A

PROJECT REPORT

ON

"Stugo - College Canteen orders made egy"

Batch: 2021-2025

J.C. BOSE UNIVERSITY OF SCIENCE AND TECHNOLOGY, FARIDABAD

In the partial fulfillment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

IN

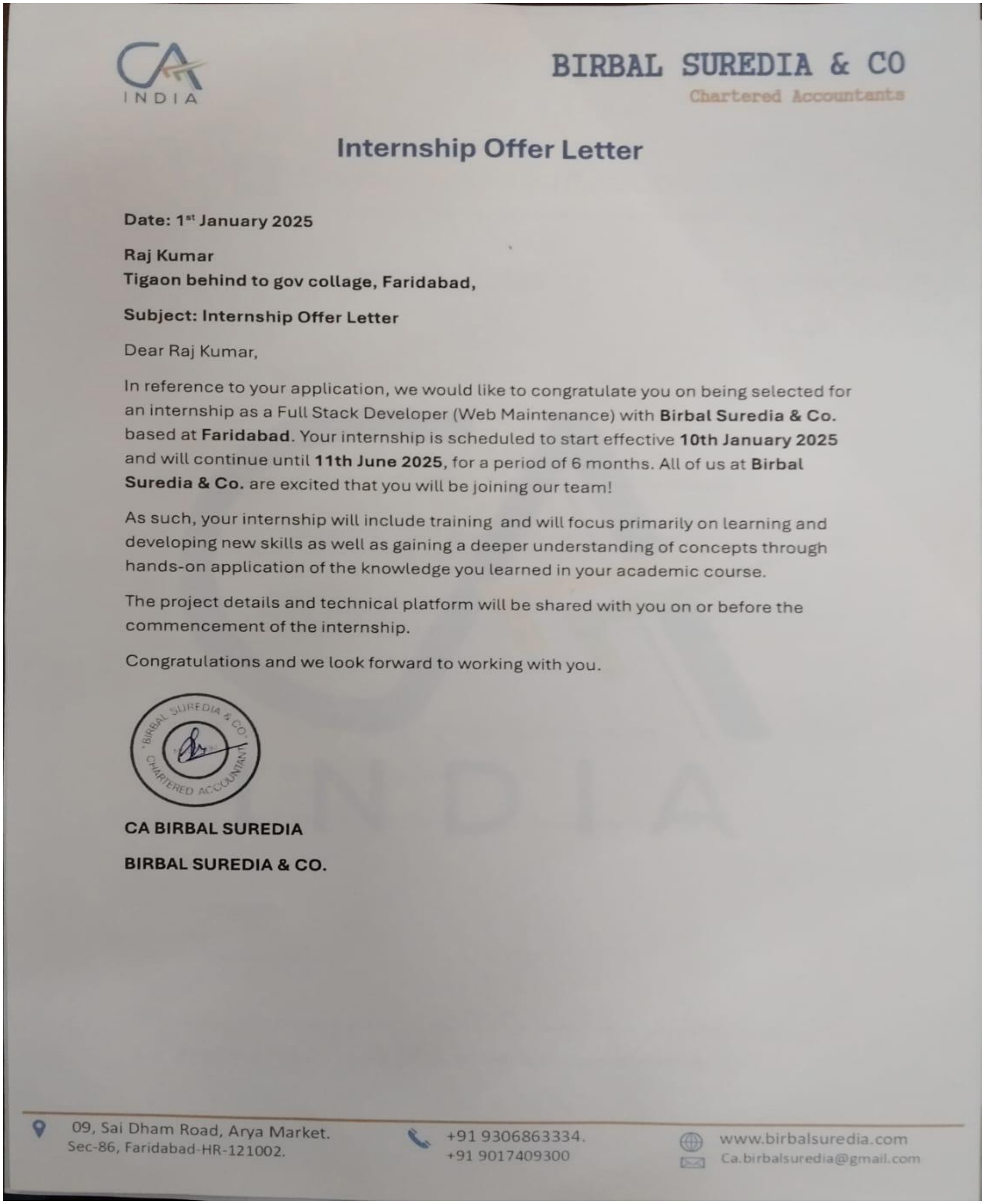
COMPUTER SCIENCE ENGINEERING

Submitted To:
Ms. Sangeeta

Submitted By:
Raj Kumar (21cse71)
University Roll no. 2101104071

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
ARAVALI COLLEGE OF ENGINEERING AND MANAGEMENT
FARIDABAD-121001

TRAINING LETTER



INDIA

BIRBAL SUREDIA & CO

Chartered Accountants

Internship Offer Letter

Date: 1st January 2025

Raj Kumar
Tigaon behind to gov collage, Faridabad,

Subject: Internship Offer Letter

Dear Raj Kumar,

In reference to your application, we would like to congratulate you on being selected for an internship as a Full Stack Developer (Web Maintenance) with **Birbal Suredia & Co.** based at **Faridabad**. Your internship is scheduled to start effective **10th January 2025** and will continue until **11th June 2025**, for a period of 6 months. All of us at **Birbal Suredia & Co.** are excited that you will be joining our team!

As such, your internship will include training and will focus primarily on learning and developing new skills as well as gaining a deeper understanding of concepts through hands-on application of the knowledge you learned in your academic course.

The project details and technical platform will be shared with you on or before the commencement of the internship.

Congratulations and we look forward to working with you.

BIRBAL SUREDIA & CO. CHARTERED ACCOUNTANT

[illegible]

CA BIRBAL SUREDIA

BIRBAL SUREDIA & CO.

09, Sai Dham Road, Arya Market.
Sec-86, Faridabad-HR-121002.

+91 9306863334.
+91 9017409300

www.birbalsuredia.com
Ca.birbalsuredia@gmail.com

CANDIDATE DECLARATION

I hereby declare that the project work entitled "**Stugo - College Canteen orders made eassy**" submitted to JCBUST, Faridabad (Haryana), is a record of an original work done by me under the guidance of **Ms. Sangeeta** (Assistant Professor) in Computer Science and Engineering, ARAVALI COLLEGE OF ENGINEERING AND MANAGEMENT, FARIDABAD,

and this project work is submitted in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in COMPUTER SCIENCE AND ENGINEERING.

Dated: **Student Name:**

Raj kumar

ACKNOWLEDGEMENT

We feel humble pleasure to express my sincere thanks and sense of gratitude to all those people who played a valuable role for the successful completion of this project by their invaluable suggestions and advice. We are very grateful to **Ms. Sakshi Kumar**, Head of the Department, (Computer Science and Engineering), whose motivation and constant encouragement has led me to pursue a project. We are very thankful to my guide (project Coordinator), and also thankful to the institute for providing this opportunity and constant encouragement given by them during the course. We are grateful to the teachers for their valuable guidance and suggestions during my project work. From the bottom of my heart, we express very thanks to my internal guide and all other professors, who are always giving motivation & suggestions to me and also giving help for the completion of my project. and without their guidance completion of the project is not possible.

Dated: **Name of the Student:**

Raj Kumar

Signature:

TABLE OF CONTENTS

COMPANY PROFILE

CA Birbal Suredia & Co. is a forward-thinking chartered accountancy firm committed to delivering comprehensive financial, compliance, and advisory services to individuals, businesses, and organizations. Established in 2023, the firm has swiftly built a reputation for providing reliable, transparent, and client-focused services in the areas of taxation, audit, and financial consulting. Based in Faridabad, CA Birbal Suredia & Co. continues to expand its reach through trusted client relationships and professional excellence.

CA Birbal Suredia & Co. offers a broad range of services, including:

● **Taxation Services:** Expert guidance and filing for Income Tax, GST (Goods and Services Tax), TDS, and other direct and indirect taxes, along with strategic tax planning and compliance support.

● **Audit & Assurance:** Statutory audits, internal audits, tax audits, and management audits conducted in accordance with applicable standards and regulations to ensure transparency and financial accuracy.

● **Accounting & Bookkeeping:** End-to-end accounting solutions, payroll management, and financial reporting tailored to the specific needs of businesses across sectors.

● **Business Advisory:** Financial planning, business setup advisory, investment consultancy, and assistance with statutory registrations such as MSME, Startup India, and FSSAI**.**

● **Company Incorporation & ROC Filing:** Services related to company formation, LLP registration, annual compliance, and other MCA (Ministry of Corporate Affairs) requirements**.**

● **GST Consulting:** Registration, return filing, reconciliation, and representation services under the Goods and Services Tax regime.

● **Startup Support & Financial Consultancy:** Assisting startups and MSMEs with financial structuring, funding documentation, and business projections.

CA Birbal Susredia & Co. strives to be a strategic financial partner for clients, offering them not just compliance but also proactive advice to help them grow responsibly and sustainably. With a focus on integrity, client education, and long-term relationships, the firm is dedicated to empowering businesses and individuals with practical solutions that align with their financial goals.

CHAPTER-1

INTRODUCTION

INTRODUCTION

In the modern college environment, food ordering systems within campuses are often fragmented, inconvenient, and outdated. Students are required to visit the canteen physically, stand in long queues, and face inconsistencies in service, while vendors struggle with unorganized order management and limited reach.

To solve this, STUGO – *College Canteen Orders Made Easy* – is a smart platform that connects students and multiple canteen vendors across various colleges based on location. It streamlines food ordering through a digital medium while providing additional features like event updates, wallet-based payments, and a "pay later" option verified via student documentation. The app bridges gaps in accessibility, communication, and management, aiming to transform the college food service experience into a smart, seamless, and scalable ecosystem.

1.2 OBJECTIVE OF THE PROJECT

The main objectives of the STUGO app are:

- To simplify and digitize food ordering within college campuses.
- To support multiple vendors and colleges within one unified platform.
- To enable students to access the canteen menu based on their college/location.
- To provide event and information updates from their college administration.
- To implement a wallet system with options like recharge, pay-later, and document-based verification.
- To explore future scalability toward partnerships with street vendors and small food businesses.

1.3 SCOPE OF THE PROJECT

The project has wide applicability across:

- Students: Can order food, pay via wallet or pay-later, and receive college updates.
- Vendors: Manage orders and menus across multiple colleges.
- College Administration: Share events and announcements.
- Future Partners: Like street vendors can join to expand reach.

1.4 SIGNIFICANCE OF THE PROJECT

The project holds significance in the following ways:

- Reduces wait time and overcrowding at college canteens.
- Enables smart management of food orders for vendors.
- Provides students with easy access to food, college events, and updates.
- Encourages digital adoption among small-scale vendors.
- Supports inclusion with the wallet and pay-later features for students in need.
- Opens potential revenue and engagement channels for colleges and vendors.

1.5 SOLUTION OF THE PROPOSED SYSTEM

The **STUGO** platform offers:

- A mobile application for students to order food, manage and view college-related information.
- A vendor dashboard for managing orders, updating menus, and analyzing sales.
- A college portal to publish announcements and connect with students.
- Wallet and Pay Later system that supports affordability for students.
- Location-based access, ensuring students see only their college vendors and menus.
- A future-ready framework for expansion to city-wide food partners, street vendors.

1.6 PROJECT FRAME WORK

The Stugo project follows a layered development approach encompassing:

- **Requirement Analysis** – Identifying user needs and healthcare data standards.
- **System Design** – Structuring modules like user profiles, access control, order management etc.
- **Development** – Backend in Node.js/Express with Supabase, frontend using Flutter for both Students and admin users.
- **Integration** – API-based communication with external systems like college details and Locations.
- **Testing** – Module-wise and integration testing, followed by usability testing.

1.7 DATA AND INFORMATION

Stugo manages the following key data:

- **Student Data**: Name, college, contact, and order history.
- **Vendor Data**: Menus, pricing, order details, and location.
- **Transaction Data**: Wallet balances, pay-later records, and payment history.
- **College Updates**: Event announcements, notices, and general communication logs.

1.8 METHODOLOGY / PROCEDURE

- **User Authentication**: Secure login via OTP, password, or biometrics.
- **Menu Access:** Students view college-specific menus.
- **Order Placement**: Orders placed via app, with wallet or pay-later options.
- **Vendor Dashboard**: Vendors manage menus and orders in real-time.
- **Notification System**: Alerts for order updates and college events.
- **Multi-College Support:** Vendors serve multiple colleges with location-based service.
- **Dashboard & Visualization**: Interactive graphs and summaries of vital health indicators.
- **Multilingual Interface**: Supports Hindi, English, and other Indian languages for accessibility.

1.9 FEASIBILITY STUDY

Technical Feasibility

The project uses open-source technologies like:

- **Backend**: Node.js, Express.js
- **Frontend**: Flutter
- **Database**: Firebase, Cloudnary, Supabase
- **Machine Learning**: Python (NLP with getting smart Discount offer)
- **Blockchain Layer**: Ethereum/Hyperledger (optional prototype level)

Economic Feasibility

The cost associated with the project is minimal as:

- All software tools are free and open-source.
- Deployment can be done using free-tier services like Render or GitHub Pages (for frontend).
- No licensing fees are required.

1.10 Operational Feasibility

- Easy-to-use interface for students and vendors
- Real-time order tracking
- Designed for scalability to local vendors and other campuses

CHAPTER-2

REQUIREMENT ANALYSIS

SYSTEM ANALYSIS

2.1 USER INTERFACE

The user interface of the Stugo system will be a responsive, **web and mobile-based platform**, designed to cater to both user & Admin. The UI will:

- Allow users to **log in securely** and manage their profiles.
- Enable students to view college-specific menus and place food orders.
- Provide **dashboards** for vendors to manage orders, menus, and sales analytics.
- Include a college update section for sharing events, announcements, and notices.
- Support multi-language UI for wider accessibility across colleges.

2.2 HARDWARE REQUIREMENTS

Component	**Specification**
Processor	Intel Core i5 or higher
RAM	8 GB or higher
Storage	500 GB SSD or higher
Network	Internet connection (minimum 2 Mbps)

2.3 SOFTWARE REQUIREMENTS

Software	**Purpose**
Operating System	Windows / Linux / Mac OS
Programming Languages	Dart,Html,css,js
Libraries/Frameworks	Express.js, Flutter
AI Libraries	spaCy, scikit-learn, TextBlob
Backend Services	Firebase, Supabase, Cloudanry
Development Tools	VS Code, Android studio
API Services	Payment,
Design Tools	Figma, Canva

2.4 TOOLS USED

- *Node.js & Express.js* – For backend development and API handling.
- *Firebase* – For NoSQL database management of user records.
- *Dart* – For frontend UI development.
- *Cloudnairy* – For Storing img and video.
- *Figma* – For UI/UX prototyping.

2.5 PROBLEM ANALYSIS

The app addresses key issues in college canteen systems:

- No centralized system for multiple vendors and colleges.
- Students struggle to access menus, offers, and event info in one place.
- Vendors lack tools to manage cross-college orders efficiently.
- Communication gaps exist between students, vendors, and college admins.
- Missed engagement opportunities like birthday discounts or event-driven offers.

2.6 FUNCTIONAL REQUIREMENTS

The system shall:

- Provide secure login for students, vendors, and admins.
- Allow students to browse menus, place orders, and pay via wallet or pay-later.
- Enable vendors to update menus, manage orders, and track sales.
- Let colleges post events and notices for their students.
- Use AI/NLP to analyze order history and generate personalized offers.
- Store and manage user data for behavior-based rewards (e.g., birthday discounts).
- Offer a searchable menu and college-based filtering system.

2.7 NON-FUNCTIONAL REQUIREMENTS

- Clean and user-friendly interface for all user types.
- Real-time updates for menu, offers, and event notifications.
- Secure storage of user and order data with access control.
- Scalability to support multiple colleges and vendors.
- Multilingual support for accessibility across regions.
- Fast performance even on lower-end devices.

CHAPTER-3

SYSTEM DESIGN

SYSTEM DESIGN

3.1 SYSTEM FUNCTIONALITY

The Stugo system is designed to manage, organize, and the system is designed to allows users:

- Connect students with food vendors specific to their college location.
- Allow students to view menus, place food orders, and pay via digital wallet or “pay later” option.
- Enable vendors to manage menus, accept/reject orders, and monitor performance.
- Let college administrators post events, announcements, or alerts to students.
- Use AI to analyze user preferences and offer discounts or meal recommendations.

3.2 USER CHARACTERISTICS

The intended users of the Stugo system include:

- **Students:** Can log in, view college-specific menus, place orders, access events, and receive offers.
- **Vendors:** Can manage multiple college menus, view sales analytics, and handle orders.
- **College Admins:** Can manage announcements and moderate communication with students.
- **System Admin:** Oversees platform performance, vendor onboarding, and dispute resolution.

3.3 SPECIFICATION OF THE PROPOSED PROJECT

- **User Module:** For student authentication, order history, and wallet management.
- **Vendor Module:** Menu uploading, order tracking, and sales dashboard.
- **Admin Module:** Manages colleges, vendors, and students.
- **Recommendation Engine**: Uses previous order data and events (like birthdays) to suggest meals or apply offers.
- **Notification System:** Sends real-time updates on orders, promotions, or events.

3.4 SYSTEM MODULES

1. Login & Registration Module

- Handles registration, login, OTP-based verification, and session management.

2. Menu & Ordering Module

- Real-time food menu display and ordering per college location.

3. Wallet/Payment Module

- Recharge wallet, pay later (with KYC), and transaction tracking.

4. Vendor Management Module

- Menu edits, order handling, and analytics dashboard.
- Secure communication via audit-logged access.

5. College Info Module

- Events, announcements, and notices per college.

