

Project Report on

**“Pizza Ordering App”**

**CSE299 – Junior Design Course**

**Summer 2019**

Section: **3**

Submitted to:

**Dr.Md.Monirujjaman Khan(KMM Sir)**

Prepared by:

**Md . Marjan** – **1621396042**

[marjan.md@northsouth.edu](mailto:marjan.md@northsouth.edu)

**Khan Zuhair Zahin Rafid-1521114042**

Submission date:

**05.09.2019**

**APPROVAL**

Md Marjan (ID #1621396042) and Khan Zuhair Zahin Rafid

(ID #1521114042) from Electrical and Computer Engineering Department of North South University, have worked on the Junior Design Project titled “A mobile application based automated disease predicting doctor” under the supervision of Dr. Mohammad Monirujjaman Khan partial fulfillment of the requirement for the degree of Bachelors of Science in Engineering and has been accepted as satisfactory.

**Supervisor’s Signature**

…………………………………….

**Dr. Mohammad Monirujjaman Khan**

**Associate Professor**

Department of Electrical Engineering & Computer Science

North South University

Dhaka, Bangladesh.

.

**DECLARATION**

This is to certify that this Project is our original work. No part of this work has been submitted elsewhere partially or fully for the award of any other degree or diploma. Any material reproduced in this project has been properly acknowledged.

Students’ names & Signatures

**1. Md Marjan**

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

**2. Khan Zuhair Zahin Rafid**

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

**CHAPTER 1**

**OVERVIEW**

**Abstract**

A pizza restaurant also known as quick service restaurant (QSR) within the food service industry is a specific type of restaurant characterized both by its pizza cuisine and by minimal table service. Food served in pizza restaurants is offered from a limited menu, cooked after ordering, packaged for order and is usually available ready for pickup or to be delivered though seating may also be provided. Some restaurants have the provision of customers making a call to the restaurant in advance to order a pizza to be ready for them for pick or to be delivered to them.

This project is aimed at developing a complete online ordering system app for use in the food service industry which will allow the restaurants to quickly and easily manage an online menu which customer can browse and use to place orders with just a few clicks. The customers will have to choose whether they want the food to be delivered to them and the payment method will be upon delivery or online payment.

The development of this system will be based on Android Studio with JAVA as the programming languages while Firebase Database as the database of the system.

**1.1 Introduction**

Life in the 21st century is full of technological advancement and in this technological age it is very difficult for any organization to survive without utilizing technology. In today’s age of fast food and take-out, many restaurants have chosen to focus on quick preparation and speedy delivery of orders rather than offering a rich dining experience. Until very recently, all of these delivery orders were placed over the phone call.

This Pizza Ordering App is based on a concept of making pizza orders and payments. Here, the user can login, select pizza sets and proceed towards payment. Other features include viewing full payment receipt. Prices will be displayed in BDT currency. This mini project contains limited features, but the essential ones.

Talking about the features of the Pizza Ordering System, a user can view all the sizes in the order menu with their respective prices and select among them. Then the user has to make custom pizza to proceed towards payments. The system displays the total amount and the user has to pay the amount equivalent to his/her total cost or more than that via online or cash on delivery. After all these, the system asks whether to display payment receipt or not. If the user presses yes then the payment receipt is displayed which includes date, phone number, order number, cashier’s name and description with total quantity, price and amount.

In this system, the user can place an order by selecting the Pizza size, Crust, and various Toppings. The layout design will be simple and it can be used in different gadgets like tablets, smartphones & even smart televisions. This mini project is not difficult to use, operate and understand by the users. We will develop this Pizza Ordering App using Java Programming Language and different software tools will also be used for the development of it.

Moreover, this project will provide the simplest system for managing pizza orders and payment in a restaurant.

**1.2 Project Details**

There are two modules in our system:

1. **User’s Module:**

* The user can download the application from the playstore.
* After downloading he/she should sign up in our application for the future support provided from us.
* User can select a disease and answering some questions he/she can learn how much possibility he/she have to attack by that disease.

**2. Admin Module:**

The Admin module does an important task. They will monitor the entire system. At that point we have the indispensable module of this system which is the administrator module. The activity of the administrator module is to screen every one of the assignments in the system. Administrator has the option to make new clients if anyone has a problem to signing up in this application. Subsequently the administrator has the full benefit to make any kind of changes all through the whole system.

**Installing app**

**Select a Pizza**

**Select Payment Method**

**Confirm Order**

Figure 1.1: Process Flow Diagram

The figure 1.1 illustrates the process of mobile application based Pizza Ordering Application.

|  |  |  |
| --- | --- | --- |
| No. | User | Admin |
| 01 | Sign up | See the statistics (how much people are using the application) |
| 02 | Select a pizza and Check and Confirm payment method and Confirm order. | N/A |
| 03 | App using problem facing. | Admin will handle the situation. |
| 04 | Problem with signing up | Admin will manually do sign up for that user. |

Figure 1.2: Module Flow Table

**1.3Project Goals**

The goal of the project is to concern our country people’s time and money. So that we are making this app. Using this app people will easily choose and order a pizza and get the home delivery service. This is mainly our project goal.

