimary,
        ),
      ),
      subtitle: Text(
        application.companyName,
        style: TextStyle(
          fontFamily: fontFamilySFPro,
          color: application.status == "PENDING"
              ? colorWarningLight
              : application.status == "REJECTED"
                  ? colorErrorLight
                  : application.status == "SELECTED"
                      ? colorSuccessLight
                      : colorPrimaryLight,
        ),
      ),
      trailing: const Icon(
        Icons.arrow_drop_down,
        color: colorTextPrimary,
        size: 30,
      ),
      onExpansionChanged: onExpansionChanged,
      children: <Widget>[
        ApplicationCard(
          companyName: application.companyName,
          applicantName: application.applicantName,
          contactNumber: application.contactNumber,
          gitHub: application.github,
          linkedIn: application.linkedIn,
          nic: application.nic,
          university: application.university,
          status: application.status,
        )
      ],
    ),
  );
 }
}

class ApplicationCard extends StatelessWidget {
```

```
final String companyName;
final String applicantName;
final String dateTime;
final String nic;
final String contactNumber;
final String linkedIn;
final String gitHub;
final String status;
final String university;
final List<String> skills;

const ApplicationCard({
  Key key,
  this.companyName,
  this.applicantName,
  this.dateTime,
  this.nic,
  this.contactNumber,
  this.linkedIn,
  this.gitHub,
  this.status,
  this.university,
  this.skills,
}) : super(key: key);

@override
Widget build(BuildContext context) {
  return Card(
    elevation: 0.0,
    color: colorDarkMidGround,
    child: SizedBox(
      child: Align(
        alignment: Alignment.topLeft,
        child: Padding(
          padding: const EdgeInsets.only(left: 17),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Row(
                children: [
                  const Icon(
                    FontAwesomeIcons.building,
                    size: 16,
```

```
          color: colorTextDisabled,
        ),
        const Text(
          " Company: ",
          style: TextStyle(
            fontFamily: fontFamilySFPro,
            color: colorTextPrimary,
            fontSize: 16,
          ),
        ),
        Expanded(
          child: Text(
            companyName,
            style: const TextStyle(
              fontFamily: fontFamilySFPro,
              fontSize: 16,
              color: colorTextPrimary,
            ),
          ),
        ),
      ],
    ),
    const SizedBox(
      height: 5,
    ),
    Row(
      children: [
        const Icon(
          Icons.person,
          size: 16,
          color: colorTextDisabled,
        ),
        const Text(
          " Applicant name: ",
          style: TextStyle(
            fontFamily: fontFamilySFPro,
            color: colorTextPrimary,
            fontSize: 16,
          ),
        ),
        Expanded(
          child: Text(
            applicantName,
```

```
          style: const TextStyle(
            fontFamily: fontFamilySFPro,
            fontSize: 16,
            color: colorTextPrimary,
          ),
        ),
      ),
    ],
  ),
  const SizedBox(
    height: 5,
  ),
  Row(
    children: [
      const Icon(
        FontAwesomeIcons.school,
        size: 13,
        color: colorTextDisabled,
      ),
      const Text(
        "  University: ",
        style: TextStyle(
          fontFamily: fontFamilySFPro,
          color: colorTextPrimary,
          fontSize: 16,
        ),
      ),
      Expanded(
        child: Text(
          university,
          style: const TextStyle(
            fontFamily: fontFamilySFPro,
            fontSize: 16,
            color: colorTextPrimary,
          ),
        ),
      ),
    ],
  ),
  const SizedBox(
    height: 5,
  ),
  Row(
```

```
          children: [
            const Icon(
              FontAwesomeIcons.idCard,
              size: 13,
              color: colorTextDisabled,
            ),
            const Text(
              "  NIC: ",
              style: TextStyle(
                fontFamily: fontFamilySFPro,
                color: colorTextPrimary,
                fontSize: 16,
              ),
            ),
            Expanded(
              child: Text(
                nic,
                style: const TextStyle(
                  fontFamily: fontFamilySFPro,
                  fontSize: 16,
                  color: colorTextPrimary,
                ),
              ),
            ),
          ],
        ),
        const SizedBox(
          height: 5,
        ),
        Row(
          children: [
            const Icon(
              Icons.phone,
              size: 14,
              color: colorTextDisabled,
            ),
            const Text(
              "  Contact No: ",
              style: TextStyle(
                fontFamily: fontFamilySFPro,
                color: colorTextPrimary,
                fontSize: 16,
              ),
```