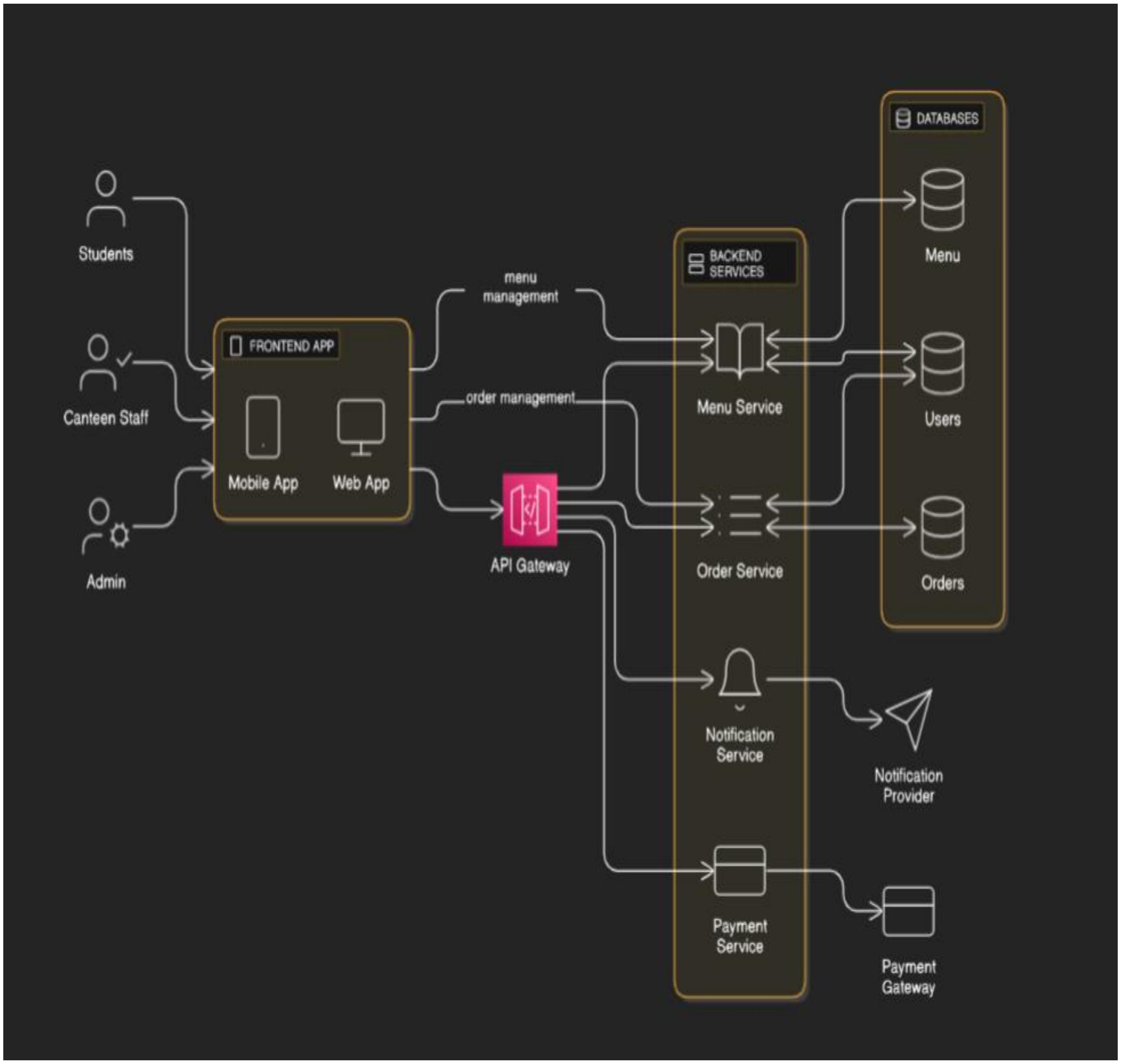
6. Analytics & Offers Module

- Tracks preferences, applies discounts, and handles birthday/seasonal offers.

7. Dashboard & Reporting Module

- Interactive dashboards trends food, Receiving orders, and history.
- Export and share records in PDF format.

3.5 ENTITY RELATION FLOW DIAGRAM



CHAPTER-4

TESTING & TEST RESULT

TESTING &TEST RESULTS

4.1 OBJECTIVE OF TESTING

The primary objective of testing is to ensure that the Stugo system performs accurately and reliably across all functionalities — including uploading documents, processing medical data, summarizing health reports, and allowing secure doctor access. Testing also verifies that the UI is responsive, data is handled securely, and features work as expected under real usage conditions.

4.2 STEPS IN TESTING

- *Unit Testing:* Verifying individual modules such as user authentication, file upload, and OCR.
- *Integration Testing:* Ensuring that modules such as the upload system, AI summarization, and dashboard correctly interact with each other.
- *System Testing:* Testing the complete end-to-end system including doctor-patient interaction, emergency access, and multi-platform deployment.

4.3 LEVELS OF TESTING

1. Unit Testing

Each functional unit like:

- Authentication module
- Medical file upload handler
- Document classification logic
- AI-based summary generator
  ...was tested independently to ensure they produce correct outputs for given inputs.

2. Integration Testing

- Testing how the system handles the flow of records from upload → storage → summary → dashboard display.

- Ensuring data passed from one module (e.g., OCR engine) integrates cleanly with downstream modules (e.g., NLP summarizer).

3. Acceptance Testing

- Conducted with end-users (patients and doctors) to verify the system's ease of use, effectiveness of medical data visualization, and emergency access reliability.
- Results were compared against predefined requirements and feedback was recorded for improvement.

4.4 TESTING STRATEGY FOR PROPOSED SYSTEM

The testing strategy adopted includes the following steps:

- *Creation of a Controlled Test Dataset:* Sample medical records (lab reports, prescriptions, scans) were collected in various formats (PDF, image, handwritten scanned copies).
- *Data Preprocessing:* Applied OCR and noise filtering to extract text, followed by structuring it for AI analysis.
- *Application of AI Summarization Module:* The processed text was passed through the health history summarizer to generate output.
- *Evaluation of Functional Results:*
    - Cross-checked extracted values (e.g., BP, sugar levels) with original data.
    - Validated summary accuracy and user access paths (doctor, patient, emergency).

4.5 TESTING ANALYSIS

The testing analysis focused on:

- Accuracy of data extraction (OCR/NLP).
- Correctness of summaries and timelines generated.
- Security of access layers and QR-based emergency view.
- Performance under multiple uploads and real-time access scenarios.
- Cross-browser/device compatibility of the user interface.

Key Observations:

- OCR accuracy exceeded 92% on typed reports and ~80% on clean handwriting.
- The system passed 95% of test cases, including role-based access and timeline generation.
- Emergency access was available within 2 seconds of QR scan.

CHAPTER-5

SYSTEM IMPLEMENTATION

SYSTEM IMPLEMENTATION

Main.dart

```
lib/
├── main.dart            # Entry point of the app
├── src/                 # All application code goes here
│   ├── common_widgets/    # Reusable UI components
│   │   ├── custom_button.dart
│   │   ├── loading_indicator.dart
│   │   └── ...
│   ├── constants/        # App-wide constants (colors, strings, etc.)
│   │   ├── app_colors.dart
│   │   ├── app_strings.dart
│   │   └── ...
│   ├── exceptions/       # Custom exception classes
│   │   ├── network_exception.dart
│   │   └── ...
│   ├── features/         # Core features of the app
│   │   ├── authentication/ # Authentication feature
│   │   │   ├── presentation/ # UI (widgets, screens)
│   │   │   │   ├── login_screen.dart
│   │   │   │   └── ...
│   │   │   ├── application/  # Business logic (use cases, services)
│   │   │   │   ├── auth_service.dart
│   │   │   │   └── ...
│   │   │   ├── domain/      # Core business logic (entities, models)
│   │   │   │   ├── user.dart
│   │   │   │   └── ...
│   │   │   └── data/        # Data access (repositories, data sources)
│   │   │       └── ...
│   │   ├── products/      # Products feature
│   │   │   ├── ...
│   │   └── ...
│   ├── localization/      # Internationalization (i18n)
│   │   ├── app_localizations.dart
│   │   └── ...
│   ├── routing/           # Navigation and routing
│   │   ├── app_router.dart
│   │   └── ...
│   ├── utils/             # Helper functions, extensions
│   │   ├── date_utils.dart
│   │   └── ...
│   └── providers/         # Providers (state management)
│       ├── auth_provider.dart
│       ├── cart_provider.dart
│       └── ...
└── services/              # Services (Firebase, API calls, etc.)
    ├── firebase_service.dart
    └── ...
```

main.dart (vendor)

```
lib/
├── main.dart
├── src/
│   ├── common_widgets/
│   │   ├── custom_button.dart
│   │   └── loading_indicator.dart
│   ├── constants/
│   │   ├── app_colors.dart
│   │   └── app_strings.dart
│   ├── exceptions/
│   │   └── network_exception.dart
│   ├── features/
│   │   ├── authentication/
│   │   │   ├── presentation/
│   │   │   │   ├── login_screen.dart
│   │   │   │   └── ...
│   │   │   ├── application/
│   │   │   │   └── auth_service.dart
│   │   │   ├── domain/
│   │   │   │   └── user.dart
│   │   │   └── data/
│   │   │       └── auth_repository.dart
│   │   ├── products/
│   │   │   ├── presentation/
│   │   │   │   └── product_screen.dart
│   │   │   ├── application/
│   │   │   │   └── product_service.dart
│   │   │   │   └── product.dart
│   │   │   └── data/
│   │   │       └── product_repository.dart
│   │   ├── profile/
│   │   │   ├── presentation/
│   │   │   │   ├── profile_screen.dart
│   │   │   │   └── edit_profile_screen.dart
│   │   │   ├── application/
│   │   │   │   └── profile_service.dart
│   │   │   ├── domain/
│   │   │   │   └── user_profile.dart
│   │   │   └── data/
│   │   │       └── profile_repository.dart
│   │   └── ...
│   ├── localization/
│   │   └── app_localizations.dart
│   ├── routing/
│   │   └── app_router.dart
│   ├── utils/
│   │   └── date_utils.dart
│   └── providers/
│       ├── auth_provider.dart
│       ├── cart_provider.dart
│       ├── product_provider.dart
│       └── profile_provider.dart
└── services/
    └── firebase_service.dart
```

SYSTEM IMPLEMENTATION CODING

MAIN.DART

```
import 'package:flutter/material.dart';
import 'package:provider/provider.dart';
import 'package:stugo/providers/settings_provider.dart';
import 'package:stugo/providers/cart_provider.dart';
import 'package:stugo/providers/food_provider.dart';
import 'package:stugo/providers/auth_provider.dart';
import 'package:stugo/providers/profile_provider.dart';
import 'package:stugo/providers/location_provider.dart';
import 'package:stugo/screens/splash_screen.dart';
import 'package:stugo/services/firebase_service.dart';
import 'package:stugo/services/college_service.dart';
import 'package:shared_preferences/shared_preferences.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();

  // Initialize SharedPreferences
  final prefs = await SharedPreferences.getInstance();

  // Initialize Firebase
  try {
    await FirebaseService.initialize();
    print('Firebase initialized successfully');
  } catch (e) {
    print('Error initializing Firebase: $e');
  }

  // Initialize CollegeService with SharedPreferences
  final collegeService = CollegeService(prefs);

  runApp(
    MultiProvider(
      providers: [
        ChangeNotifierProvider(create: (_) => AuthProvider()),
        ChangeNotifierProvider(create: (_) => SettingsProvider()),
        ChangeNotifierProvider(create: (_) => CartProvider()),
        ChangeNotifierProvider(create: (_) => FoodProvider()),
        ChangeNotifierProvider(create: (_) => ProfileProvider()),
        ChangeNotifierProvider(
          create: (context) => LocationProvider(
            collegeService,
            prefs,
            context,
          ),
        ), // <-- LocationProvider is now here
      ],
      child: Builder(
    builder: (context) => MaterialApp(
      title: 'Stugo',
      debugShowCheckedModeBanner: false,
      theme: ThemeData(
        primaryColor: const Color(0xFF3F51B5),
        colorScheme: ColorScheme.fromSeed(
          seedColor: const Color(0xFF3F51B5),
          secondary: const Color(0xFFFF9800),
          tertiary: const Color(0xFF4CAF50),
        ),
        scaffoldBackgroundColor: Colors.grey[50],
        appBarTheme: const AppBarTheme(
          backgroundColor: Colors.white,
          elevation: 0,
          iconTheme: IconThemeData(color: Colors.black87),
          titleTextStyle: TextStyle(
            color: Colors.black87,
            fontSize: 20,
            fontWeight: FontWeight.bold,
          ),
        ),
        elevatedButtonTheme: ElevatedButtonThemeData(
          style: ElevatedButton.styleFrom(
            backgroundColor: const Color(0xFF3F51B5),
            foregroundColor: Colors.white,
            padding:
            const EdgeInsets.symmetric(horizontal: 24, vertical: 12),
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(8),
            ),
          ),
        ),
        bottomNavigationBarTheme: const BottomNavigationBarThemeData(
          selectedItemColor: Color(0xFF3F51B5),
          unselectedItemColor: Colors.grey,
          type: BottomNavigationBarType.fixed,
        ),
      ),
      home: SplashScreen(), // <-- No more LocationProvider here
    ),
  ),
 ),
);
}
```

HOME.DART

```dart
import 'package:flutter/material.dart';
import 'package:provider/provider.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import '../providers/location_provider.dart';
import '../providers/profile_provider.dart';
import '../providers/cart_provider.dart';
import '../models/college.dart';
import '../models/food.dart';
import '../widgets/location_selector_sheet.dart';
import '../widgets/food_card.dart';
import '../widgets/category_card.dart';
// import 'package:google_mobile_ads/google_mobile_ads.dart';  // Temporarily removed
import '../services/ad_service.dart' as stub_ads;
import '../widgets/advertisement_banner.dart';
import '../widgets/header.dart';
import '../widgets/footer.dart';

class HomeScreen extends StatefulWidget {
  const HomeScreen({super.key});

  @override
  State<HomeScreen> createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  final PageController _pageController = PageController();
  final PageController _adBannerController = PageController();
  int _currentBannerIndex = 0;
  bool _isLoading = true;
  String? _error;
  final ScrollController _scrollController = ScrollController();
  stub_ads.BannerAd? _bannerAd;
  stub_ads.NativeAd? _nativeAd;
  bool _isLoadingMore = false;
  int _currentPage = 1;
  static const int _itemsPerPage = 10;

  @override
  void initState() {
    super.initState();
    _initializeLocation();
    // Temporarily disable ad initialization
    // _initAds();
    _scrollController.addListener(_onScroll);
  }

  Future<void> _initializeLocation() async {
    try {
      final locationProvider =
    await locationProvider.initializeLocation();
  } catch (e) {
    setState(() => _error = e.toString());
  } finally {
    setState(() => _isLoading = false);
  }
}

Future<void> _initAds() async {
  final bannerAd = stub_ads.AdService().createBannerAd();
  final nativeAd = stub_ads.AdService().createNativeAd();
  if (bannerAd != null) {
    bannerAd.load();
    setState(() => _bannerAd = bannerAd);
  }
  if (nativeAd != null) {
    nativeAd.load();
    setState(() => _nativeAd = nativeAd);
  }
}

void _onScroll() {
  if (_scrollController.position.pixels >=
          _scrollController.position.maxScrollExtent * 0.8 &&
      !_isLoadingMore) {
    _loadMoreItems();
  }
}

Future<void> _loadMoreItems() async {
  if (_isLoadingMore) return;
  setState(() => _isLoadingMore = true);
  _currentPage++;
  // Load will happen automatically through StreamBuilder
  setState(() => _isLoadingMore = false);
}

@override
void dispose() {
  _scrollController.dispose();
  _bannerAd?.dispose();
  _nativeAd?.dispose();
  super.dispose();
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: _buildAppBar(),
    body: _buildBody(),
    bottomNavigationBar: const Footer(),
  );
```

```
}

AppBar _buildAppBar() {
  return AppBar(
    title: Consumer<LocationProvider>(
      builder: (context, locationProvider, _) {
        final college = locationProvider.selectedCollege;
        return GestureDetector(
          onTap: _showLocationSelector,
          child: Container(
            padding: const EdgeInsets.symmetric(horizontal: 12, vertical: 8),
            decoration: BoxDecoration(
              color: Colors.grey[100],
              borderRadius: BorderRadius.circular(20),
            ),
            child: Row(
              mainAxisSize: MainAxisSize.min,
              children: [
                Icon(Icons.location_on,
                    color: Theme.of(context).primaryColor),
                const SizedBox(width: 8),
                Text(
                  college?.name ?? 'Select Location',
                  style: const TextStyle(fontSize: 16),
                ),
                const Icon(Icons.arrow_drop_down),
              ],
            ),
          ),
        );
      },
    ),
    actions: [
      IconButton(
        icon: const Icon(Icons.qr_code_scanner),
        onPressed: () {
          ScaffoldMessenger.of(context).showSnackBar(
            const SnackBar(
              content: Text('QR Scanner temporarily unavailable'),
              duration: Duration(seconds: 2),
            ),
          );
        },
      ),
      Consumer<CartProvider>(
        builder: (context, cart, _) => Stack(
          children: [
            IconButton(
              icon: const Icon(Icons.shopping_cart),
              onPressed: () => Navigator.pushNamed(context, '/cart'),
            ),
            if (cart.itemCount > 0)
              Positioned(
                right: 0,
                top: 0,
                child: Container(
                  padding: const EdgeInsets.all(2),
                  decoration: BoxDecoration(
                    color: Theme.of(context).primaryColor,
                    shape: BoxShape.circle,
                  ),
                  child: Text(
                    cart.itemCount.toString(),
                    style: const TextStyle(
                      color: Colors.white,
                      fontSize: 12,
                    ),
                  ),
                ),
              ),
          ],
        ),
      ),
    ],
  );
}

Widget _buildBody() {
  return Consumer<LocationProvider>(
    builder: (context, locationProvider, _) {
      if (_isLoading) {
        return const Center(child: CircularProgressIndicator());
      }

      if (_error != null) {
        return Center(
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              Text(_error!),
              ElevatedButton(
                onPressed: _initializeLocation,
                child: const Text('Retry'),
              ),
            ],
          ),
        );
      }

      final college = locationProvider.selectedCollege;
      if (college == null) {
        return _buildNoLocationSelected();
      }
      return RefreshIndicator(
        onRefresh: _initializeLocation,
        child: SingleChildScrollView(
          child: Column(
            children: [
              const Header(),
              AdvertisementBanner(
                controller: _adBannerController,
                onPageChanged: (index) =>
                    setState(() => _currentBannerIndex = index),
              ),
              _buildCategorySection(college),
              _buildPopularItems(college),
              _buildRecentItems(college),
            ],
          ),
        ),
      );
    },
  );
}

Widget _buildNoLocationSelected() {
  return Center(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        const Icon(Icons.location_off, size: 64),
        const SizedBox(height: 16),
        const Text('No location selected'),
        ElevatedButton(
          onPressed: _showLocationSelector,
          child: const Text('Select Location'),
        ),
      ],
    ),
  );
}

Widget _buildCategorySection(College college) {
  return StreamBuilder<QuerySnapshot>(
    stream: FirebaseFirestore.instance
        .collection('colleges')
        .doc(college.id)
        .collection('categories')
        .where('isActive', isEqualTo: true)
        .orderBy('order')
        .snapshots(),
    builder: (context, snapshot) {
      if (!snapshot.hasData) return const SizedBox();
      return Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          const Padding(
            padding: EdgeInsets.all(16),
            child: Text(
              'Categories',
              style: TextStyle(
                fontSize: 20,
                fontWeight: FontWeight.bold,
              ),
            ),
          ),
          SizedBox(
            height: 120,
            child: ListView.builder(
              scrollDirection: Axis.horizontal,
              padding: const EdgeInsets.symmetric(horizontal: 16),
              itemCount: categories.length,
              itemBuilder: (context, index) {
                final category =
                    categories[index].data() as Map<String, dynamic>;
                return CategoryCard(
                  name: category['name'],
                  icon: IconData(category['iconCode'],
                      fontFamily: 'MaterialIcons'),
                  onTap: () => _onCategorySelected(category['id']),
                );
              },
            ),
          ),
        ],
      );
    },
  );
}

Widget _buildPopularItems(College college) {
  return Consumer<ProfileProvider>(
    builder: (context, profile, _) {
      final isVerified =
          profile.user?.studentVerification.college == college.id &&
              profile.user?.studentVerification.status == 'verified';

      return StreamBuilder<QuerySnapshot>(
        stream: FirebaseFirestore.instance
            .collection('colleges')
            .doc(college.id)
            .collection('menu')
            .where('isActive', isEqualTo: true)
            .orderBy('orderCount', descending: true)
            .snapshots(),
        builder: (context, snapshot) {
          if (!snapshot.hasData) return const SizedBox();

          final items = snapshot.data!.docs;
          return Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Padding(
                padding: const EdgeInsets.all(16),
                child: Row(
                  mainAxisAlignment: MainAxisAlignment.spaceBetween,
                  children: [
                    const Text(
                      'Popular Items',
                      style: TextStyle(
                        fontSize: 20,
                        fontWeight: FontWeight.bold,
                      ),
                    ),
                    TextButton(
                      onPressed: () => Navigator.pushNamed(context, '/menu'),
                      child: const Text('View All'),
                    ),
                  ],
                ),
              ),
              Expanded(
                child: ListView.builder(
                  controller: _scrollController,
                  padding: const EdgeInsets.symmetric(horizontal: 16),
                  itemCount: items.length + (_isLoadingMore ? 1 : 0),
                  itemBuilder: (context, index) {
                    if (index == items.length) {
                      return const Center(
                        child: Padding(
                          padding: EdgeInsets.all(8.0),
                          child: CircularProgressIndicator(),
                        ),
                      );
                    }

                    final item = items[index].data() as Map<String, dynamic>;
                    return FoodCard(
                      foodItem: Food.fromMap(item),
                      showStudentPrice: isVerified,
                      onTap: () => _showFoodDetails(Food.fromMap(item)),
                    );
                  },
                ),
              ),
          );
        },
      );
    },
  );
}

Widget _buildRecentItems(College college) {
  return Consumer<ProfileProvider>(
    builder: (context, profile, _) {
      final isVerified =
          profile.user?.studentVerification.college == college.id &&
              profile.user?.studentVerification.status == 'verified';

      return StreamBuilder<QuerySnapshot>(
        stream: FirebaseFirestore.instance
            .collection('colleges')
            .doc(college.id)
            .collection('menu')
            .where('isActive', isEqualTo: true)
            .orderBy('createdAt', descending: true)
            .limit(10)
            .snapshots(),
        builder: (context, snapshot) {
          if (!snapshot.hasData) return const SizedBox();

          final items = snapshot.data!.docs;
          return Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Padding(
                padding: const EdgeInsets.all(16),
                child: Row(
                  mainAxisAlignment: MainAxisAlignment.spaceBetween,
                  children: [
                    const Text(
                      'Recently Added',
                      style: TextStyle(
                        fontSize: 20,
                        fontWeight: FontWeight.bold,
                      ),
                    ),
                    TextButton(
                      onPressed: () => Navigator.pushNamed(context, '/menu'),
                      child: const Text('View All'),
                    ),
                  ],
                ),
              ),
              SizedBox(
                height: 250,
                    scrollDirection: Axis.horizontal,
                    padding: const EdgeInsets.symmetric(horizontal: 16),
                    itemCount: items.length,
                    itemBuilder: (context, index) {
                      final item = items[index].data() as Map<String, dynamic>;
                      return FoodCard(
                        foodItem: Food.fromMap(item),
                        showStudentPrice: isVerified,
                        onTap: () => _showFoodDetails(Food.fromMap(item)),
                      );
                    },
                  ),
                ),
              ],
            );
          },
        );
      },
    );
  }

  void _showLocationSelector({bool showQR = false}) {
    showModalBottomSheet(
      context: context,
      isScrollControlled: true,
      builder: (context) => const LocationSelectorSheet(),
    );
  }

  void _onCategorySelected(String categoryId) {
    Navigator.pushNamed(
      context,
      '/category',
      arguments: {'categoryId': categoryId},
    );
  }

  void _showFoodDetails(Food food) {
    Navigator.pushNamed(
      context,
      '/food-details',
      arguments: {'food': food},
    );
  }
}
```