## **Goals of this system**

* People can choose their desired pizza.
* People can order pizza online through wifi/mobile data.
* They can pay either cash on delivery or bkash.
* We can help them saving their time and money.

**1.4 Summary**

This chapter gave us the insight of the modules that we have in the proposed system. This chapter provided a clear picture on how this system is effective to use in online pizza ordering system and our project goals.

**CHAPTER 2**

**MOTIVATION**

**2.1 Motivation**

With the rapid development of information technology, web application and Android application have been increasing in recent years. Compared with the Desktop application, the advantages of the Android application are as follows:

- Mobile application is convenient to carry

- Global partnerships and large install base

- Powerful development framework

- Open marketplace for distributing apps

Based on the advantages of android applications, we motivated ourselves to develop a project on Android application.

The whole world is in love with pizzas. The billions of dollars earned by different pizzerias across the globe just prove this. Meanwhile, with the number of customers increasing, the new problem occurs. Because the space of the restaurant is limited, the restaurant can only serve a certain number of customers at the time.

Therefore, the full customer resource cannot be utilized. Mobile Pizza Ordering Application is the key to solve this problem. Using this application, the customers need not go to the restaurant by themselves, but they can order the pizzas through Android mobiles anywhere. As the internet users are increasing exponentially, many companies have introduced Online Pizza ordering system for taking orders from customers.

This system also greatly lightens the load on the restaurant’s end, as the entire process of taking orders is automated. This system not only improves customer’s experience but also eases the workload on the staff of pizzerias. This allows restaurant stuffs to quickly go through the orders as they are placed and produce the necessary items with minimal delay and confusion.

**2.2 Objectives**

**Build your own pizza -** This system will help customers in ordering custom pizzas. So, the customer will pick exactly the things which he/she wants in his/her pizza. This will surely enhance the image of the pizzeria and customer satisfaction will be more.

**Online Payment -** This system will give the option to the customer for online payment. This will make pizza buying experience cash free.

**Better Knowledge -** This system will provide customer all the details of his order before making the order. This confirmation will help customers to check the items ordered with their prices.

**Know Delivery Time -** This system will show the time by which the order will be delivered to the customer. For pick-ups, customers can fix the time by which they will pick their order.

**Reduce Paper Work -** As most of the things will be performed online, it will reduce the usage of paper for the pizzeria.

**Improves Efficiency -** This system will make things easier for staff as whole ordering process is done by the customer only.

**2.3 User Justification**

The system will be designed to be user friendly. The user friendly and interactive interfaces design help to achieve this by enabling customers to easily browse through the menu and place orders with just a few clicks. The system will be simple to use. The user can increase efficiency by shortening the purchasing time and eliminating paper work like receipts through online transaction. This system will increase customer satisfaction by speeding up food delivery.

**CHAPTER 3**

**RELATED WORK**

**Methodology**

**3.1 How This System Works**

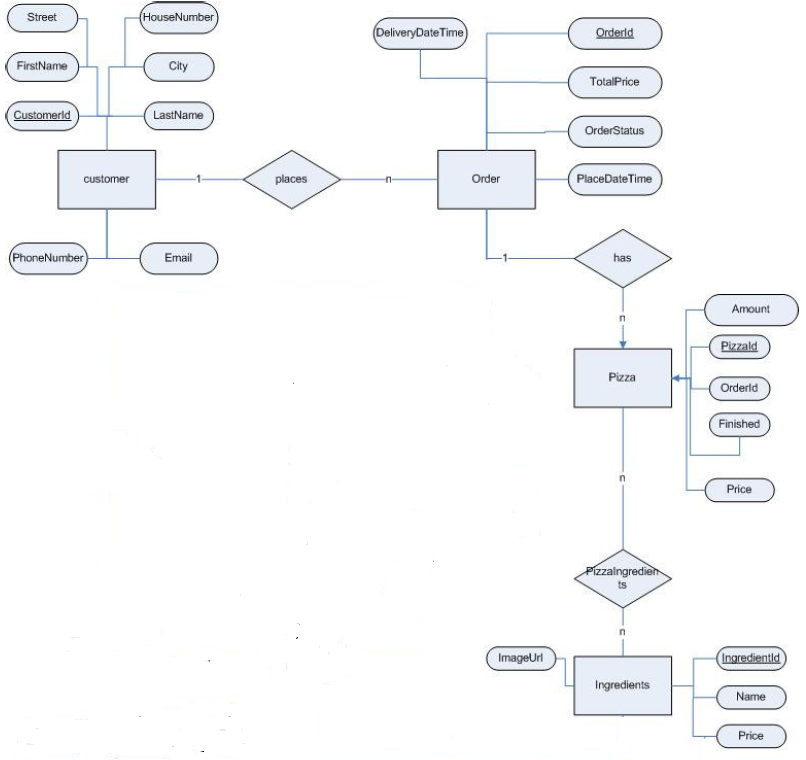
Whenever a customer visits the mobile application of the pizza ordering system, he/she will have to sign-up/ login. The customer has to provide his/her details like name, phone number, address into the registration field. After successfully logging in, the menu will be visible to the customer with the pizza customizations and other non-pizza products on offers. All the ingredients will be shown along with their prices.