```
    ),
    Expanded(
      child: Text(
        contactNumber,
        style: const TextStyle(
          fontFamily: fontFamilySFPro,
          fontSize: 16,
          color: colorTextPrimary,
        ),
      ),
    ),
  ],
),
const SizedBox(
  height: 5,
),
Row(
  mainAxisAlignment: MainAxisAlignment.end,
  children: [
    const Icon(
      FontAwesomeIcons.linkedin,
      size: 14,
      color: colorTextDisabled,
    ),
    const Text(
      " LinkedIn: ",
      style: TextStyle(
        fontFamily: fontFamilySFPro,
        color: colorTextPrimary,
        fontSize: 16,
      ),
    ),
    Expanded(
      child: Text(
        linkedIn,
        style: const TextStyle(
          fontFamily: fontFamilySFPro,
          fontSize: 16,
          color: colorPrimaryLight,
        ),
      ),
    ),
  ],
```

```
),
const SizedBox(
  height: 5,
),
Row(
  mainAxisAlignment: MainAxisAlignment.end,
  children: [
    const Icon(
      FontAwesomeIcons.github,
      size: 14,
      color: colorTextDisabled,
    ),
    const Text(
      " GitHub: ",
      style: TextStyle(
        fontFamily: fontFamilySFPro,
        color: colorTextPrimary,
        fontSize: 16,
      ),
    ),
    Expanded(
      child: Text(
        gitHub,
        style: const TextStyle(
          fontFamily: fontFamilySFPro,
          fontSize: 16,
          color: colorPrimaryLight,
        ),
      ),
    ),
  ],
),
const SizedBox(
  height: 5,
),
Row(
  mainAxisAlignment: MainAxisAlignment.end,
  children: [
    const Icon(
      FontAwesomeIcons.check,
      size: 14,
      color: colorTextDisabled,
    ),
```

```
                const Text(
                  " Status: ",
                  style: TextStyle(
                    fontFamily: fontFamilySFPro,
                    color: colorTextPrimary,
                    fontSize: 16,
                  ),
                ),
                Expanded(
                  child: Text(
                    status,
                    style: TextStyle(
                      fontFamily: fontFamilySFPro,
                      fontSize: 16,
                      color: status == "PENDING"
                          ? colorWarningLight
                          : status == "REJECTED"
                              ? colorErrorLight
                              : status == "SELECTED"
                                  ? colorSuccessLight
                                  : colorPrimaryLight,
                    ),
                  ),
                ),
              ],
            ),
            const SizedBox(
              height: 10,
            )
          ],
        ),
      ),
    ),
  );
 }
}
```

Edit application

```
import 'package:flutter/material.dart';
```

```dart
import 'package:fluttertoast/fluttertoast.dart';
import 'package:linkup/components/rounded_button.dart';
import 'package:linkup/components/rounded_text_field.dart';
import 'package:linkup/constants.dart';
import 'package:linkup/providers/application_provider.dart';
import 'package:provider/provider.dart';

class EditApplicationScreen extends StatefulWidget {
  final String applicationId;
  final String applicantName;
  final String university;
  final String nic;
  final String contactNumber;
  final String skills;
  final String languages;
  final String linkedIn;
  final String github;

  const EditApplicationScreen({
    Key key,
    this.applicationId,
    this.applicantName,
    this.university,
    this.contactNumber,
    this.github,
    this.languages,
    this.linkedIn,
    this.nic,
    this.skills,
  }) : super(key: key);

  @override
  _EditApplicationState createState() => _EditApplicationState();
}

class _EditApplicationState extends State<EditApplicationScreen> {
  GlobalKey<FormState> editApplication = GlobalKey();
  ApplicationProvider _applicationProvider;
  String _applicationId = "";
  String _applicantName = "";
  String _nic = "";
  String _contactNumber = "";
  String _university = "";
```