PROFILE.DART

```
import 'package:flutter/material.dart';
import 'package:stugo/utils/responsive_helper.dart';
import 'package:provider/provider.dart';
import 'package:stugo/providers/settings_provider.dart';
import 'package:stugo/providers/auth_provider.dart';
import 'package:stugo/providers/profile_provider.dart';
import 'package:stugo/providers/location_provider.dart';
import 'package:stugo/screens/edit_profile_screen.dart';
import 'package:stugo/screens/personal_info_screen.dart';
import 'package:stugo/screens/addresses_screen.dart';
import 'package:stugo/screens/payment_methods_screen.dart';
import 'package:stugo/screens/notifications_screen.dart';
import 'package:stugo/screens/language_screen.dart';
import 'package:stugo/screens/help_center_screen.dart';
import 'package:stugo/screens/feedback_screen.dart';
import 'package:stugo/screens/transaction_history_screen.dart';
import 'package:stugo/screens/student_verification_screen.dart';

enum VerificationStatus { unverified, pending, verified, rejected }

class ProfileScreen extends StatelessWidget {
  const ProfileScreen({super.key});

  @override
  Widget build(BuildContext context) {
    final settings = Provider.of<SettingsProvider>(context);
    final authProvider = Provider.of<AuthProvider>(context);
    final profileProvider = Provider.of<ProfileProvider>(context);
    final isMobile = ResponsiveHelper.isMobile(context);
    final padding = ResponsiveHelper.getPadding(context);
    final fontSize = ResponsiveHelper.getFontSize(context, 24.0);
    final iconSize = ResponsiveHelper.getIconSize(context, 24.0);

    return Scaffold(
      body: CustomScrollView(
        slivers: [
          SliverAppBar(
            expandedHeight: isMobile ? 200 : 250,
            pinned: true,
            flexibleSpace: FlexibleSpaceBar(
              background: Container(
                decoration: BoxDecoration(
                  gradient: LinearGradient(
                    begin: Alignment.topCenter,
                    end: Alignment.bottomCenter,
                    colors: [
                      Theme.of(context).primaryColor,
                      Theme.of(context).primaryColor.withOpacity(0.8),
      ),
    ),
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Stack(
          children: [
            CircleAvatar(
              radius: isMobile ? 40 : 50,
              backgroundColor: Colors.white,
              backgroundImage:
                  profileProvider.user?.profileImage != null
                      ? NetworkImage(
                          profileProvider.user!.profileImage!)
                      : null,
              child: profileProvider.user?.profileImage == null
                  ? Icon(
                      Icons.person,
                      size: isMobile ? 40 : 50,
                      color: Theme.of(context).primaryColor,
                    )
                  : null,
            ),
            if (profileProvider.user?.studentVerification.status ==
                'verified')
              Positioned(
                right: 0,
                bottom: 0,
                child: Container(
                  padding: const EdgeInsets.all(4),
                  decoration: BoxDecoration(
                    color: Colors.green,
                    shape: BoxShape.circle,
                    border:
                        Border.all(color: Colors.white, width: 2),
                  ),
                  child: const Icon(
                    Icons.check,
                    color: Colors.white,
                    size: 16,
                  ),
                ),
              ),
          ],
        ),
        const SizedBox(height: 16),
        Text(
          profileProvider.user?.name ?? 'User',
          style: TextStyle(
            fontSize: fontSize,
            color: Colors.white,
          ),
        ),
        Text(
          authProvider.currentUser?.email ?? '',
          style: TextStyle(
            fontSize: fontSize * 0.7,
            color: Colors.white.withOpacity(0.8),
          ),
        ),
        if (profileProvider.user?.studentVerification.status ==
            'verified')
          Container(
            margin: const EdgeInsets.only(top: 8),
            padding: const EdgeInsets.symmetric(
              horizontal: 12,
              vertical: 4,
            ),
            decoration: BoxDecoration(
              color: Colors.white.withOpacity(0.2),
              borderRadius: BorderRadius.circular(20),
            ),
            child: Row(
              mainAxisSize: MainAxisSize.min,
              children: [
                Icon(
                  Icons.school,
                  size: iconSize * 0.8,
                  color: Colors.white,
                ),
                const SizedBox(width: 4),
                Text(
                  'Verified Student',
                  style: TextStyle(
                    color: Colors.white,
                    fontSize: fontSize * 0.6,
                    fontWeight: FontWeight.w500,
                  ),
                ),
              ],
            ),
          ),
      ],
    ),
  ),
),
actions: [
  IconButton(
    icon: Icon(Icons.edit, size: iconSize),
    onPressed: () => Navigator.push(
      context,
      MaterialPageRoute(builder: (_) => const EditProfileScreen()),
    ),
  ],
),
SliverToBoxAdapter(
  child: Padding(
    padding: EdgeInsets.all(padding),
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        _buildCombinedWalletVerificationSection(context, fontSize),
        const SizedBox(height: 24),
        _buildSectionTitle('Account Settings', fontSize),
        _buildProfileMenuItem(
          context,
          icon: Icons.person_outline,
          title: 'Personal Information',
          subtitle: 'Update your profile details',
          onTap: () => Navigator.push(
            context,
            MaterialPageRoute(
                builder: (_) => const PersonalInfoScreen()),
          ),
        ),
        _buildProfileMenuItem(
          context,
          icon: Icons.location_on_outlined,
          title: 'Delivery Addresses',
          subtitle: 'Manage your delivery locations',
          onTap: () => Navigator.push(
            context,
            MaterialPageRoute(
                builder: (_) => const AddressesScreen()),
          ),
        ),
        _buildProfileMenuItem(
          context,
          icon: Icons.payment_outlined,
          title: 'Payment Methods',
          subtitle: 'Manage your payment options',
          onTap: () => Navigator.push(
            context,
            MaterialPageRoute(
                builder: (_) => const PaymentMethodsScreen()),
          ),
        ),
        const SizedBox(height: 24),
        _buildSectionTitle('Preferences', fontSize),
        _buildProfileMenuItem(
          context,
          icon: Icons.notifications_outlined,
          title: 'Notifications',
  onTap: () => Navigator.push(
    context,
    MaterialPageRoute(
        builder: (_) => const NotificationsScreen()),
  ),
),
_buildProfileMenuItem(
  context,
  icon: Icons.language_outlined,
  title: 'Language',
  subtitle: 'Change app language',
  onTap: () => Navigator.push(
    context,
    MaterialPageRoute(builder: (_) => const LanguageScreen()),
  ),
),
const SizedBox(height: 24),
_buildSectionTitle('Support', fontSize),
_buildProfileMenuItem(
  context,
  icon: Icons.help_outline,
  title: 'Help Center',
  subtitle: 'Get help and support',
  onTap: () => Navigator.push(
    context,
    MaterialPageRoute(
        builder: (_) => const HelpCenterScreen()),
  ),
),
_buildProfileMenuItem(
  context,
  icon: Icons.feedback_outlined,
  title: 'Feedback',
  subtitle: 'Share your thoughts with us',
  onTap: () => Navigator.push(
    context,
    MaterialPageRoute(builder: (_) => const FeedbackScreen()),
  ),
),
const SizedBox(height: 24),
_buildSectionTitle('About', fontSize),
_buildProfileMenuItem(
  context,
  icon: Icons.info_outline,
  title: 'About App',
  subtitle: 'Version 1.0.0',
  onTap: () => _showAboutDialog(context),
),
_buildProfileMenuItem(
  context,
  icon: Icons.logout,
              subtitle: 'Sign out of your account',
              onTap: () => _showLogoutDialog(context),
            ),
          ],
        ),
      ),
    ),
  ],
 ),
);
}

Widget _buildCombinedWalletVerificationSection(
  BuildContext context, double fontSize) {
 final settings = Provider.of<SettingsProvider>(context);
 final profileProvider = Provider.of<ProfileProvider>(context);
 final isVerified =
    profileProvider.user?.studentVerification.status == 'verified';
 final currency = settings.getCurrencySymbol();

 return Column(
  children: [
   AnimatedSwitcher(
    duration: const Duration(milliseconds: 300),
    child: isVerified
       ? _buildWalletSection(context, fontSize)
       : _buildVerificationSection(context, fontSize),
   ),
   const SizedBox(height: 16),
   _buildQuickActions(context, fontSize),
  ],
 );
}

Widget _buildWalletSection(BuildContext context, double fontSize) {
 final settings = Provider.of<SettingsProvider>(context);
 final profileProvider = Provider.of<ProfileProvider>(context);
 final currency = settings.getCurrencySymbol();
 final walletBalance = profileProvider.user?.wallet.balance ?? 0.0;

 return Container(
  key: const ValueKey('wallet'),
  padding: const EdgeInsets.all(16),
  decoration: BoxDecoration(
   gradient: LinearGradient(
    colors: [
     Theme.of(context).primaryColor,
     Theme.of(context).primaryColor.withOpacity(0.8),
    ],
   ),
   borderRadius: BorderRadius.circular(16),
    BoxShadow(
      color: Theme.of(context).primaryColor.withOpacity(0.3),
      blurRadius: 10,
      offset: const Offset(0, 5),
    ),
  ],
),
child: Column(
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
    Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [
        Text(
          'Student Wallet',
          style: TextStyle(
            fontSize: fontSize * 0.9,
            color: Colors.white,
            fontWeight: FontWeight.bold,
          ),
        ),
        IconButton(
          icon: const Icon(Icons.add_circle_outline, color: Colors.white),
          onPressed: () => _showAddMoneyDialog(context),
        ),
      ],
    ),
    const SizedBox(height: 16),
    Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [
        Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Text(
              'Available Balance',
              style: TextStyle(
                fontSize: fontSize * 0.7,
                color: Colors.white.withOpacity(0.8),
              ),
            ),
            const SizedBox(height: 4),
            Text(
              '$currency ${walletBalance.toStringAsFixed(2)}',
              style: TextStyle(
                fontSize: fontSize * 1.2,
                color: Colors.white,
                fontWeight: FontWeight.bold,
              ),
            ),
          ],
          ElevatedButton(
            onPressed: () => Navigator.push(
              context,
              MaterialPageRoute(
                  builder: (_) => const TransactionHistoryScreen()),
            ),
            style: ElevatedButton.styleFrom(
              backgroundColor: Colors.white,
              foregroundColor: Theme.of(context).primaryColor,
            ),
            child: const Text('View History'),
          ),
        ],
      ),
    ],
  ),
 );
}

Widget _buildVerificationSection(BuildContext context, double fontSize) {
  return Container(
    key: const ValueKey('verification'),
    padding: const EdgeInsets.all(16),
    decoration: BoxDecoration(
      color: Colors.white,
      borderRadius: BorderRadius.circular(16),
      boxShadow: [
        BoxShadow(
          color: Colors.grey.withOpacity(0.1),
          blurRadius: 10,
          offset: const Offset(0, 5),
        ),
      ],
    ),
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        Row(
          children: [
            Icon(
              Icons.verified_user,
              color: Theme.of(context).primaryColor,
              size: fontSize,
            ),
            const SizedBox(width: 8),
            Text(
              'Student Verification Required',
              style: TextStyle(
                fontSize: fontSize * 0.9,
                fontWeight: FontWeight.bold,
              ),
        ],
      ),
      const SizedBox(height: 16),
      Row(
        children: [
          Container(
            padding: const EdgeInsets.all(8),
            decoration: BoxDecoration(
              color: Colors.orange.withOpacity(0.1),
              borderRadius: BorderRadius.circular(8),
            ),
            child: const Icon(
              Icons.warning_amber_rounded,
              color: Colors.orange,
            ),
          ),
          const SizedBox(width: 16),
          Expanded(
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                Text(
                  'Wallet Access Pending',
                  style: TextStyle(
                    fontSize: fontSize * 0.8,
                    fontWeight: FontWeight.w500,
                  ),
                ),
                Text(
                  'Verify your student status to activate your wallet',
                  style: TextStyle(
                    fontSize: fontSize * 0.6,
                    color: Colors.grey[600],
                  ),
                ),
              ],
            ),
          ),
          ElevatedButton(
            onPressed: () => _showVerificationForm(context),
            style: ElevatedButton.styleFrom(
              backgroundColor: Theme.of(context).primaryColor,
            ),
            child: const Text('Verify Now'),
          ),
        ],
      ),
    ],
  ),
 );
}
```

```
void _showVerificationForm(BuildContext context) {
  Navigator.push(
    context,
    MaterialPageRoute(builder: (_) => const StudentVerificationScreen()),
  );
}

void _showAddMoneyDialog(BuildContext context) {
  final TextEditingController amountController = TextEditingController();
  final settings = Provider.of<SettingsProvider>(context, listen: false);
  final currency = settings.getCurrencySymbol();

  showDialog(
    context: context,
    builder: (context) => AlertDialog(
      title: const Text('Add Money to Wallet'),
      content: Column(
        mainAxisSize: MainAxisSize.min,
        children: [
          TextField(
            controller: amountController,
            keyboardType: TextInputType.number,
            decoration: InputDecoration(
              labelText: 'Amount',
              prefixIcon: Icon(Icons.currency_rupee),
              hintText: 'Enter amount in $currency',
            ),
          ),
          const SizedBox(height: 16),
          Row(
            mainAxisAlignment: MainAxisAlignment.spaceEvenly,
            children: [
              _buildQuickAddButton(context, '100', amountController),
              _buildQuickAddButton(context, '500', amountController),
              _buildQuickAddButton(context, '1000', amountController),
            ],
          ),
        ],
      ),
      actions: [
        TextButton(
          onPressed: () => Navigator.pop(context),
          child: const Text('Cancel'),
        ),
        ElevatedButton(
          onPressed: () {
            if (amountController.text.isNotEmpty) {
              // Handle add money logic here
              Navigator.pop(context);
              ScaffoldMessenger.of(context).showSnackBar(
                const SnackBar(
                backgroundColor: Colors.green,
              ),
            );
          }
        },
        child: const Text('Add Money'),
      ),
    ],
  ),
);
}

Widget _buildQuickAddButton(
  BuildContext context,
  String amount,
  TextEditingController controller,
) {
  return TextButton(
    onPressed: () {
      controller.text = amount;
    },
    child: Text(amount),
  );
}

void _showAboutDialog(BuildContext context) {
  showDialog(
    context: context,
    builder: (context) => AlertDialog(
      title: const Text('About Stugo'),
      content: Column(
        mainAxisSize: MainAxisSize.min,
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          _buildDetailRow('Version', '1.0.0'),
          _buildDetailRow('Build', '2024.1.1'),
          const SizedBox(height: 16),
          const Text(
            'Stugo is your campus canteen companion, making food ordering easy and convenient for
students.',
            style: TextStyle(fontSize: 14),
          ),
        ],
      ),
      actions: [
        TextButton(
          onPressed: () => Navigator.pop(context),
          child: const Text('Close'),
        ),
      ],
    ),
}

void _showLogoutDialog(BuildContext context) {
  showDialog(
    context: context,
    builder: (context) => AlertDialog(
      title: const Text('Logout'),
      content: const Text('Are you sure you want to logout?'),
      actions: [
        TextButton(
          onPressed: () => Navigator.pop(context),
          child: const Text('Cancel'),
        ),
        ElevatedButton(
          onPressed: () async {
            try {
              // Show loading indicator
              showDialog(
                context: context,
                barrierDismissible: false,
                builder: (context) => const Center(
                  child: CircularProgressIndicator(),
                ),
              );

              final authProvider =
                  Provider.of<AuthProvider>(context, listen: false);
              final profileProvider =
                  Provider.of<ProfileProvider>(context, listen: false);
              final locationProvider =
                  Provider.of<LocationProvider>(context, listen: false);

              // Sign out using AuthProvider
              await authProvider.signOut();

              // Clear other providers
              profileProvider.clearUser();
              await locationProvider.clearSelectedCollege();

              // Pop loading dialog and alert dialog
              Navigator.pop(context); // Pop loading dialog
              Navigator.pop(context); // Pop alert dialog

              // Navigate to login and remove all previous routes
              Navigator.pushNamedAndRemoveUntil(
                context,
                '/login',
                (route) => false,
              );
            } catch (e) {
              // Pop loading dialog if error occurs
              Navigator.pop(context);
              ScaffoldMessenger.of(context).showSnackBar(
                SnackBar(
                  content: Text('Error logging out: ${e.toString()}'),
                  backgroundColor: Colors.red,
                ),
              );
            }
          },
          style: ElevatedButton.styleFrom(
            backgroundColor: Colors.red,
          ),
          child: const Text('Logout'),
        ),
      ],
    ),
  );
}

Widget _buildSectionTitle(String title, double fontSize) {
  return Padding(
    padding: const EdgeInsets.symmetric(vertical: 8),
    child: Text(
      title,
      style: TextStyle(
        fontSize: fontSize * 0.8,
        fontWeight: FontWeight.bold,
      ),
    ),
  );
}

Widget _buildProfileMenuItem(
  BuildContext context, {
  required IconData icon,
  required String title,
  required String subtitle,
  required VoidCallback onTap,
}) {
  return ListTile(
    leading: Icon(icon, color: Theme.of(context).primaryColor),
    title: Text(title),
    subtitle: Text(subtitle),
    trailing: const Icon(Icons.chevron_right),
    onTap: onTap,
    shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(8)),
  );
}

Widget _buildDetailRow(String label, String value) {
  return Padding(
    padding: const EdgeInsets.symmetric(vertical: 4),
    child: Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [
        Text(
          label,
          style: const TextStyle(fontWeight: FontWeight.bold),
        ),
        Text(value),
      ],
    ),
  );
}

Widget _buildQuickActions(BuildContext context, double fontSize) {
  return Container(
    padding: const EdgeInsets.all(16),
    decoration: BoxDecoration(
      color: Colors.white,
      borderRadius: BorderRadius.circular(16),
      boxShadow: [
        BoxShadow(
          color: Colors.grey.withOpacity(0.1),
          blurRadius: 10,
          offset: const Offset(0, 5),
        ),
      ],
    ),
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        Text(
          'Quick Actions',
          style: TextStyle(
            fontSize: fontSize * 0.9,
            fontWeight: FontWeight.bold,
          ),
        ),
        const SizedBox(height: 16),
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceAround,
          children: [
            _buildQuickActionButton(
              context,
              icon: Icons.history,
              label: 'Orders',
              onTap: () => Navigator.push(
                context,
                MaterialPageRoute(
                    builder: (_) => const TransactionHistoryScreen()),
              ),
            ),
            _buildQuickActionButton(
              context,
          icon: Icons.favorite_border,
          label: 'Favorites',
          onTap: () => Navigator.push(
            context,
            MaterialPageRoute(
                builder: (_) => const TransactionHistoryScreen()),
          ),
        ),
        _buildQuickActionButton(
          context,
          icon: Icons.card_giftcard,
          label: 'Rewards',
          onTap: () => Navigator.push(
            context,
            MaterialPageRoute(
                builder: (_) => const TransactionHistoryScreen()),
          ),
        ),
        _buildQuickActionButton(
          context,
          icon: Icons.help_outline,
          label: 'Help',
          onTap: () => Navigator.push(
            context,
            MaterialPageRoute(builder: (_) => const HelpCenterScreen()),
          ),
        ),
      ],
    ),
  ],
 ),
);
}

Widget _buildQuickActionButton(
  BuildContext context, {
  required IconData icon,
  required String label,
  required VoidCallback onTap,
}) {
  return GestureDetector(
    onTap: onTap,
    child: Column(
      children: [
        Container(
          padding: const EdgeInsets.all(12),
          decoration: BoxDecoration(
            color: Theme.of(context).primaryColor.withOpacity(0.1),
            shape: BoxShape.circle,
          ),
          child: Icon(
            icon,
            color: Theme.of(context).primaryColor,
          ),
        ),
        const SizedBox(height: 8),
        Text(
          label,
          style: const TextStyle(
            fontSize: 12,
            fontWeight: FontWeight.w500,
          ),
        ),
      ],
    ),
  );
 }
}
```