Next, the customer will customize his pizza and make changes in the ingredients if he wishes for and select the quantity for it. After selecting the items to be ordered, the customer will select the type of drop-off process whether it will be a home delivery or a pick up.

Lastly, the payment option will be shown to the customer. He/she will choose from the various online payment methods or cash on delivery option. After this process, the order is made and the customer is notified about the time by which the pizza will be delivered to them.

**3.2 Project Diagram**

The proposed ER diagram for online pizza ordering system is shown below.



**3.3 Proposed Solution**

As people are being smarter every year and the uses of mobile phones,internet are increasing day by day so that we developed an application for ordering pizza online.In our country a smart phone is available to each and every family. It is very difficult to find a family or person who has no smart phone. So smart phone can be our main weapon to make people aware of saving time and money. An mobile application should be developed as soon as possible to make awareness among our country people.That mobile application must be developed like it can choose and order a pizza,user can pay through the app for saving both time and money.

**3.4 Summary**

Considering all aspects, our proposed solution is an optimal one. The online pizza ordering application is pretty much easier thing through which we can make our country people conscious. The system interface is easy and can be kept up by a non-IT proficient too. This system has been created so that it gives a sentiment of ease of use and there won't emerge any issues with respect to confusions. In this way this section gives the thought regarding the present systems that are accessible in the market and about the inspiration towards building up this dynamic system.

**CHAPTER 4**

**TECHNICAL DESIGN**

**4.1 Introduction**

In this chapter we discuss the aspect of the technical design of our system. By going through the system level design it would be easier to conceptualize the entire data flow of the system.

**4.2 Technical Design: System Level**

**Installing app**

**Select a Pizza**

**Select Payment Method**

**Confirm Order**

Figure 1.1: Process Flow Diagram

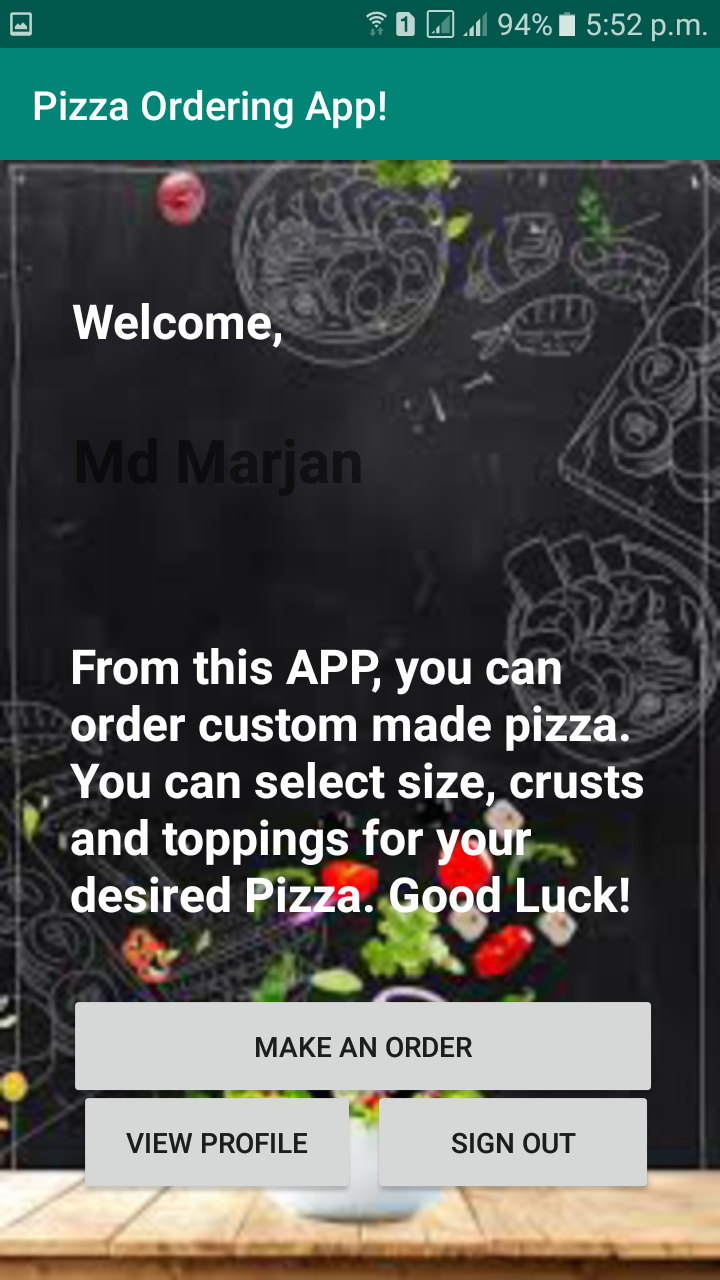


Figure 4.2: Home page of Pizza Ordering Application

In figure 4.1, a user will sign up for the registration. Then he/she will select a pizza and using the app he/she will order their desired pizza. In figure 4.2, It is shown that how does the homepage look like.