```
String _skills = "";
String _languages = "";
String _linkedIn = "";
String _github = "";

@override
void initState() {
  super.initState();
  _applicationProvider = context.read<ApplicationProvider>();
  _applicationId = widget.applicationId;
  _applicantName = widget.applicantName;
  _contactNumber = widget.contactNumber;
  _nic = widget.nic;
  _university = widget.university;
  _skills = widget.skills;
  _languages = widget.languages;
  _linkedIn = widget.linkedIn;
  _github = widget.github;
}

void submit() {
  if (editApplication.currentState.validate()) {
    _applicationProvider.updateApplication(
      context,
      _applicationId,
      _applicantName,
      _nic,
      _contactNumber,
      _university,
      _skills,
      _languages,
      _linkedIn,
      _github,
    );
  } else {
    Fluttertoast.showToast(msg: 'Please check the input fields');
  }
}

@override
Widget build(BuildContext context) {
  final size = MediaQuery.of(context).size;
  final orientation = MediaQuery.of(context).orientation;
```

```
return Scaffold(
  appBar: AppBar(
    title: const Text(
      "Edit Application",
      style: TextStyle(fontFamily: fontFamilySFPro),
    ),
    backgroundColor: colorDarkMidGround,
    elevation: 0.0,
  ),
  backgroundColor: colorDarkBackground,
  body: SingleChildScrollView(
    child: Align(
      alignment: Alignment.topCenter,
      child: Container(
        width: orientation == Orientation.landscape
            ? size.width * 0.8
            : size.width,
        padding: const EdgeInsets.all(0.0),
        child: Padding(
          padding: const EdgeInsets.all(5.0),
          child: Form(
            key: editApplication,
            child: Column(
              children: [
                SizedBox(
                  height: size.height * 0.01,
                ),
                RoundedTextField(
                  text: "Applicant Name",
                  onChange: (value) {
                    setState(() {
                      _applicantName = value;
                    });
                  },
                  backgroundColor: colorDarkBackground,
                  textAreaColor: colorDarkMidGround,
                  value: _applicantName,
                  isRequired: true,
                ),
                SizedBox(
                  height: size.height * 0.03,
                ),
```

```
RoundedTextField(
  text: "NIC",
  onChange: (value) {
    setState(() {
      _nic = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: _nic,
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
RoundedTextField(
  text: "Contact Number",
  type: "phone",
  onChange: (value) {
    setState(() {
      _contactNumber = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: _contactNumber,
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
RoundedTextField(
  text: "University",
  onChange: (value) {
    setState(() {
      _university = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: _university,
  isRequired: true,
),
```

```
SizedBox(
  height: size.height * 0.03,
),
RoundedTextField(
  text: "Skills",
  onChange: (value) {
    setState(() {
      _skills = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: _skills,
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
RoundedTextField(
  text: "Languages",
  onChange: (value) {
    setState(() {
      _languages = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: _languages,
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
RoundedTextField(
  text: "LinkedIn",
  onChange: (value) {
    setState(() {
      _linkedIn = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: _linkedIn,
```

```
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
RoundedTextField(
  text: "Github",
  onChange: (value) {
    setState(() {
      _github = value;
    });
  },
  backgroundColor: colorDarkBackground,
  textAreaColor: colorDarkMidGround,
  value: _github,
  isRequired: true,
),
SizedBox(
  height: size.height * 0.03,
),
MediaQuery.of(context).orientation == Orientation.landscape
    ? SizedBox(
        height: size.height * 0.06,
      )
    : SizedBox(
        height: size.height * 0.01,
      ),
Padding(
  padding: const EdgeInsets.only(
    left: 15,
    right: 15,
  ),
  child: RoundedButton(
    color: colorTextPrimary,
    fontSize: 16,
    height: 45,
    width: size.width * 0.89,
    text: "Apply",
    textColor: colorDarkBackground,
    onPressed: () {
      submit();
    },
  ),
```