ORDER_SCREEN .DART

```
import 'package:flutter/material.dart';
import 'package:audioplayers/audioplayers.dart';
import 'dart:async';
import 'package:stugo/utils/responsive_helper.dart';

class OrderSuccessScreen extends StatefulWidget {
  const OrderSuccessScreen({super.key});

  @override
  State<OrderSuccessScreen> createState() => _OrderSuccessScreenState();
}

class _OrderSuccessScreenState extends State<OrderSuccessScreen>
    with SingleTickerProviderStateMixin {
  late AnimationController _controller;
  late Animation<double> _animation;
  int _timeLeft = 180; // 3 minutes in seconds
  Timer? _timer;
  final AudioPlayer _audioPlayer = AudioPlayer();
  bool _isSoundPlaying = false;

  @override
  void initState() {
    super.initState();
    _initializeScreen();
  }

  Future<void> _initializeScreen() async {
    _startTimer();
    _controller = AnimationController(
      duration: const Duration(seconds: 1),
      vsync: this,
   );
   _animation = CurvedAnimation(
     parent: _controller,
     curve: Curves.easeInOut,
   );
   _controller.forward();

   // Initialize audio player
   await _audioPlayer.setSource(AssetSource('success.wav'));
   await _audioPlayer.setVolume(1.0);

   // Play sounds with proper error handling and source
   try {
     // First sound
     _isSoundPlaying = true;
     await _audioPlayer.resume();

     // Wait before playing second sound
     await Future.delayed(const Duration(milliseconds: 1000));

     // Second sound
     await _audioPlayer.setSource(AssetSource('notification.mp3'));
     await _audioPlayer.setVolume(1.0);
     await _audioPlayer.resume();

     debugPrint('Sounds played successfully');
   } catch (e) {
     debugPrint('Error playing sound: $e');
     if (mounted) {
       ScaffoldMessenger.of(context).showSnackBar(
         SnackBar(
           content: Text('Unable to play sound effects: ${e.toString()}'),
           backgroundColor: Colors.red,
           duration: const Duration(seconds: 3),
         ),
       );
     }
   }
 }

 @override
 void dispose() {
   _timer?.cancel();
   _controller.dispose();
   if (_isSoundPlaying) {
     _audioPlayer.stop();
   }
   _audioPlayer.dispose();
   super.dispose();
 }

 void _startTimer() {
  _timer = Timer.periodic(const Duration(seconds: 1), (timer) {
    if (_timeLeft > 0) {
      setState(() {
        _timeLeft--;
      });
    } else {
      timer.cancel();
    }
  });
}

String _formatTime(int seconds) {
  int minutes = seconds ~/ 60;
  int remainingSeconds = seconds % 60;
  return '${minutes.toString().padLeft(2, '0')}:${remainingSeconds.toString().padLeft(2, '0')}';
}

@override
Widget build(BuildContext context) {
  final isMobile = ResponsiveHelper.isMobile(context);
  final isTablet = ResponsiveHelper.isTablet(context);
  final padding = ResponsiveHelper.getPadding(context);
  final fontSize = ResponsiveHelper.getFontSize(context, 24.0);
  final iconSize = ResponsiveHelper.getIconSize(context, 50.0);

  return Scaffold(
    body: SafeArea(
      child: Column(
        children: [
          Expanded(
            child: SingleChildScrollView(
              child: Column(
                children: [
                  SizedBox(height: isMobile ? 20 : 40),
                  // Success Animation and Timer
                  Stack(
                    alignment: Alignment.center,
                    children: [
                      SizedBox(
                        width: isMobile ? 150 : (isTablet ? 180 : 200),
                        height: isMobile ? 150 : (isTablet ? 180 : 200),
                        child: CircularProgressIndicator(
                          value: _timeLeft / 180,
                          strokeWidth: isMobile ? 6 : 8,
                          backgroundColor: Colors.grey[200],
                          color: const Color(0xFF3F51B5),
                        ),
                      ),
                      Column(
                        mainAxisSize: MainAxisSize.min,
                        children: [
                          ScaleTransition(
          scale: _animation,
          child: Container(
            padding: EdgeInsets.all(isMobile ? 12 : 16),
            decoration: BoxDecoration(
              color: const Color(0xFF3F51B5),
              shape: BoxShape.circle,
              boxShadow: [
                BoxShadow(
                  color: const Color(0xFF3F51B5)
                      .withOpacity(0.3),
                  blurRadius: 20,
                  spreadRadius: 5,
                ),
              ],
            ),
            child: Icon(
              Icons.check,
              color: Colors.white,
              size: iconSize,
            ),
          ),
        ),
        const SizedBox(height: 20),
        Text(
          _formatTime(_timeLeft),
          style: TextStyle(
            fontSize: fontSize,
            fontWeight: FontWeight.bold,
            color: const Color(0xFF3F51B5),
          ),
        ),
      ],
    ),
  ],
),
SizedBox(height: isMobile ? 20 : 40),
// Order Success Message
Text(
  'Order Placed Successfully!',
  style: TextStyle(
    fontSize: fontSize,
    fontWeight: FontWeight.bold,
  ),
),
const SizedBox(height: 12),
Text(
  'Your order will be ready in ${_formatTime(_timeLeft)}',
  style: TextStyle(
    fontSize: fontSize * 0.7,
    color: Colors.grey[600],
  ),
),
SizedBox(height: isMobile ? 20 : 40),
// Order Details Card
Container(
  margin:
      EdgeInsets.symmetric(horizontal: isMobile ? 16 : 24),
  padding: EdgeInsets.all(isMobile ? 16 : 20),
  decoration: BoxDecoration(
    color: Colors.white,
    borderRadius: BorderRadius.circular(16),
    boxShadow: [
      BoxShadow(
        color: Colors.grey.withOpacity(0.1),
        blurRadius: 10,
        spreadRadius: 5,
      ),
    ],
  ),
  child: Column(
    children: [
      _buildOrderDetailRow('Order ID', '#123456', isMobile),
      const Divider(height: 20),
      _buildOrderDetailRow('Items', '3 items', isMobile),
      const Divider(height: 20),
      _buildOrderDetailRow(
          'Total Amount', '₹190', isMobile),
      const Divider(height: 20),
      _buildOrderDetailRow(
          'Payment Method', 'Pay at Counter', isMobile),
    ],
  ),
),
SizedBox(height: isMobile ? 20 : 40),
// Advertisement Section
Container(
  margin:
      EdgeInsets.symmetric(horizontal: isMobile ? 16 : 24),
  decoration: BoxDecoration(
    gradient: const LinearGradient(
      begin: Alignment.topLeft,
      end: Alignment.bottomRight,
      colors: [
        Color(0xFF6B1AB1),
        Color(0xFF4527A0),
      ],
    ),
    borderRadius: BorderRadius.circular(20),
    boxShadow: [
      BoxShadow(
        color: const Color(0xFF6B1AB1).withOpacity(0.3),
        blurRadius: 15,
        spreadRadius: 2,
      ),
    ],
  ),
  child: Stack(
    children: [
      // Background Design Elements
      Positioned(
        right: -30,
        top: -30,
        child: Container(
          width: isMobile ? 80 : 100,
          height: isMobile ? 80 : 100,
          decoration: BoxDecoration(
            shape: BoxShape.circle,
            color: Colors.white.withOpacity(0.1),
          ),
        ),
      ),
      Positioned(
        left: -20,
        bottom: -20,
        child: Container(
          width: isMobile ? 60 : 80,
          height: isMobile ? 60 : 80,
          decoration: BoxDecoration(
            shape: BoxShape.circle,
            color: Colors.white.withOpacity(0.1),
          ),
        ),
      ),
      // Content
      Padding(
        padding: EdgeInsets.all(isMobile ? 16 : 20),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Row(
              children: [
                Container(
                  padding: const EdgeInsets.symmetric(
                    horizontal: 12,
                    vertical: 6,
                  ),
                  decoration: BoxDecoration(
                    color: Colors.white,
                    borderRadius: BorderRadius.circular(20),
                    boxShadow: [
                      BoxShadow(
                        color:
                            Colors.black.withOpacity(0.1),
                        blurRadius: 10,
                      ),
                    ],
        ),
        child: Row(
          children: [
            Icon(
              Icons.local_offer,
              color:
                  Theme.of(context).primaryColor,
              size: isMobile ? 14 : 16,
            ),
            const SizedBox(width: 4),
            Text(
              'LIMITED TIME',
              style: TextStyle(
                color: const Color(0xFF6B1AB1),
                fontWeight: FontWeight.bold,
                fontSize: isMobile ? 10 : 12,
              ),
            ),
          ],
        ),
      ),
    ],
  ),
  const SizedBox(height: 16),
  Row(
    children: [
      Text(
        '20% ',
        style: TextStyle(
          color: Colors.white,
          fontSize: isMobile ? 32 : 40,
          fontWeight: FontWeight.bold,
        ),
      ),
      Text(
        'OFF',
        style: TextStyle(
          color: Colors.white,
          fontSize: isMobile ? 32 : 40,
          fontWeight: FontWeight.bold,
        ),
      ),
    ],
  ),
  Text(
    'on your next order!',
    style: TextStyle(
      color: Colors.white70,
      fontSize: isMobile ? 14 : 16,
    ),
  ),
  Row(
    mainAxisAlignment:
        MainAxisAlignment.spaceBetween,
    children: [
      Column(
        crossAxisAlignment:
            CrossAxisAlignment.start,
        children: [
          Text(
            'Use code',
            style: TextStyle(
              color: Colors.white60,
              fontSize: isMobile ? 10 : 12,
            ),
          ),
          const SizedBox(height: 4),
          Container(
            padding: const EdgeInsets.symmetric(
              horizontal: 16,
              vertical: 8,
            ),
            decoration: BoxDecoration(
              color: Colors.white,
              borderRadius:
                  BorderRadius.circular(8),
              boxShadow: [
                BoxShadow(
                  color: Colors.black
                      .withOpacity(0.1),
                  blurRadius: 10,
                ),
              ],
            ),
            child: Text(
              'NEXT20',
              style: TextStyle(
                color: const Color(0xFF6B1AB1),
                fontWeight: FontWeight.bold,
                fontSize: isMobile ? 16 : 18,
                letterSpacing: 2,
              ),
            ),
          ),
        ],
      ),
      Container(
        decoration: BoxDecoration(
          color: Colors.white,
          shape: BoxShape.circle,
          boxShadow: [
            BoxShadow(
              color:
```

```
                                    Colors.black.withOpacity(0.1),
                                blurRadius: 10,
                              ),
                            ],
                          ),
                          child: IconButton(
                            onPressed: () {
                              ScaffoldMessenger.of(context)
                                  .showSnackBar(
                                const SnackBar(
                                  content:
                                      Text('Promo code copied!'),
                                  duration: Duration(seconds: 2),
                                ),
                              );
                            },
                            icon: Icon(
                              Icons.copy,
                              color: const Color(0xFF6B1AB1),
                              size: isMobile ? 20 : 24,
                            ),
                          ),
                        ),
                      ],
                    ),
                  ],
                ),
              ),
            ],
          ),
        ),
      ],
    ),
  ),
),
// Bottom Buttons
Padding(
  padding: EdgeInsets.all(isMobile ? 16 : 24),
  child: Row(
    children: [
      Expanded(
        child: OutlinedButton(
          onPressed: () {
            Navigator.of(context)
                .popUntil((route) => route.isFirst);
          },
          style: OutlinedButton.styleFrom(
            padding: EdgeInsets.symmetric(
              vertical: isMobile ? 12 : 16,
            ),
            side: const BorderSide(color: Color(0xFF3F51B5)),
                    borderRadius: BorderRadius.circular(12),
                  ),
                ),
                child: Text(
                  'Back to Home',
                  style: TextStyle(
                    fontSize: isMobile ? 14 : 16,
                  ),
                ),
              ),
            ),
            SizedBox(width: isMobile ? 12 : 16),
            Expanded(
              child: ElevatedButton(
                onPressed: () {
                  // Navigate to track order screen
                },
                style: ElevatedButton.styleFrom(
                  padding: EdgeInsets.symmetric(
                    vertical: isMobile ? 12 : 16,
                  ),
                  backgroundColor: const Color(0xFF3F51B5),
                  shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(12),
                  ),
                ),
                child: Text(
                  'Track Order',
                  style: TextStyle(
                    fontSize: isMobile ? 14 : 16,
                  ),
                ),
              ),
            ),
          ],
        ),
      ),
    ],
  ),
),
);
}

Widget _buildOrderDetailRow(String label, String value, bool isMobile) {
  return Row(
    mainAxisAlignment: MainAxisAlignment.spaceBetween,
    children: [
      Text(
        label,
        style: TextStyle(
          color: Colors.grey[600],
          ),
        ),
        Text(
          value,
          style: TextStyle(
            fontWeight: FontWeight.bold,
            fontSize: isMobile ? 12 : 14,
          ),
        ),
      ],
    );
  }
}
```

MORE 56 FILE CODE ETC……………………………………………………………………….

CHAPTER-6

SUMMARY

SUMMARY

6.0 SUMMARY

This project introduces a smart digital solution for college canteens, connecting students, vendors, and college administrations on a unified platform. The application enables students to browse menus, place orders, and access canteen services based on their college location. Vendors can manage orders, update menus, and serve multiple colleges efficiently. The system also integrates a wallet feature, pay-later options, and real-time event updates shared by the college.

With an intuitive interface, the app streamlines the food ordering process and enhances student engagement through personalized offers based on order history using AI algorithms. The platform's scalability allows future expansion to local street vendors and food service partners, making it a promising foundation for a larger food service ecosystem

6.1 ADVANTAGES OF THE SYSTEM

- Digital and paperless food ordering and management.
- Wallet integration and pay-later option for flexibility.
- Location-based access for students to their respective college canteens.
- Easy vendor management with a multi-college dashboard.
- Personalized offers using AI and purchase history analysis.
- Real-time updates about college events and notices.
- Scalable platform for local street vendors and food service partnerships.

6.2 LIMITATIONS OF THE SYSTEM

- Requires stable internet access for real-time functionalities.
- Vendor response time may affect user satisfaction.
- NLP and recommendation accuracy depends on quality and quantity of user data.
- Limited language support in the current version (future scope for multilingual features).
- Initial adoption might be slow without effective onboarding and marketing.