**4.3 Summary**

As mentioned earlier, the technical design has enabled us to get a clear picture of how our system is operating. Therefore considering the above data flow diagram we can comprehend the method in which our system is being operated.

**CHAPTER 5**

**SOFTWARE DESIGN**

**5.1 Introduction**

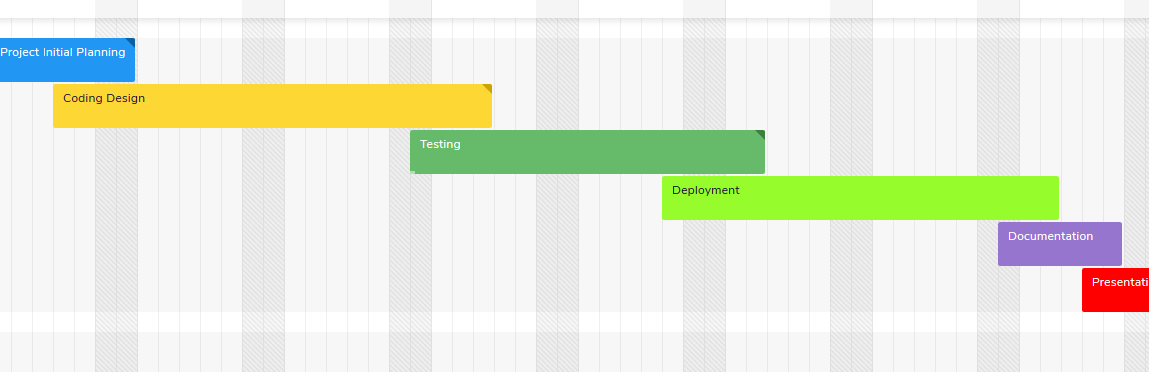
In this chapter we discuss the software design of our system. One can consider this part as the essential structure of the system as it will provide as the necessary features that our system entails.

**5.2 Planning**

**5.2.1 Gantt Chart**

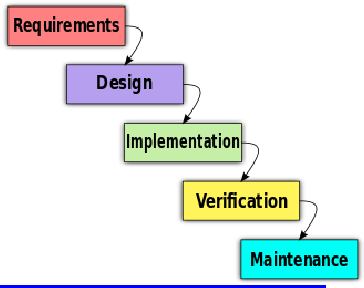
**June july july july july august august**

**30-6-19 7-7-19 14-7-2019 21-7-2019 28-7-2019 4-8-2019 -------- 25-9-2019**

****

**5.2.2 Process Model**

For this project we have planned to use waterfall as a process model. The waterfall model is a sequential design process, often used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing and Maintenance.

****

**5.3 Summary**

This chapter provided all the necessary details on the software design framework of our system.

**CHAPTER 6**

**SYSTEM FRAMEWORK**

**And FEATURES**

**6.1 Introduction**

In this chapter we explore the system in a real time manner through pictorial illustrations which will further help us to visualize the system before getting hold on to it physically. This chapter entails the step by step process on how the system is operated. Through the help of the visual representation it will enable the reader to get a clearer picture of the Automated Disease Predicting Doctor.