```
                ),
                SizedBox(
                  height: size.height * 0.03,
                ),
              ],
            ),
          ),
        ),
      ),
    ),
  );
 }
}
```

Providers

Application Provider

```
import 'dart:convert';

import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:http/http.dart' as http;
import 'package:flutter_secure_storage/flutter_secure_storage.dart';
import 'package:linkup/constants.dart';
import 'package:linkup/models/application_model.dart';

class ApplicationProvider extends ChangeNotifier {
 final storage = const FlutterSecureStorage();

 void createApplication(
   BuildContext context,
   String jobId,
   String applicantName,
   String nic,
   String phoneNumber,
   String university,
```

```
  String skills,
  String languages,
  String linkedIn,
  String gitHub,
) async {
  var userId = await storage.read(key: 'userId');
  var authToken = await storage.read(key: 'authToken');
  var response = await http.post(
    Uri.parse('$baseApi/applications/user/$userId'),
    headers: <String, String>{
      'Content-Type': 'application/json; charset=UTF-8',
      'x-auth-token': authToken,
    },
    body: jsonEncode(
      <String, String>{
        'jobId': jobId,
        'applicantName': applicantName,
        'nic': nic,
        'contactNumber': phoneNumber,
        'university': university,
        'skills': skills,
        'languages': languages,
        'linkedIn': linkedIn,
        'github': gitHub,
      },
    ),
  );

  if (response.statusCode == 200) {
    Fluttertoast.showToast(
      msg: 'Application Sent',
      backgroundColor: colorSuccess,
      textColor: colorTextPrimary,
    );
    notifyListeners();
    // Navigator.pushNamed(context, '/home');
  } else if (response.statusCode == 400) {
    Fluttertoast.showToast(msg: 'Authentication Failed');
    Navigator.pushNamed(context, '/login');
```

```
    notifyListeners();
  } else {
    Fluttertoast.showToast(msg: 'Server Error');
    notifyListeners();
  }
}

Future<List<Application>> getUserApplications(BuildContext context) async {
  List<Application> userApplications = [];
  var userId = await storage.read(key: 'userId');
  var authToken = await storage.read(key: 'authToken');
  var response = await http.get(
    Uri.parse('$baseApi/applications/user/$userId'),
    headers: <String, String>{
      'Content-Type': 'application/json; charset=UTF-8',
      'x-auth-token': authToken,
    },
  );

  if (response.statusCode == 200) {
    final applications = jsonDecode(response.body) as List;

    for (Map<String, dynamic> application in applications) {
      userApplications.add(Application.fromJson(application));
    }
    return userApplications;
  } else if (response.statusCode == 400) {
    Fluttertoast.showToast(msg: 'Authentication Failed');
    Navigator.pushNamed(context, '/login');
    notifyListeners();
    return null;
  } else {
    Fluttertoast.showToast(msg: 'Server Error');
    notifyListeners();
    return null;
  }
}

void updateApplication(
```

```
  BuildContext context,
  String applicationId,
  String applicantName,
  String nic,
  String phoneNumber,
  String university,
  String skills,
  String languages,
  String linkedIn,
  String gitHub,
) async {
  var authToken = await storage.read(key: 'authToken');
  var response = await http.put(
    Uri.parse('$baseApi/applications/edit/$applicationId'),
    headers: <String, String>{
      'Content-Type': 'application/json; charset=UTF-8',
      'x-auth-token': authToken,
    },
    body: jsonEncode(<String, String>{
      'applicantName': applicantName,
      'nic': nic,
      'contactNumber': phoneNumber,
      'university': university,
      'skills': skills,
      'languages': languages,
      'linkedIn': linkedIn,
      'github': gitHub,
    }),
  );

  if (response.statusCode == 200) {
    Navigator.pop(context);
    notifyListeners();
  } else if (response.statusCode == 400) {
    Fluttertoast.showToast(msg: 'Authentication Failed');
    Navigator.pushNamed(context, '/login');
    notifyListeners();
    return null;
  } else {
```