6.3 GOALS OF THE SYSTEM

- Streamline the food ordering experience for students across multiple colleges.
- Enable vendors to efficiently manage operations across various institutions.
- Use AI for personalized discounts, offers, and user engagement.
- Provide colleges with a digital notice board and event announcement system.
- Expand the service to support local street food vendors and broader communities.

6.4 PROJECT LEGACY

This college canteen application establishes a scalable and flexible platform for digitizing student food services. Beyond its current scope, it can evolve into a comprehensive student lifestyle ecosystem with features such as:

- Personalized AI-driven food recommendations and health-based menus
- Integration with student ID cards and campus access systems
- Expansion to local food vendors and street sellers
- In-app event ticketing and merchandise sales
- Advanced analytics for vendors and colleges to improve service delivery

CHAPTER-7

USER MANUAL

USER MANUAL

HOW TO USE STUGO

1 Register/Login

- Open the Stugo web application.
- Click on Register (new users) or Login.
- Provide email/phone number and verify via OTP.

2 Upload Medical Records

- After login, navigate to the Upload section.
- Choose a file (PDF/image) from your device.
- Click Upload. The system will process and store the file.

3 View Health Summary

- Go to the Timeline/Dashboard tab.
- View AI-generated summaries organized by date, type, and medical parameter.
- Click on a report to see extracted values and doctor notes.

4 Share with Doctor

- Go to My Doctors section.
- Add a doctor using their email or ID.
- Grant them access to selected reports or your entire timeline.

5 Emergency QR Access

- Go to Emergency Settings.
- Click Generate QR Code.
- Print or save the QR. When scanned, it provides temporary read-only access to vital health records.

SECURITY INSTRUCTIONS

- Keep your login credentials secure.
- Revoke doctor access if no longer needed.
- Regularly update your emergency QR if there are changes in your health data.

TROUBLESHOOTING

Issue	Solution
Upload fails	Check file format (PDF/Image), refresh browser
Summarization seems incorrect	Ensure uploaded text is clear and typed
Doctor can’t view records	Re-check sharing permissions
QR doesn’t scan	Ensure good lighting and camera focus

CHAPTER-8

CONCLUSION

CONCLUSION

The project *"Stugo – College canteen orders made easy"* has been successfully designed and developed to streamline the food ordering experience for students across multiple colleges. By integrating real-time menus, wallet payments, event updates, and vendor management into a single platform, it offers a modern and accessible solution for both students and canteen vendors.

The app's scalable design, clean interface, and future-ready features like personalized offers and AI-driven recommendations make it well-suited for broader implementation. With plans to expand to local food vendors and enhance its ecosystem with smart analytics and social connectivity, *Stugo* has strong potential to grow into a comprehensive student utility platform.

This project serves not only as a technical solution but also as a startup prototype — providing valuable lessons in technology, user behavior, and market potential. Its modular and scalable architecture ensures adaptability for future enhancements, including AI offers engine, health-friendly food suggestions, multilingual support, and partner dashboards.

CHAPTER-9

FUTURE SCOPE

FUTURE SCOPE

While the current version of **Stugo** is designed to simplify canteen ordering within college campuses, future expansion aims to extend the platform beyond colleges to include **local street vendors and food service providers**..

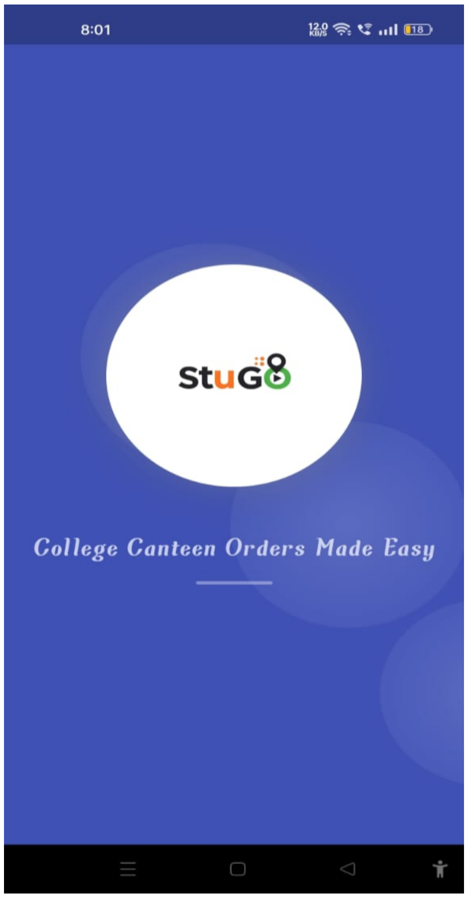
Planned future enhancements include:

- **Expansion to Public Marketplaces**: Allowing local food vendors and street stalls to join the platform and serve customers beyond just students.
- **Geo-based Vendor Discovery**: Users will be able to discover nearby registered vendors through location-based recommendations.
- **Personalized Discounts and Offers**: AI will analyze individual user behavior, ordering patterns, and birthdays to suggest customized offers and loyalty rewards.
- **Unified Wallet System**: A digital wallet usable across both college and public vendors with features like top-ups, cashback, and spending limits.
- **Business Dashboard for Vendors**: A scalable dashboard for all vendors to manage menus, track orders, and analyze customer data across different locations.
- **Offline and Low-Bandwidth Support**: Features like order queuing, SMS-based updates, and lightweight UI for vendors in areas with limited internet access.

This future vision positions Stugo as a **versatile and inclusive digital ordering platform** connecting students, local food vendors, and the broader community.

CHAPTER-10

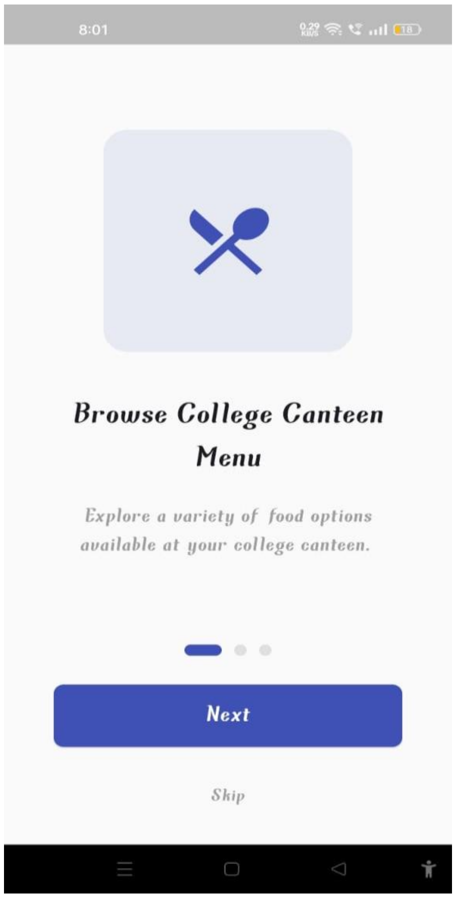
SCREENSHOTS



8:01

StuG8

College Canteen Orders Made Easy



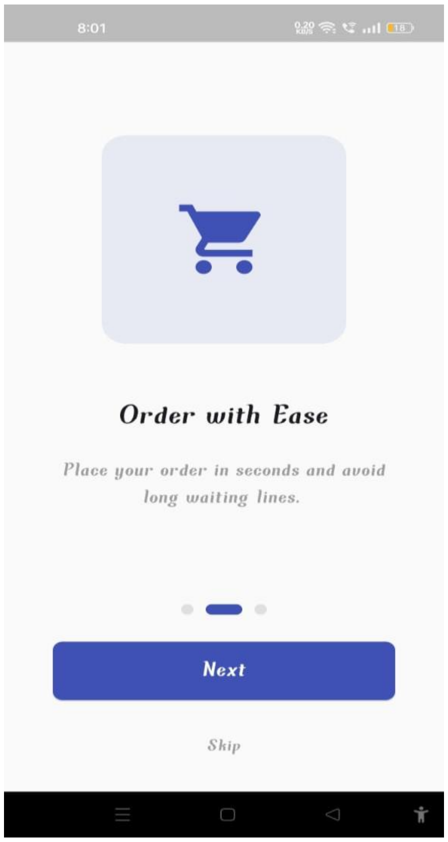
8:01

Browse College Canteen Menu

Explore a variety of food options available at your college canteen.

Next

Skip



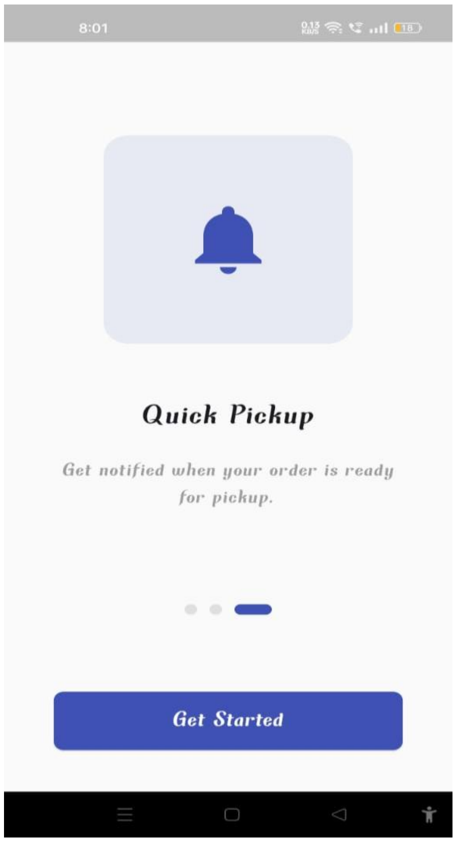
8:01

Order with Ease

Place your order in seconds and avoid long waiting lines.

Next

Skip

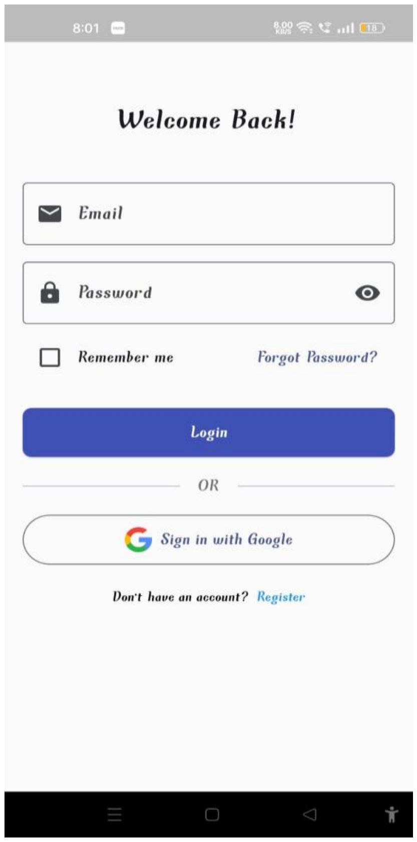


8:01

Quick Pickup

Get notified when your order is ready for pickup.

Get Started



8:01

Welcome Back!

Email

Password

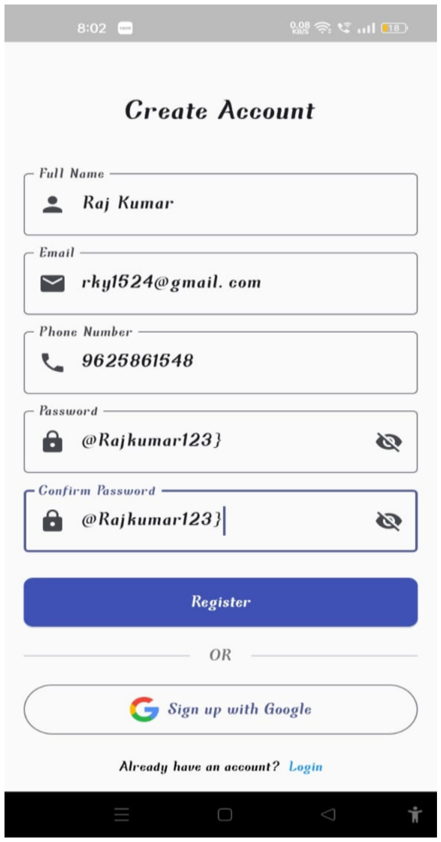
Remember me Forgot Password?

Login

OR

Sign in with Google

Don't have an account? Register



8:02

Create Account

Full Name: Raj Kumar

Email: rky1524@gmail. com

Phone Number: 9625861548

Password: @Rajkumar123}

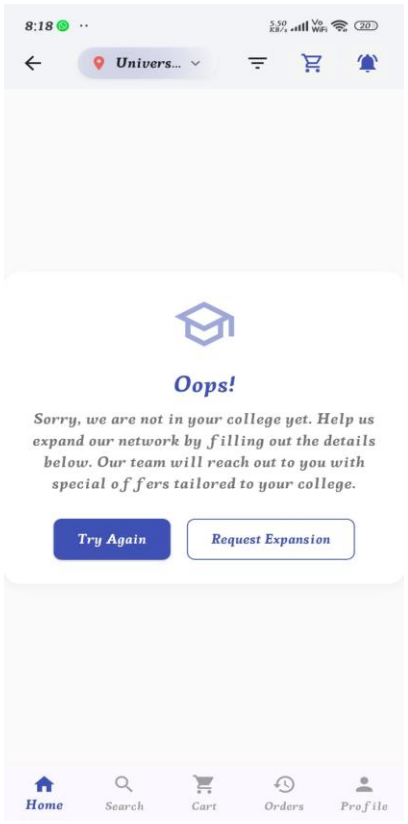
Confirm Password: @Rajkumar123}

Register

OR

Sign up with Google

Already have an account? Login



8:18

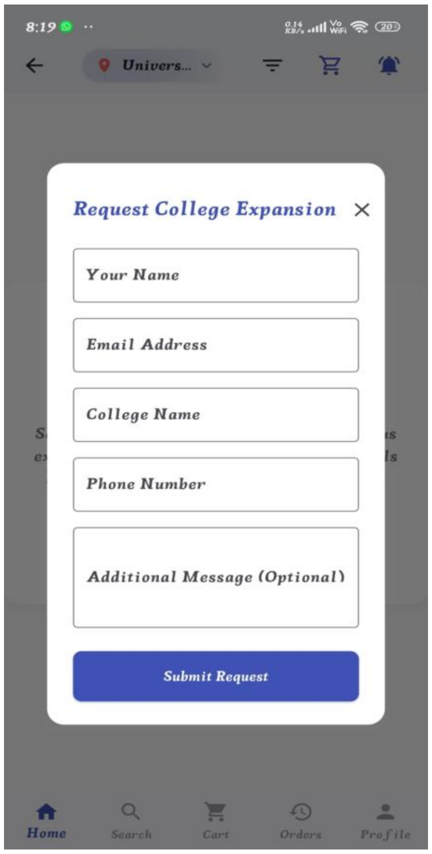
Univers...

Oops!

Sorry, we are not in your college yet. Help us expand our network by filling out the details below. Our team will reach out to you with special offers tailored to your college.

Try Again | Request Expansion

Home Search Cart Orders Profile



8:19

Univers...

Request College Expansion

Your Name

Email Address

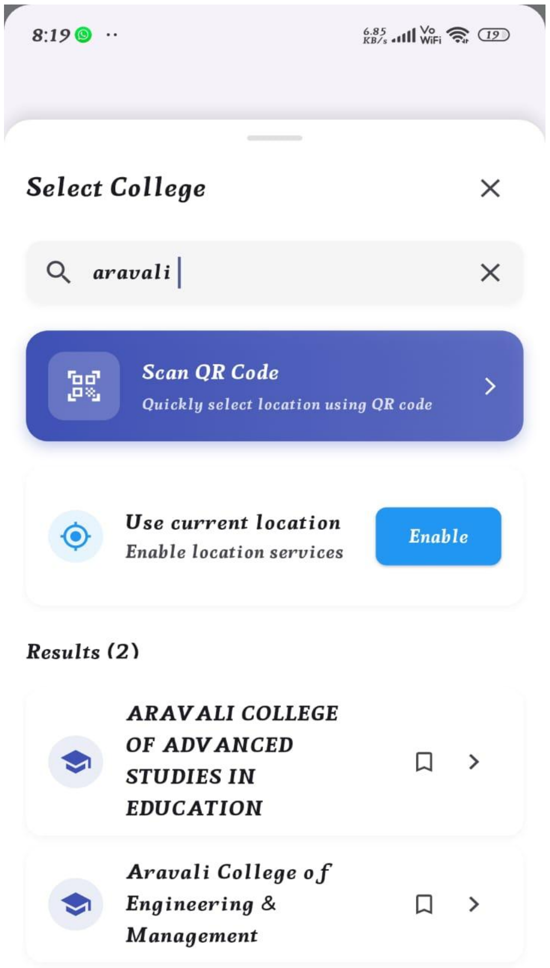
College Name

Phone Number

Additional Message (Optional)

Submit Request

Home Search Cart Orders Profile



8:19

Select College

aravali

Scan QR Code

Quickly select location using QR code

Use current location

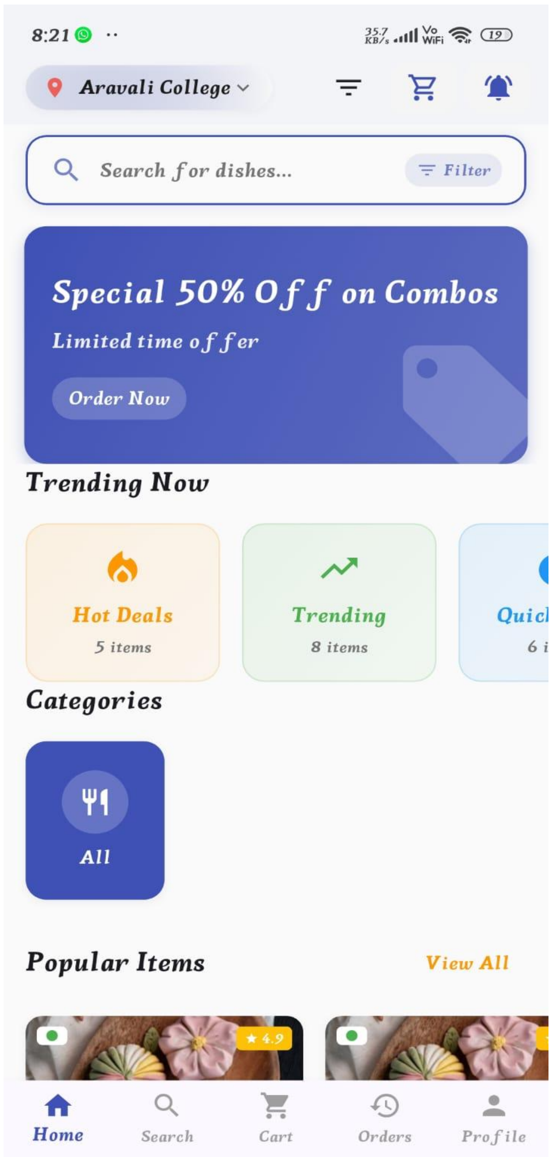
Enable location services

Enable

Results (2)

ARAVALI COLLEGE OF ADVANCED STUDIES IN EDUCATION

Aravali College of Engineering & Management



8:21

Aravali College

Search for dishes...