**6.2System Design Framework**

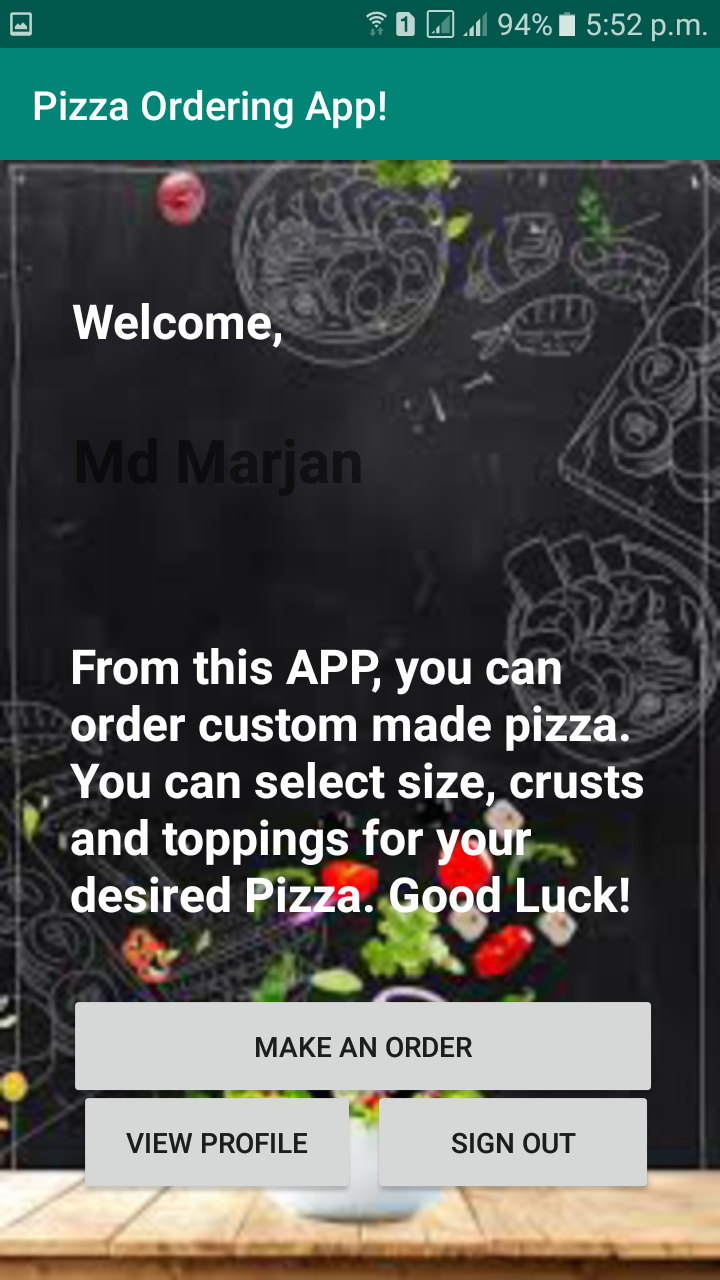


Figure6.2: Home Page of app

In figure 6.2 we observe the second part of the homepage of our mobile application. Here user will see the features of the app. They can select the their desired features to make an order.

**6.3 Features**

**a. Login**

|  |  |
| --- | --- |
| Here, a user can successfully login to the app with his phone number and password. The phone number and password should match with his given and saved information in the registration process. If the number is not in the database then system will show an error. Also, if the password does not match then it will also show an error. If the user is new, then he can register with the register button. | C:\Users\user\Desktop\69248988_496229557619439_8272645180088123392_n.png |

**b. Registration**

|  |  |
| --- | --- |
| Here, a user can successfully register to the app. The user needs to provide his name, phone number, address, house, road, block and password. After clicking the register button, all the information of that user will be stored to the database. If the phone number is existing in the database already, then it will show an error message. | C:\Users\user\Desktop\Summer 2019\CSE299\work\69277201_2910817745811105_2546620190087446528_n.png |

**c. View Profile**

|  |  |
| --- | --- |
| After clicking the view profile button in Home Page, a user can see his profile. Here, all the information of that user will be shown. All the information is directly getting retrieved from the database. User can also edit this information by clicking edit profile button. | C:\Users\user\Desktop\Summer 2019\CSE299\work\69515090_489593331610313_7429489649015324672_n.png |

**d. Edit Profile**

|  |  |
| --- | --- |
| Here, a user can edit all his given information. Initially, all the given information is shown in the text fields. The user just needs to edit those data field by field. Here, the user can also change his password. Clicking the update button, will allow all the updated information to update in the database. | C:\Users\user\Desktop\Summer 2019\CSE299\work\68958225_1168878986634856_6454839798640148480_n.png |

**e. Pizza Size Selection**

|  |  |
| --- | --- |
| For adding a new pizza, a user needs to select pizza size here. Three pizza sizes with prices are shown in this dialog box. The user must select a size to make progress of adding a new pizza. | C:\Users\user\Desktop\Summer 2019\CSE299\work\68919305_1087955444743350_3299885186686124032_n.png |

**f. Pizza Crust Selection**

|  |  |
| --- | --- |
| After selecting the pizza size, here, a user needs to select the pizza crust. Four types of crusts are shown in the dialog box. User must select a type to proceed next. | C:\Users\user\Desktop\Summer 2019\CSE299\work\68930820_1154137158117265_406791870321000448_n.png |

**g. Pizza Toppings Selection**

|  |  |
| --- | --- |
| Toppings selection is the last phase for customizing a new pizza. After selecting the pizza size and the crust accordingly, a user needs to select the toppings in terms of whole/left half/right half (i.e. refers to how much of the pizza you want the toppings to be scattered with). Various toppings are shown as pictures. User may procced next by clicking add button for selecting any toppings. | C:\Users\user\Desktop\Summer 2019\CSE299\work\69415351_365036947754193_6252176745961619456_n.png |

**h. Pizza Info Display**

|  |  |
| --- | --- |
| Here, the user can view his selections of the pizza. The pizza size, the crust, the toppings and the total amount will be shown in this page. The user can pay with the click on the pay button or can cancel with the cancel button. Basically, in this page all the selections the user has made for creating a new pizza are displayed to the user himself. | C:\Users\user\Desktop\Summer 2019\CSE299\work\69067633_359463391670688_7780942600894152704_n.png |

**i. Payment Selection**

|  |  |
| --- | --- |
| After clicking the pay button, user will proceed here for choosing the way of payment method. A dialogue box appears where the user needs to select a payment method to proceed next. “BKash” option is for making a payment through user’s own BKash number. If any user wants to pay using the cash on delivery method, then he may select “Cash On Delivery” option and it will then go to the summary page. | C:\Users\user\Desktop\Summer 2019\CSE299\work\69373906_1092520200937977_8054740701520330752_n.png |