```
    Fluttertoast.showToast(msg: 'Server Error');
    notifyListeners();
    return null;
   }
  }

  void deleteApplication(
   BuildContext context,
   String applicationId,
  ) async {
   var userId = await storage.read(key: 'userId');
   var authToken = await storage.read(key: 'authToken');
   var response = await http.delete(
    Uri.parse('$baseApi/applications/remove/$userId/$applicationId'),
    headers: <String, String>{
     'Content-Type': 'application/json; charset=UTF-8',
     'x-auth-token': authToken,
    },
   );

   if (response.statusCode == 200) {
    getUserApplications(context);
    notifyListeners();
   } else if (response.statusCode == 400) {
    Fluttertoast.showToast(msg: 'Authentication Failed');
    Navigator.pushNamed(context, '/login');
    notifyListeners();
    return null;
   } else {
    Fluttertoast.showToast(msg: 'Server Error');
    notifyListeners();
    return null;
   }
  }
}
```

Models

Application Model

```
class Application {
  String id;
  String companyName;
  String position;
  String applicantName;
  String nic;
  String contactNumber;
  String university;
  String skills;
  String languages;
  String linkedIn;
  String github;
  String status;

  Application({
    this.id,
    this.companyName,
    this.position,
    this.applicantName,
    this.nic,
    this.contactNumber,
    this.university,
    this.skills,
    this.languages,
    this.linkedIn,
    this.github,
    this.status,
  });

  factory Application.fromJson(
    Map<String, dynamic> json,
  ) =>
      Application(
        id: json['_id'],
        companyName: json['job']['companyName'],
        position: json['job']['position'],
        applicantName: json['applicantName'],
        nic: json['nic'],
        contactNumber: json['contactNumber'],
```

```
      university: json['university'],
      skills: json['skills'],
      languages: json['languages'],
      linkedIn: json['linkedIn'],
      github: json['github'],
      status: json['status'],
    );

  Map<String, dynamic> toJson() => {
      'id': id,
      'companyName': companyName,
      'position': position,
      'applicantName': applicantName,
      'nic': nic,
      'contactNumber': contactNumber,
      'university': university,
      'skills': skills,
      'languages': languages,
      'linkedIn': linkedIn,
      'gitHub': github,
      'status': status,
    };
}
```

Components

Application card

```
import "package:flutter/material.dart";
import 'package:linkup/components/rounded_button.dart';
import 'package:linkup/constants.dart';

class ApplicationCard extends StatelessWidget {
  final String applicantName;
  final String position;
  final String profileImageURL;

  const ApplicationCard({
    Key key,
    this.applicantName,
```

```
  this.position,
  this.profileImageURL,
}) : super(key: key);

@override
Widget build(BuildContext context) {
  final size = MediaQuery.of(context).size;
  final orientation = MediaQuery.of(context).orientation;

  return Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      Container(
        width: orientation == Orientation.landscape
            ? size.width * 0.75
            : size.width,
        child: Card(
          color: colorDarkMidGround,
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Padding(
                padding: const EdgeInsets.all(8.0),
                child: Row(
                  // Header section
                children: [
                  ClipRRect(
                      borderRadius: BorderRadius.circular(100),
                      child: Image.network(
                        profileImageURL,
                        scale: 18,
                      )),
                  Padding(
                    padding: const EdgeInsets.only(left: 8),
                    child: Column(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        Text(
                          position,
                          style: const TextStyle(
                            fontFamily: fontFamilySFPro,
                            fontSize: 16,
                            color: Colors.white,
```

```
                    ),
                  ),
                  Text(
                    applicantName,
                    style: const TextStyle(
                      fontFamily: fontFamilySFPro,
                      fontWeight: FontWeight.bold,
                      fontSize: 18,
                      color: Colors.white,
                    ),
                  )
                ],
              ),
            ),
            Expanded(child: Container()),
            Padding(
              padding: const EdgeInsets.only(
                left: 15,
                right: 15,
              ),
              child: RoundedButton(
                color: colorDarkForground,
                fontSize: 13,
                height: 40,
                width: size.width * 0.2,
                text: "View",
                onPressed: () {
                  print("Button clicked");
                },
              ),
            ),
          ],
        ),
      ),
    ),
  )
 ],
 );
 }
}
```