Filter

Special 50% Off on Combos

Limited time offer

Order Now

Trending Now

Hot Deals

5 items

Trending

8 items

Quicl

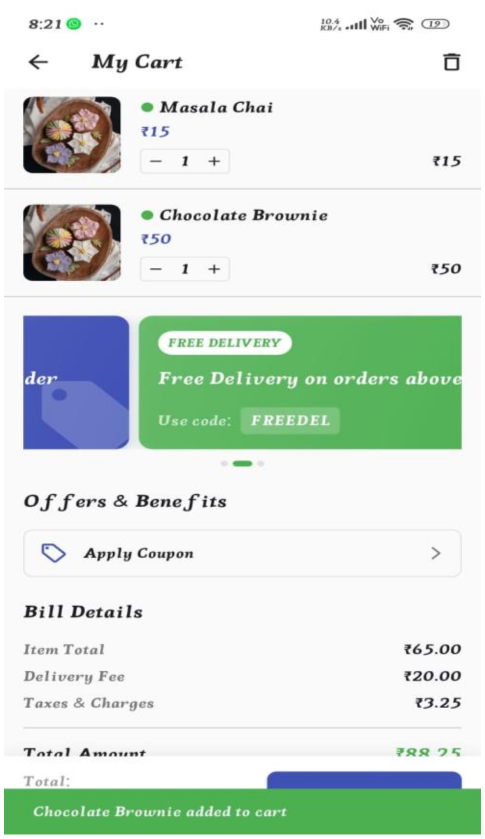
6 i

Categories

All

Popular Items

View All


8:21

My Cart

● Masala Chai
₹15
− 1 + ₹15

● Chocolate Brownie
₹50
− 1 + ₹50

der

FREE DELIVERY
Free Delivery on orders above
Use code: FREEDEL

Offers & Benefits

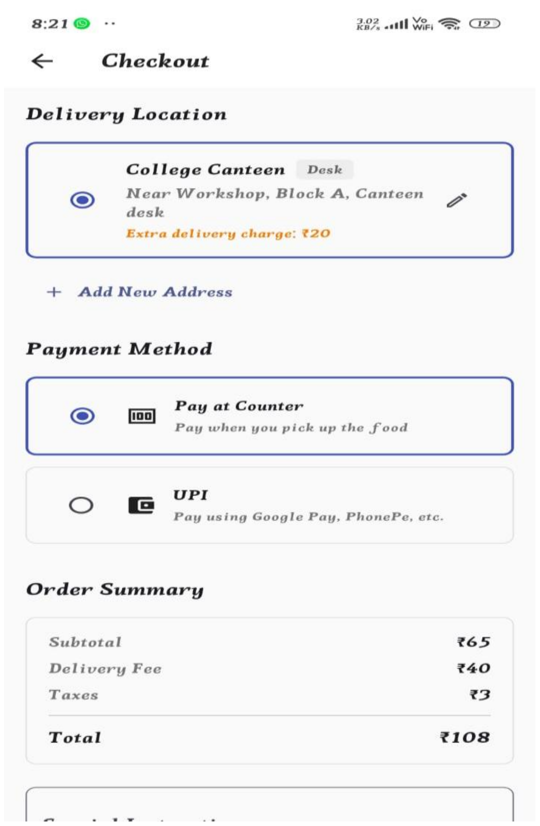
Apply Coupon >

Bill Details

Item Total	₹65.00
Delivery Fee	₹20.00
Taxes & Charges	₹3.25
Total Amount	₹88.25

Total:

Chocolate Brownie added to cart


8:21

Checkout

Delivery Location

College Canteen Desk
Near Workshop, Block A, Canteen desk
Extra delivery charge: ₹20

+ Add New Address

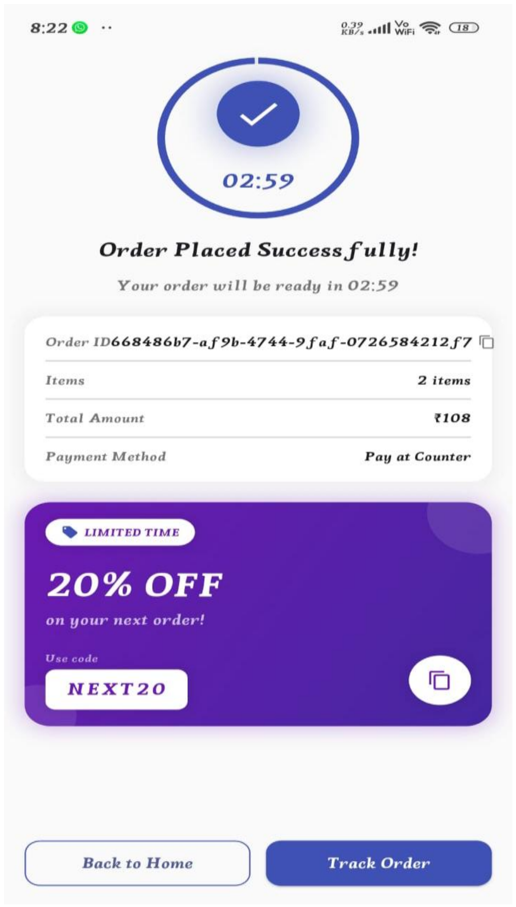
Payment Method

Pay at Counter
Pay when you pick up the food

UPI
Pay using Google Pay, PhonePe, etc.

Order Summary

Subtotal	₹65
Delivery Fee	₹40
Taxes	₹3
Total	₹108


8:22

02:59

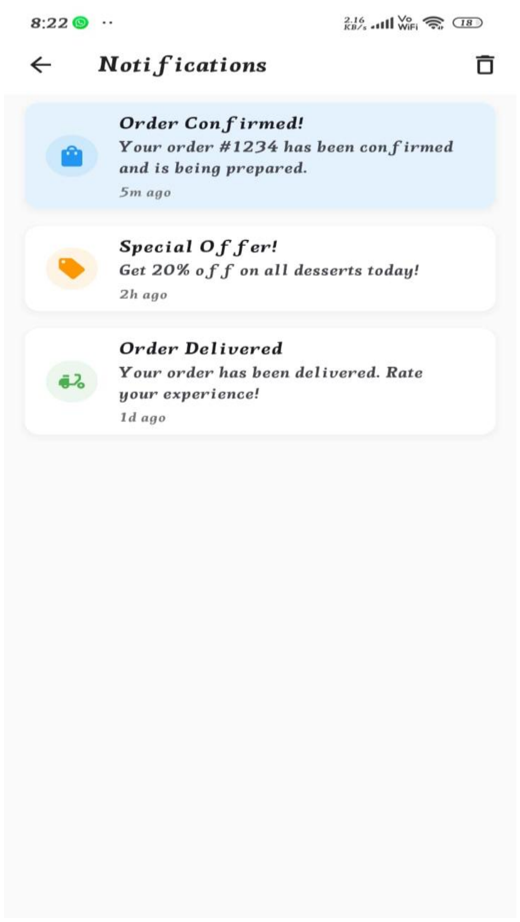
Order Placed Successfully!

Your order will be ready in 02:59

Order ID	668486b7-af9b-4744-9faf-0726584212f7
Items	2 items
Total Amount	₹108
Payment Method	Pay at Counter

LIMITED TIME
20% OFF
on your next order!
Use code
NEXT20

Back to Home
Track Order

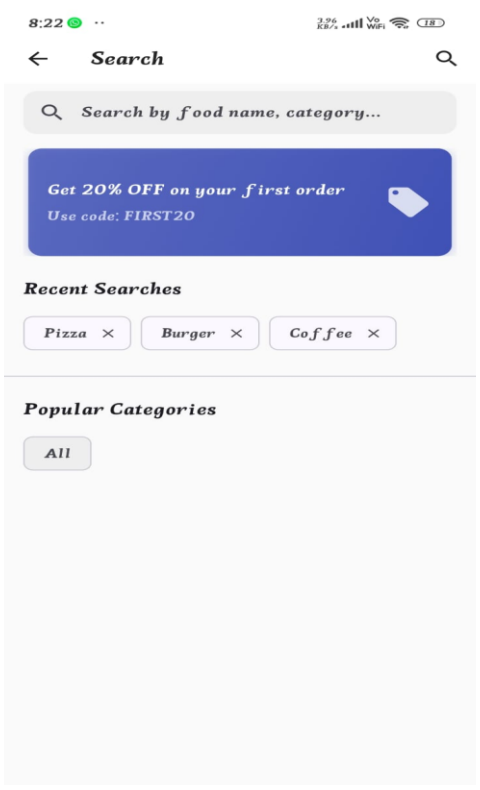

8:22

Notifications

Order Confirmed!
Your order #1234 has been confirmed and is being prepared.
5m ago

Special Offer!
Get 20% off on all desserts today!
2h ago

Order Delivered
Your order has been delivered. Rate your experience!
1d ago


8:22

Search

Search by food name, category...

Get 20% OFF on your first order

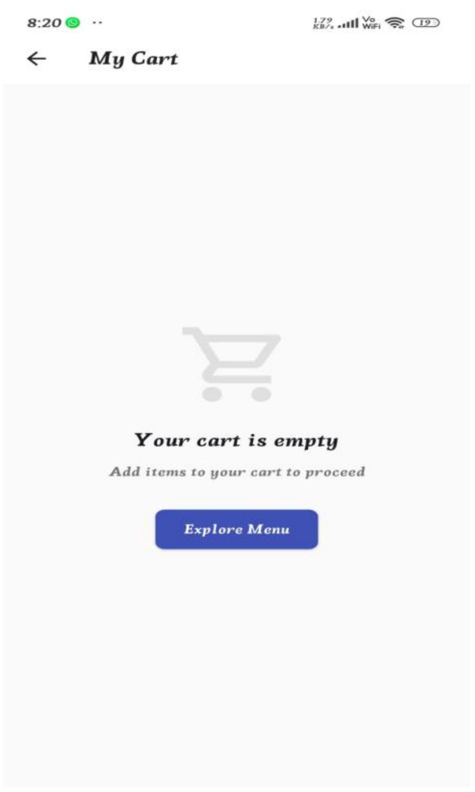
Use code: FIRST20

Recent Searches

Pizza ✕ Burger ✕ Coffee ✕

Popular Categories

All

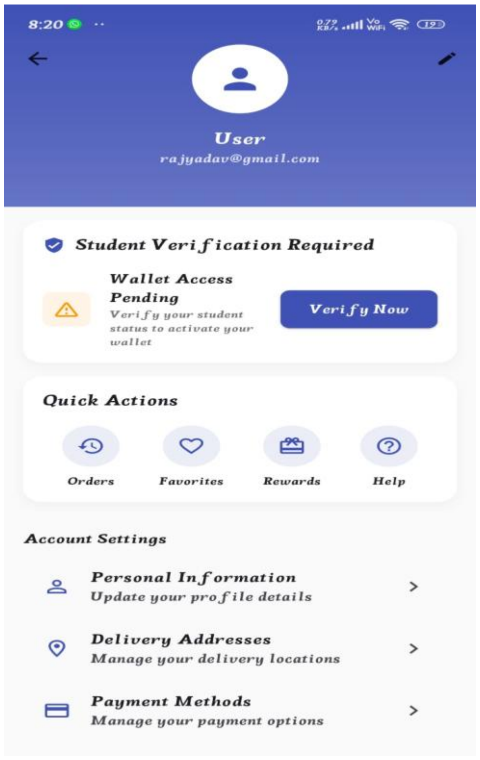

8:20

My Cart

Your cart is empty

Add items to your cart to proceed

Explore Menu


8:20

User

rajyadav@gmail.com

Student Verification Required

Wallet Access Pending

Verify your student status to activate your wallet

Verify Now

Quick Actions

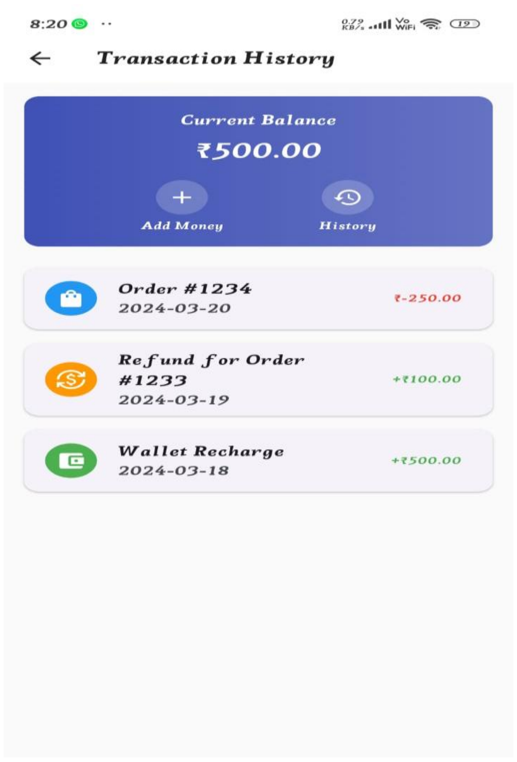
Orders Favorites Rewards Help

Account Settings

Personal Information
Update your profile details

Delivery Addresses
Manage your delivery locations

Payment Methods
Manage your payment options


8:20

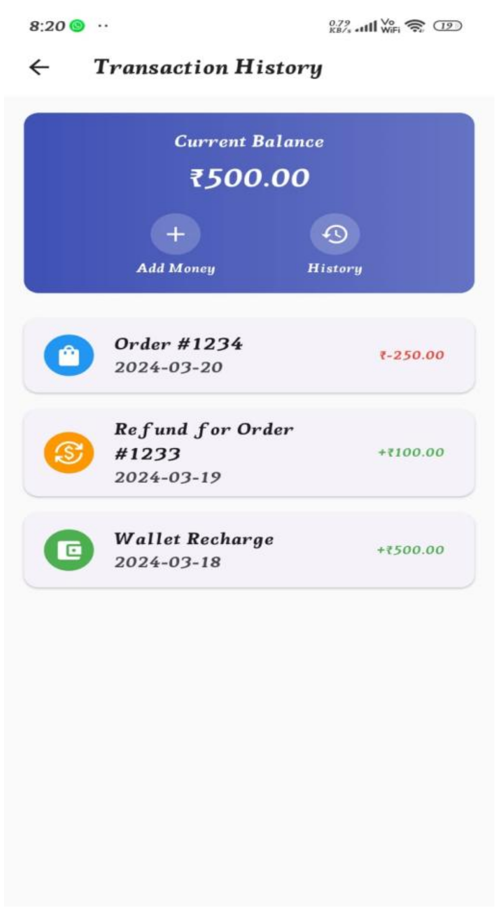
Transaction History

Current Balance

₹500.00

Add Money History

Order #1234	2024-03-20	₹-250.00
Refund for Order #1233	2024-03-19	+₹100.00
Wallet Recharge	2024-03-18	+₹500.00

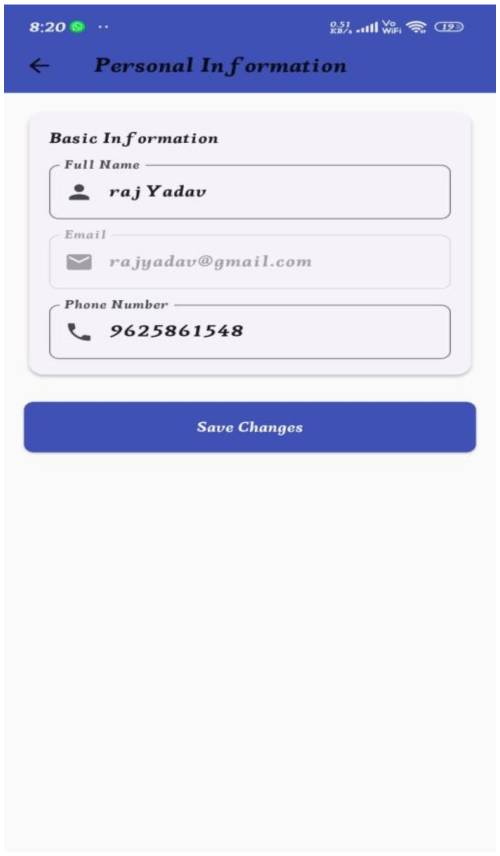

8:20

Transaction History

Current Balance

₹500.00

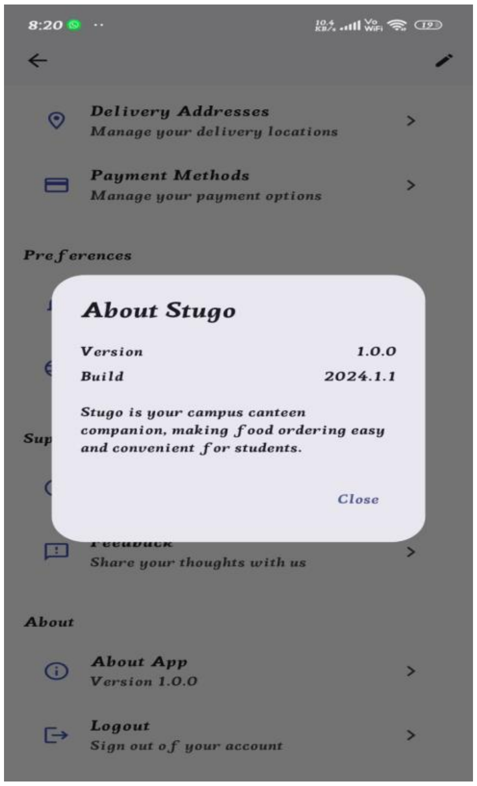
Order #1234 2024-03-20	₹-250.00
Refund for Order #1233 2024-03-19	+₹100.00
Wallet Recharge 2024-03-18	+₹500.00


8:20

Personal Information

Basic Information

Full Name	raj Yadav
Email	rajyadav@gmail.com
Phone Number	9625861548


8:20

Delivery Addresses
Manage your delivery locations

Payment Methods
Manage your payment options

Preferences

Sup

Feedback
Share your thoughts with us

About

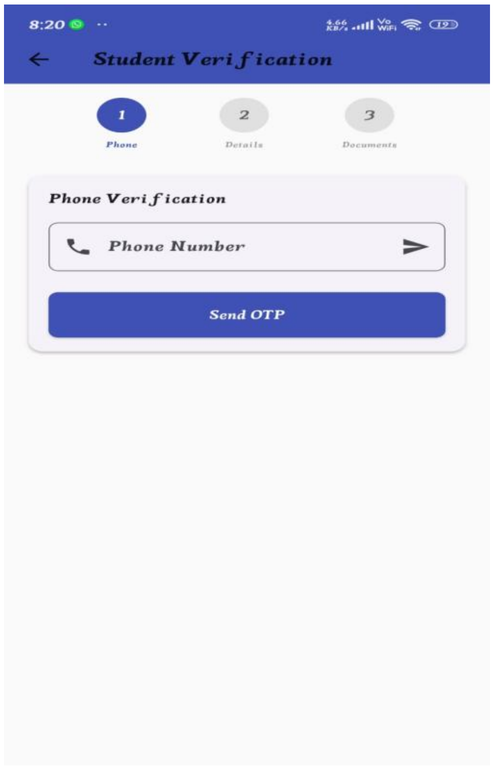
About App
Version 1.0.0

Logout
Sign out of your account

About Stugo

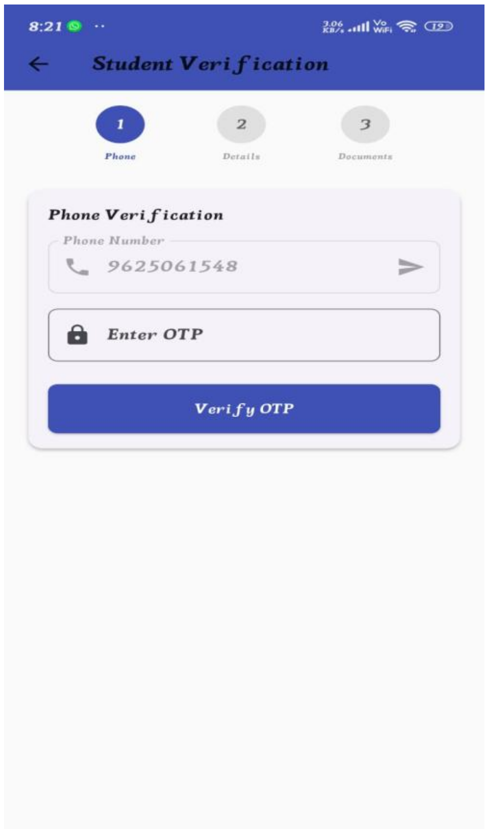
Version	1.0.0
Build	2024.1.1

Stugo is your campus canteen companion, making food ordering easy and convenient for students.