**j. BKash Payment**

|  |  |
| --- | --- |
| After selecting the BKash option, a user will be displayed with a message here. The user needs to make a Payment to the given number. After which the user must input the TxnID of that Payment to make confirmation. After clicking ok, the user will go to the summary page with the TxnID. | C:\Users\user\Desktop\Summer 2019\CSE299\work\69262476_744546415998821_456165770678763520_n.png |

**k. Summary**

|  |  |
| --- | --- |
| This is the last step to complete an order. This summary page will show the selected pizza size, the crust, the toppings, the total amount, the user’s phone number and the selected payment method with details. The user needs to check all the information for self-verification, and click the confirm button for confirming the order. All the information will then be stored in the database. The user may click cancel to go back. | C:\Users\user\Desktop\Summer 2019\CSE299\work\69382857_509513416524900_7350847427730997248_n.png |

**6.4 Summary**

In this long yet useful chapter we managed to cover the entire work process structure of how the Automated Doctor Mobile Application has been developed.

**CHAPTER 7**

**SKILLS**

**7.1 Introduction**

In this chapter we discuss the skills that we have obtained in order to develop this massive sophisticated system.

**7.2 Skills obtained**

Through this project the following skills have been developed:

* **Skill in Programming & Tools**
* **JAVA PROGAMMING**

We learn and apply java programming in our application.After working almost four months for this project,we are quite confident about java programming now.

* **Sublime Text2**

Sublime Text is a cross-platform source code editor with a Python application programming interface (API). It natively supports many programming languages and markup languages, and its functionality can be extended by users with plugins, typically community-built and maintained under free-software license.

* **Java Android Skill**

**Android** is a [mobile operating system](https://en.wikipedia.org/wiki/Mobile_operating_system) developed by [Google](https://en.wikipedia.org/wiki/Google). It is based on a modified version of the [Linux kernel](https://en.wikipedia.org/wiki/Linux_kernel) and other [open source](https://en.wikipedia.org/wiki/Open-source_software) software, and is designed primarily for [touch screen](https://en.wikipedia.org/wiki/Touchscreen) mobile devices such as [smartphones](https://en.wikipedia.org/wiki/Smartphone) and [tablets](https://en.wikipedia.org/wiki/Tablet_computer). In addition, Google has further developed [Android TV](https://en.wikipedia.org/wiki/Android_TV) for televisions, [Android Auto](https://en.wikipedia.org/wiki/Android_Auto) for cars, and [Wear OS](https://en.wikipedia.org/wiki/Wear_OS) for wrist watches, each with a specialized user interface. Variants of Android are also used on [game consoles](https://en.wikipedia.org/wiki/Video_game_console), [digital cameras](https://en.wikipedia.org/wiki/Digital_camera), [PCs](https://en.wikipedia.org/wiki/Personal_computer) and other electronics.

**7.3 Summary**

In this chapter we discussed the list of skills that have been obtained throughout the process of developing and materializing this system.

**CHAPTER 8**

**ESSENTIAL PARTS AND DEVICES**

**8.1 Introduction**

In this chapter, we shed light on the tools that we used to develop this sophisticated system and we also discuss what tools will be required if one wants to test this system in there one spheres.

**8.2 Design Requirements**

The system needs to have a front-end that not only makes the site user-friendly but also makes it easier to make modifications. Similarly the back-end must be compatible enough to support the front-end conveniently and also be flexible enough to adjust to any structural changes within the front end. We used the following tools for our system:

**1. Front End:**

* Android java

**2. Back End:**

* Firebase

In general, a framework is a real or conceptual structure intended to serve as a support or guide for the building of something that expands the structure into something useful.In computer systems, a framework is often a layered structure indicating what kind of programs can or should be built and how they would interrelate. Some computer system frameworks also include actual programs, specify programming interfaces, or offer programming tools for using the frameworks. A framework may be for a set of functions within a system and how they interrelate; the layers of an operating system; the layers of an application sub-system; how communication should be standardized at some level of a network; and so forth. A framework is generally more comprehensive than a [protocol](http://searchnetworking.techtarget.com/definition/protocol) and more prescriptive than a [structure](http://searchnetworking.techtarget.com/definition/protocol).

**4. Software:**

* + Android Studio
  + Java SDK

**5. Operating System:**

* Win-XP, Win-7, Win-8 or higher version
* Linux or any other higher version.
* Mac

**8.3 Test Requirements**

In order to test the Smart Online Health Care System the following tools are necessary:

**1. Internet Connectivity**

**2. Domain/Server**

**3. Hardware:**

* + Processor (Core i3, Core i5 or higher )
  + RAM (4 GB or Higher )
  + Disk Space
  + Android mobile phone

**4. Operating System:**

* Win-XP, Win-7, Win-8 or higher version
* Linux or any other higher version

**8.4 Summary**

In this chapter we described what tools we have taken benefit of in order to build this sophisticated system. Furthermore, we have also learnt what tools we need in order to conduct a thorough testing of this system from both the user and the admin’s perspective.

**CHAPTER 9**

**WORKING SHEETS**

**9.1 Introduction**

In this chapter, we observe the entire work structure, meaning how the scheduling was maintained throughout the developmental phase. We shall also see the financial foundation of this project and furthermore the feasibility study should be also discussed.

**9.2 Work Breakdown Structure**

We have divided the work breakdown structure into two parts. In order to develop this system, we gave importance to scheduling because we believe that if we want to provide the best of quality then we must give due importance to scheduling which helped us to garner better results. The figure below focuses the weekly work we had accomplished.

|  |  |
| --- | --- |
| **Week No.** | **Proposed Work** |
| Week-1 | Project Proposal Report and Presentation |
| Week-2 | Creating web and mobile application |
| Week-3 | Home Page and Login for web and app |
| Week-4 | Creating the logic programs for result view |
| Week-5 | Tester (Check Pending Test & Test Result) |
| Week-6 | Final Report and Presentation |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Figure 9.1: Work Breakdown Structure (1st part)

**9.3 Financial Plan**

Initially, we visited some well-known Restaurant to meet some managers for consulting about this project. Thus we had to spend a small amount on transportation. We used our personal desktop computer to develop this system. For testing, we needed to purchase domain server, hosting license and virtual serverwhich costs Tk. 5,000.

**9.4 Feasibility Study**

Depending on the results of the initial investigation the survey is now expanded to a more detailed feasibility study. “**FEASIBILITY STUDY**” is a test of system proposal according to its workability, impact of the organization, ability to meet needs and effective use of the resources. It focuses on these major questions:

1. What are the user’s demonstrable needs and how does a candidate System meets them?

2. What resources are available for given candidate system?

3. What are the likely impacts of the candidate system on the organization?

4. Whether it is worth to solve the problem?

During feasibility analysis for this project, following primary areas of interest are to be considered. Investigation and generating ideas about a new system does the following steps:

**Steps in feasibility analysis**

1. Form a project team and appoint a project leader.

2. Enumerate potential proposed system.

3. Define and identify characteristics of proposed system.

4. Determine and evaluate performance and cost effectively of each proposed system.

5. Weight system performance and cost data.

6. Select the best-proposed system.

7. Prepare and report final project directive to management.

**Technical feasibility**

A study of available resource that may affect the ability to achieve an acceptable system. This evaluation determines whether the technology needed for the proposed system is available or not.

* Can the work for the project be done with current equipment existing software technology & available personal?
* Can the system be upgraded if developed?
* If new technology is needed then what can be developed?

This is concerned with specifying equipment and software that will successfully satisfy the user requirement. The technical needs of the system may include:

**Front-end and back-end selection**

An important issue for the development of a project is the selection of suitable front-end and back-end. When we decided to develop the project we went through an extensive study to determine the most suitable platform that suits the needs of the organization as well as helps in development of the project. The aspects of our study include the following factors:

**Front-end selection:**

1. It must have a graphical user interface that assists employees thatare not from IT background.

2. Scalability and extensibility.

3. Flexibility.

4. According to the organization requirement and the culture.

5. Must provide excellent reporting features with good printing support.

6. Platform independent.

7. Easy to debug and maintain.

8. Event driven programming facility.

**Back-end Selection:**

1. Multiple user support.

2. Efficient data handling.

3. Provide inherent features for security.

4. Efficient data retrieval and maintenance.

5. Stored procedures.

6. Popularity.

7. Operating System compatible.

8. Easy to install.

9. Various drivers must be available.

10. Easy to implant with the Front-end.

**Economic feasibility**

Economic justification is generally the “Bottom Line” consideration for most systems. Economic justification includes a broad range of concerns that includes cost benefit analysis. In this we weight the cost and the benefits associated with the candidate system and if it suits the basic purpose of the organization i.e. profit making, the project is making to the analysis and design phase.

The financial and the economic questions during the preliminary investigation are verified to estimate the following:

* The cost to conduct a full system investigation.
* The cost of hardware and software for the class of application beingconsidered.
* The benefits in the form of reduced cost.
* The proposed system will give the minute information, as a result the performance is improved which in turn may be expected to provide increased profits.
* This feasibility checks whether the system can be developed with the available funds.

The **PIZZA ORDERING APP** does not require enormous amount of money to be developed. This can be done economically if planned judicially, so it is economically feasible. The cost of project depends upon the number of man-hours required.

**Operational Feasibility**

It is mainly related to human organizations and political aspects. The points to be considered are:

* What changes will be brought with the system?
* What organization structures are disturbed?
* What new skills will be required?
* Do the existing staff members have these skills? If not, can they betrained in due course of time?

The system is operationally feasible as it very easy for the End users to operate it. It only needs basic information about Windows platform.

**Schedule feasibility**

Time evaluation is the most important consideration in the development of project. The time schedule required for the developed of this project is very important since more development time effect machine time, cost and cause delay in the development of other systems.