8:20

Student Verification

Phone Verification


8:21

Student Verification

1 Phone 2 Details 3 Documents

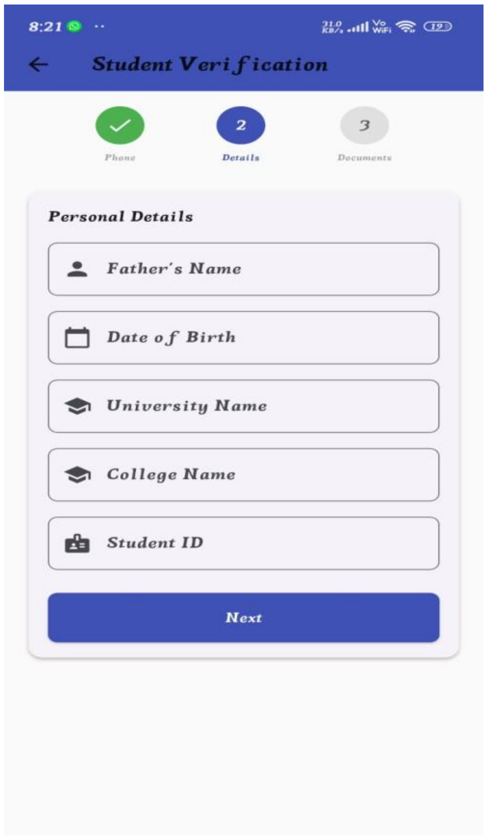
Phone Verification

Phone Number

9625061548

Enter OTP

Verify OTP


8:21

Student Verification

Phone 2 Details 3 Documents

Personal Details

Father's Name

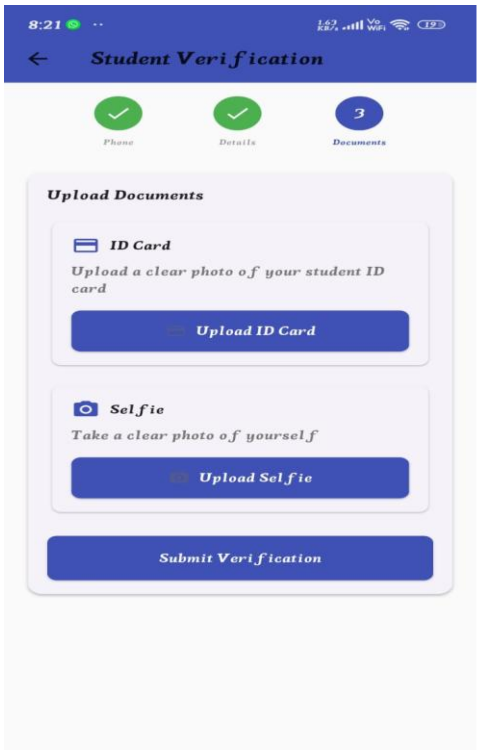
Date of Birth

University Name

College Name

Student ID

Next


8:21

Student Verification

Phone Details 3 Documents

Upload Documents

ID Card

Upload a clear photo of your student ID card

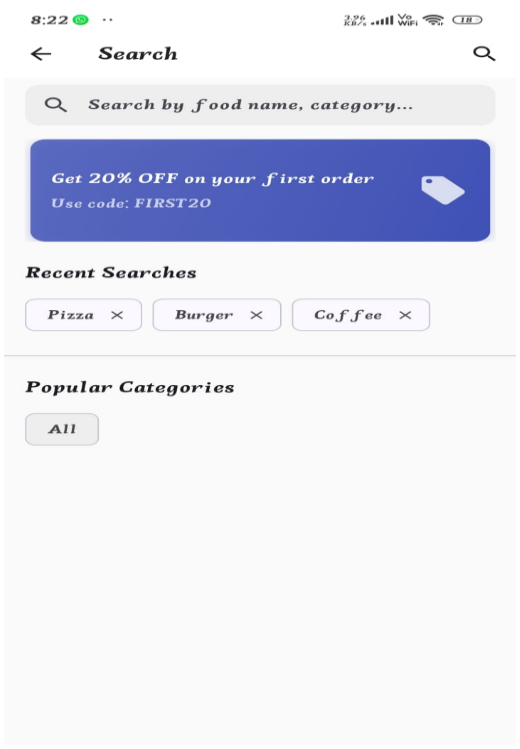
Upload ID Card

Selfie

Take a clear photo of yourself

Upload Selfie

Submit Verification


8:22

Search

Search by food name, category...

Get 20% OFF on your first order

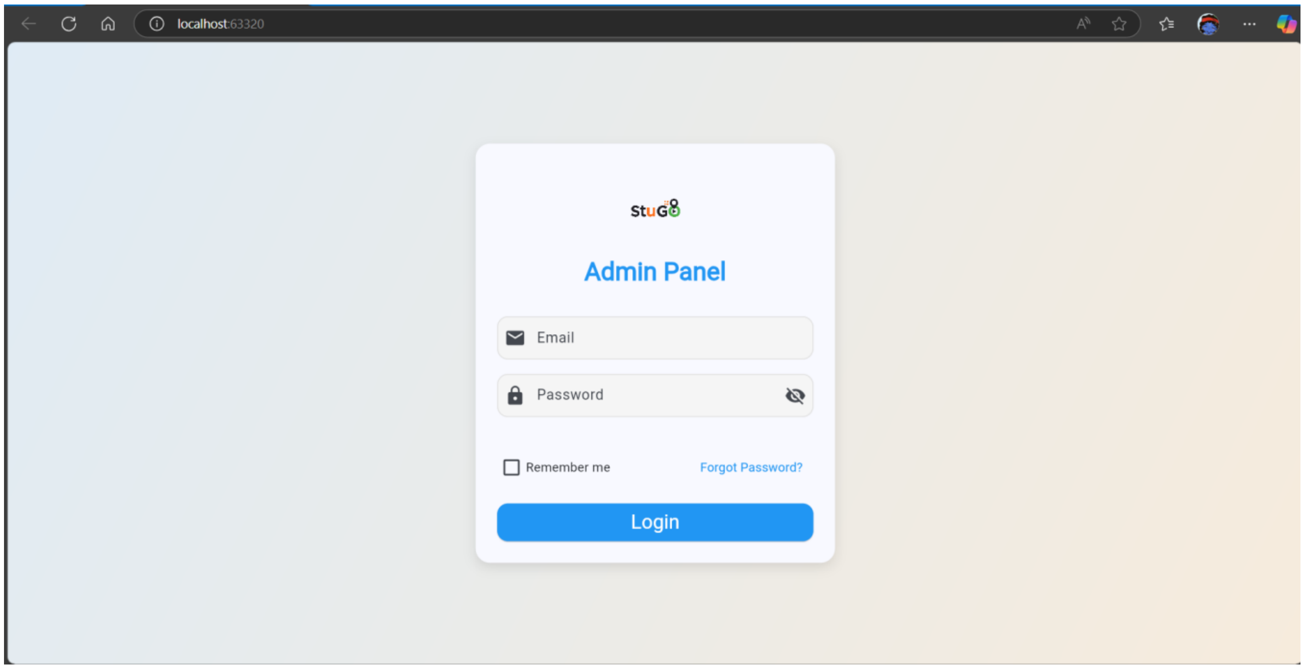
Use code: FIRST20

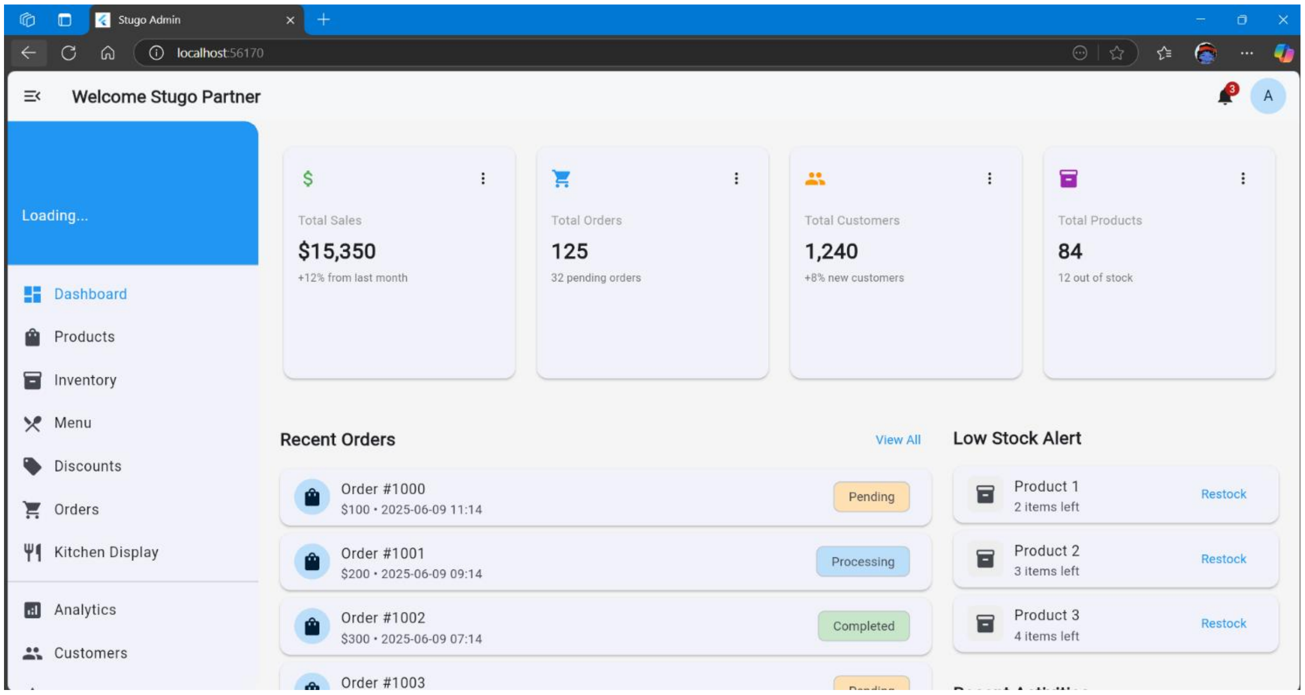
Recent Searches

Pizza × Burger × Coffee ×

Popular Categories

All





Total Sales
$15,350
+12% from last month

Total Orders
125
32 pending orders

Total Customers
1,240
+8% new customers

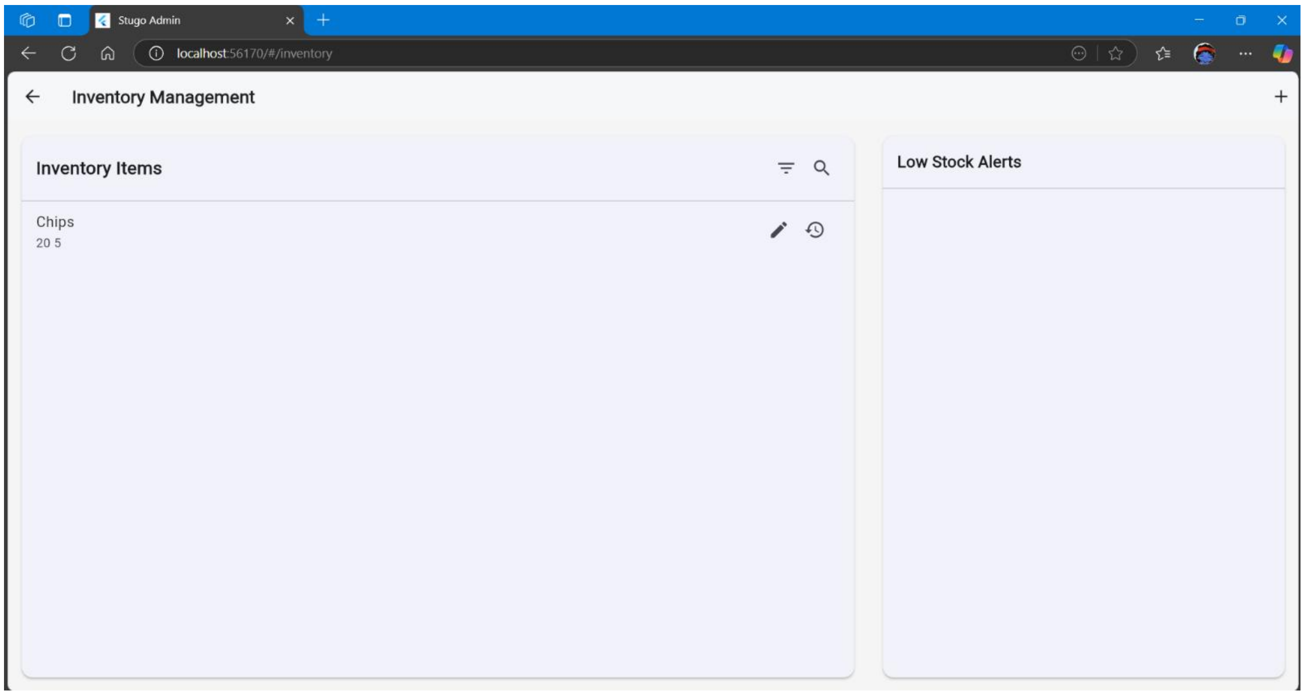
Total Products
84
12 out of stock

Recent Orders View All

Order #1000 $100 • 2025-06-09 11:14	Pending
Order #1001 $200 • 2025-06-09 09:14	Processing
Order #1002 $300 • 2025-06-09 07:14	Completed
Order #1003	

Low Stock Alert

Product 1 2 items left	Restock
Product 2 3 items left	Restock
Product 3 4 items left	Restock



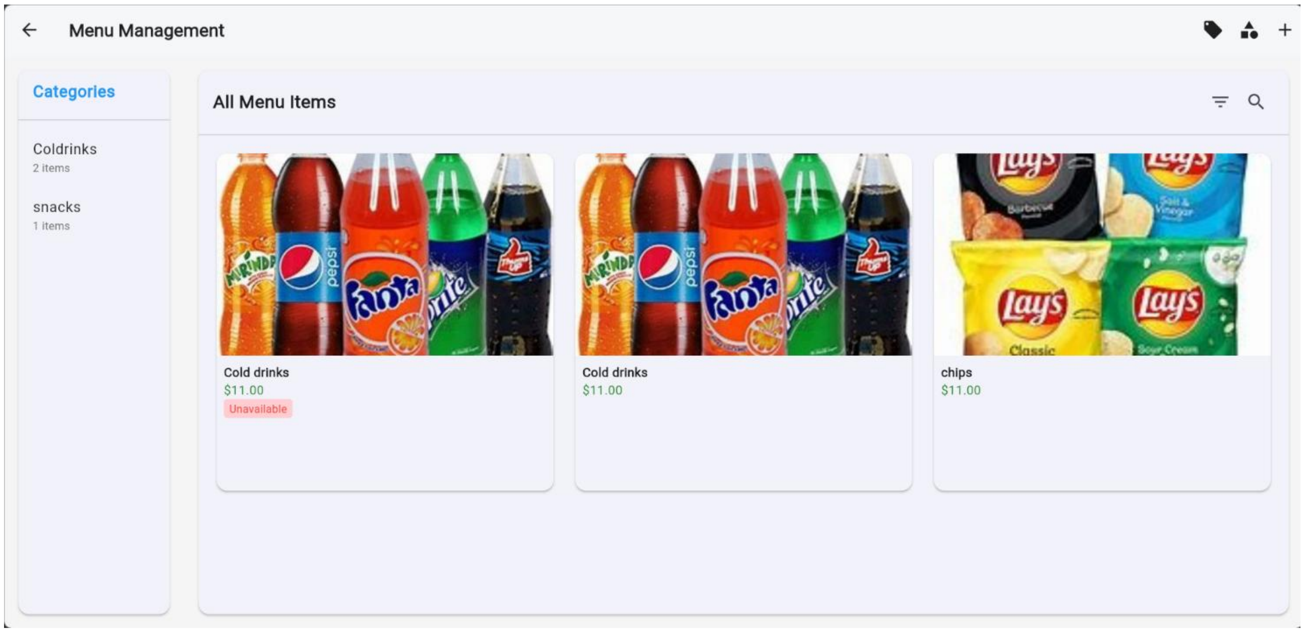
Inventory Management

Inventory Items

Chips
20 5

Low Stock Alerts

SIGNUP / REGISTER



Menu Management

Categories

Coldrinks
2 items

snacks
1 items

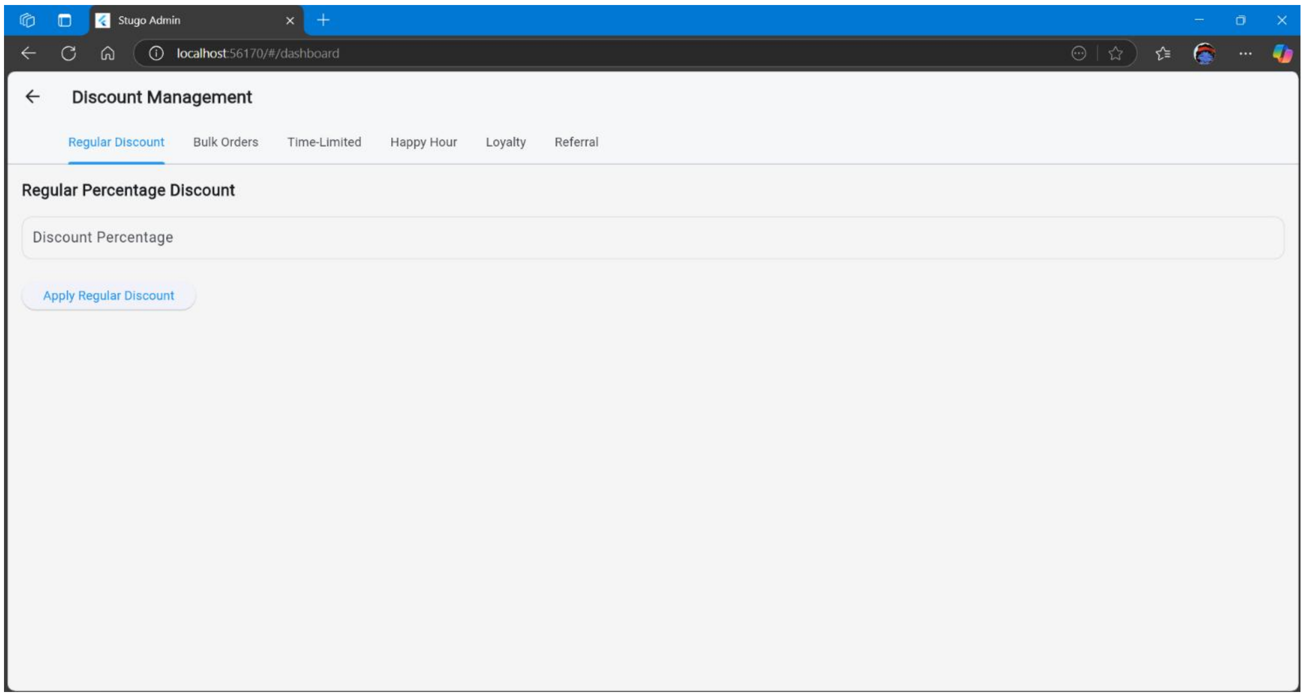
All Menu Items

Cold drinks
$11.00
Unavailable

Cold drinks
$11.00

chips
$11.00

LOGIN


Stugo Admin

localhost:56170/#/dashboard

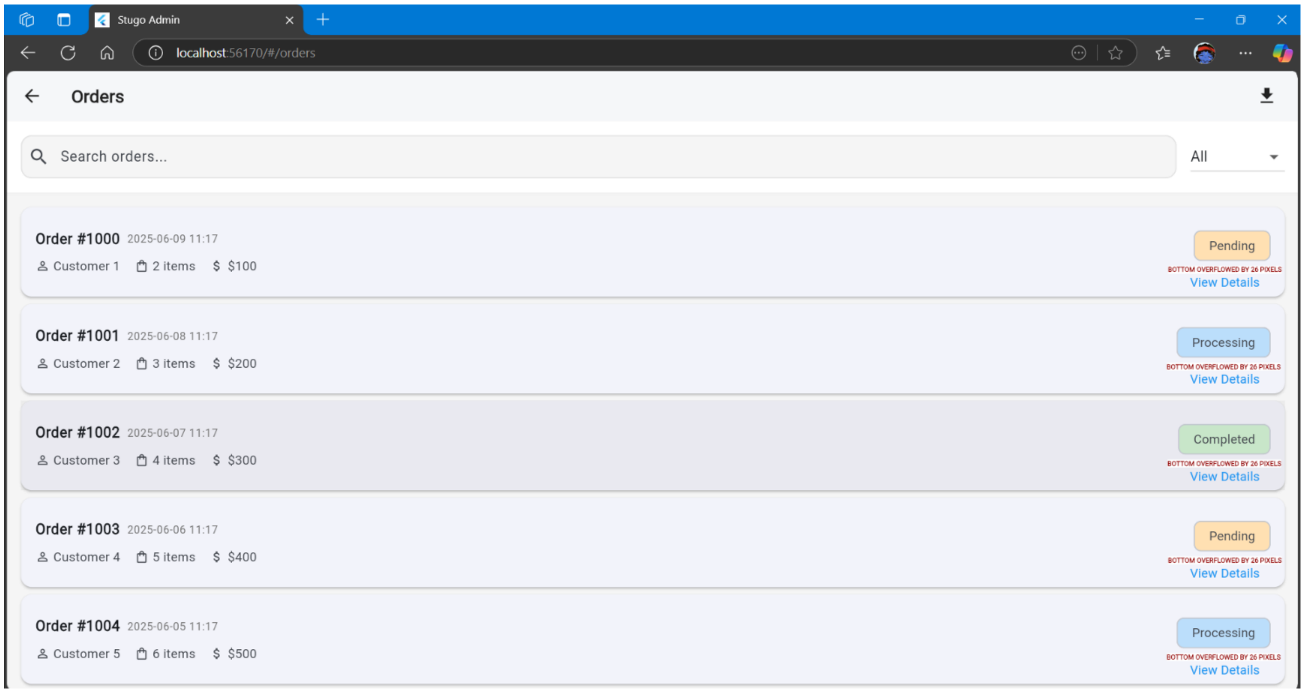
Discount Management

Regular Discount | Bulk Orders | Time-Limited | Happy Hour | Loyalty | Referral

Regular Percentage Discount

Discount Percentage

Apply Regular Discount


Stugo Admin

localhost:56170/#/orders

Orders

Search orders... All

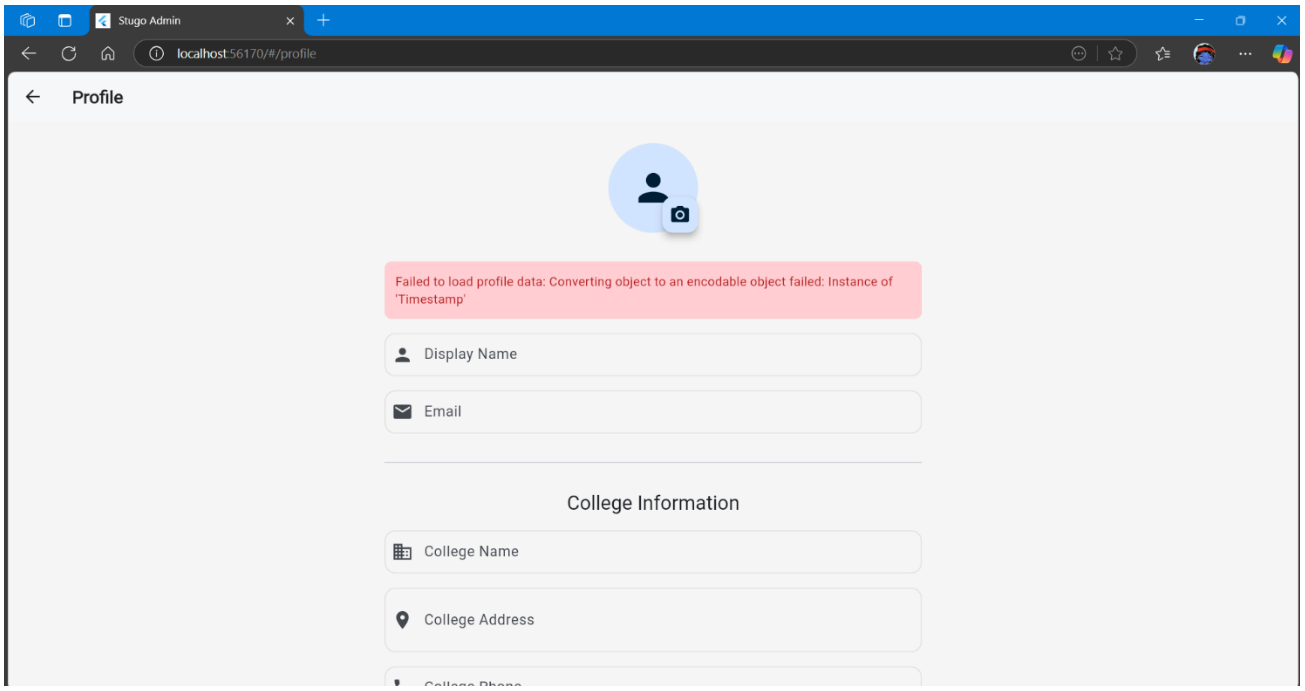
Order #1000 2025-06-09 11:17
Customer 1 2 items $100
Pending
BOTTOM OVERFLOWED BY 26 PIXELS
View Details

Order #1001 2025-06-08 11:17
Customer 2 3 items $200
Processing
BOTTOM OVERFLOWED BY 26 PIXELS
View Details

Order #1002 2025-06-07 11:17
Customer 3 4 items $300
Completed
BOTTOM OVERFLOWED BY 26 PIXELS
View Details

Order #1003 2025-06-06 11:17
Customer 4 5 items $400
Pending
BOTTOM OVERFLOWED BY 26 PIXELS
View Details

Order #1004 2025-06-05 11:17
Customer 5 6 items $500
Processing
BOTTOM OVERFLOWED BY 26 PIXELS
View Details


Stugo Admin

localhost:56170/#/profile

Profile

Failed to load profile data: Converting object to an encodable object failed: Instance of 'Timestamp'

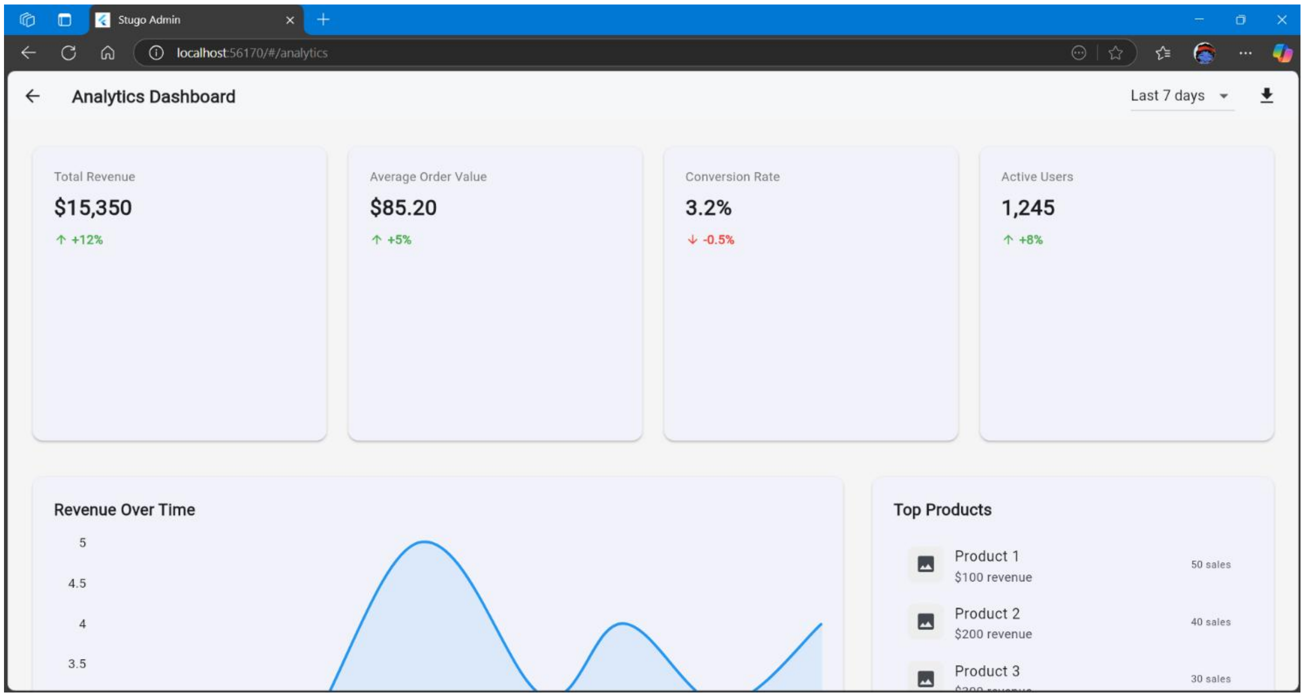
Display Name

Email

College Information

College Name

College Address


Stugo Admin

localhost:56170/#/analytics

Analytics Dashboard

Last 7 days

Total Revenue
$15,350
+12%

Average Order Value
$85.20
+5%

Conversion Rate
3.2%
-0.5%

Active Users
1,245
+8%

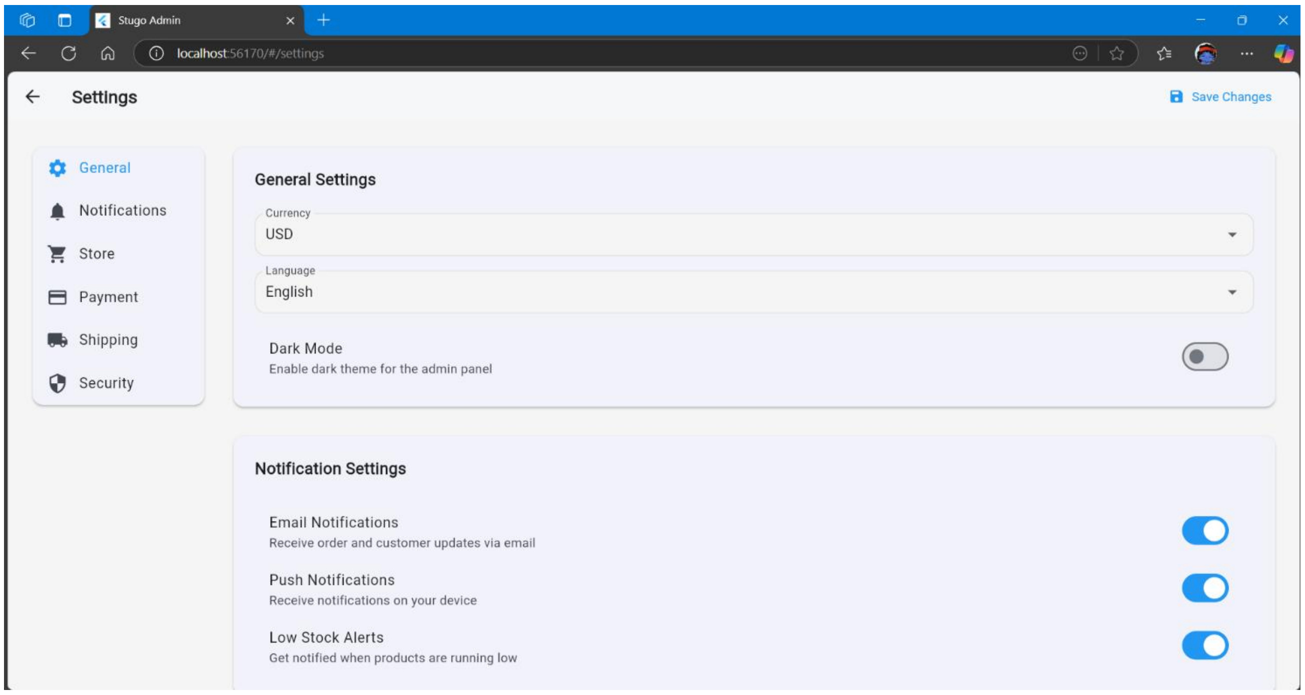
Revenue Over Time

Top Products

Product 1
$100 revenue
50 sales

Product 2
$200 revenue
40 sales

Product 3
30 sales



General Settings

Currency: USD

Language: English

Dark Mode
Enable dark theme for the admin panel

Notification Settings

Email Notifications
Receive order and customer updates via email

Push Notifications
Receive notifications on your device

Low Stock Alerts
Get notified when products are running low

CHAPTER-11

REFERENCES

REFERENCES

1. Flutter Dev Documentation:

   https://flutter.dev

2. Packages Documentation:

   https://pub..dev.

3. Firebase Documentation:

   https://firebase.google.com

4. Spreadsheet Documentation:

   https://Docs.google.com

5. Drizzle ORM Documentation:

   https://orm.drizzle.team/docs/

6. Cloud nary for image storage:

   https://cloudinary.com

7. QR Code Generation Library:

   https://www.npmjs.com/package/qrcode

8. React Documentation (if frontend is React-based):

   https://reactjs.org/docs/getting-started.html

9. GitHub for Project Collaboration:

   https://github.com/

Raj Kumar

Tigaon faridabad, Haryana

+91-9625861548 rky1524@gmail.com rajkumaryadav13 coderx-raj Portfolio

Profile

B.Tech CSE Student /Software Developer /Hackathon Winner

Full Stack Software developer with Expertise in Front-end, Experienced in leading cross-functional teams.

EXPERIENCE

Intern at BirbalSuredia and CO **Jan 2025 - June 2025**

Role - Full Stack Web Developer *Faridabad, Haryana*

- Gained practical experience on live projects, managing domain registration and server integration.
- Worked on implementing algorithms, and optimized loading times to enhance user experience.
- Updated content and features in real-time, using best practices for **SEO** to improve search rankings.
- Managed database systems and developed a responsive front-end using **HTML, CSS, and JavaScript**.

PROJECTS

Xypo.in (First project in 3rd sem) | Html, Css, Js, Google Sheets, Github, Slice etc. **2022 - 23**

- Designed a user-friendly interface to simplify onboarding and engagement for YouTube channel growth.
- Implemented features and role-based authorization to ensure secure access and personalized experiences.
- Developed secure RESTful APIs, integrating Microsoft Clarity for enhanced tracking and analytics.
- Integrated database functionalities with Google Sheets and Google Console for seamless data management.
- Enabled users to paste their YouTube channel links, select a subscription plan, and complete payments to gain high-quality subscribers, likes, comments, and watch hours to accelerate their growth.

Int-prep | React, JavaScript, Opan-ai api, IDE - VS Code **SEP-2023**

- Developed an AI-driven interview preparation page that allows users to practice with an avatar simulating real interview scenarios.
- Created a dashboard for users to view and manage upcoming mock interviews, enhancing their preparation experience.
- Implemented editing features for users to update interview details while maintaining validation for a seamless experience.

ACEM SEDULAR - Mobile Application | Flutter, Google Sheets, Amazon Appstore **2023-24**

- Published app on **Amazon Appstore**, integrated **Real-time data synchronization**.
- Module Consists of 4-5 different types of users with different functionalities.
- Added **10-12 new features** like organize the user interface of the Module's **Home Page**
- Handle the database and integrated the module with other existing modules.

TECHNICAL SKILLS

Languages: C, C++, Python
Technologies/Frameworks: HTML, CSS, Flutter, React, Node.js, Bootstrap, MongoDB
Developer Tools: VS Code, PyCharm, IntelliJ, Canva, Android Studio

CERTIFICATIONS

- Artificial Intelligence - INFOSYS FOUNDATION..
- C++ - NPTEL
- B2B, B2C, - NPTEL

Hackathons and Workshops

Hackathon Achievements2021-2024Various HackathonsNational Level

- Led a team that won multiple hackathons, including **college and national-level events**.

EDUCATION

Aravali College Of Engineering And Management, Jasana **2021 – 2025**

B.Tech - Computer Science and Engineering - ***SGPA - 7.10*** *Jasana, Haryana*