**9.5 Summary**

To conclude, we discussed the scheduling processes of developing this system. Additionally we have also identified how feasible the system is through the lens of evaluating using various feasibility studies.

**CHAPTER 10**

**FUTURE WORK**

**10.1 Introduction**

This chapter discusses the future scope or the implementation of this system. As our system is a mobile application based, various forms of new features can be incorporated to this system as per the requirements.

**10.2 Future Scope of Work**

The main objective of developing this system is to provide a basic platform for startup a food ordering service system. The system can be more improved than the current form. In the context of securing this system further work can be done by including firewalls and other means of securing this system. Additionally, the interface of the system can be a work in progress to look more user-friendly. There is a great possibility to scale up the project. This system can be extended for more facilities in future.

**10.3 Summary**

This chapter has described the possible future applications of the system. But there are a lot of possibilities with the designed system. The system may need some research for different applications, though the principle of the designed system will remain as it is.

**CHAPTER 11**

**DESIGN IMPACT**

**11.1 Introduction**

In this chapter, we discuss about the various impacts that our system has been able to generate.

**11.2 Environmental Impact**

By introducing this system in our society amounts of paper can be saved because all the tasks will be computerized therefore it would the number of paper wasted per day in official works can be hugely eradicated.

**11.3 Economic Impact**

As we can see the economy of Bangladesh, we have to think what exactly people want. We have done a simple research on it .First of all we will upload the mobile application in the playstore. Whenever this application will be downloaded more and more then we will get advertisement offer. We can earn money for showing the advertisement in our app.

**11.4 Social Impact**

People can be aware of saving money and time through this application. As a result people will order pizza easily by just sitting in the home. So it will have a great social impact.

**11.5 Sustainability**

Though Bangladesh is a developing county and becoming digitalized day by day but in the beginning of this system people will not take it normally. So in that case we need to make a positive impact in people. So in the beginning we will give free services. After certain time a reasonable amount will be charged for using this system. In this time people will understand the benefits of using this system. So people will be interested to use the system. After that the revenue earned from the system can be invested to maintain the system and cover other costs the keep the system running and up to date. In such a way, the system can be made sustainable and a successful project.

**11.6 Summary**

This chapter has covered the different types of impacts that our system offers and those has been described and discussed. From the above given impacts we can conclude that our designed system is good enough to use under any circumstance.

**CHAPTER 12**

**COMPLIANCE WITH IEEE STANDARDS**

**12.1 Introduction**

In this section we discuss about the consistence of our task with diverse standards. There are a few distinct standards, amongst which the IEEE standards, US standards and European standards are talked about in this part.

**12.2 Compliance with IEEE standard**

There are a couple of unmistakable rules set forward by IEEE Standards association. Most of them anyway are not material for our structure. We have included thought of activity with respect to the IEEE standard. A conference paper has additionally been submitted and affirmed by IEEE standards entitled "Online Food Ordering System" that points of interest out our work on this task.

**12.3. Compliance with US standard**

ANSI recommends that copyrighted programming should just be incorporated for educational purposes, or in structures which don't order specific executions of the standard. Article code ought to never be incorporated into a standard as a regularizing necessity. While ANSI restricts utilization of programming guidelines to order specific executions and trusts that utilization of programming in principles ought to be maintained a strategic distance from to the degree conceivable, ANSI perceives that there might be conditions in which incorporation of some product, if it is joined by satisfactory lawful consents, may encourage advancement of different, contending and interoperable usage of the standard. Instances of such programming could include:

* Pseudo Code (code that is human readable and similar to programming languages but cannot be directly processed or compiled directly to be processed by hardware that manipulates data according to instructions);
* Schema examples;
* Data structure definitions;
* ASN.1 structure definitions;
* ABNF grammar specifications;
* Example programming instructions that are sufficiently limited in scope that they do not, either singularly or in the aggregate, perform a complete or a substantial part of a function and are illustrative, at most, of limited sections of an independent fully described specification; or
* Sample programming instructions provided solely for conformance testing purposes.

Our project has been established based on the above ANSI principles and it completely relies upon it.

**12.4 Summary**

In this area we have inspected the distinctive agreeable measures and ensured that we are as per. These guidelines have been put decisively in order to control things, ensure prosperity and guarantee there are no prosperity perils to the utilization of particular sections. It is vitally fundamental to keep up these measures and we have done in that capacity over the range of our undertaking work.

**CHAPTER 13**

**RESULTS**

**13.1 Introduction**

This chapter of the report contains the results that we achieved throughout the course of using this system.

**13.2 Results Achieved**

From initiation through conclusion of developing this system the following results has been achieved. They are as follows:

* The system can be administered by a non-IT technician.
* The system is ready for market to commercial use.
* The User Interface (UI) is very friendly therefore it would attract more users.

**13.3 Summary**

This chapter has covered the different types of results that we have managed to obtain throughout the course of using this system.

**CHAPTER 14**

**CHALLENGES AND OUTCOMES**

**Challenges**

We have learnt a lot in terms of how difficult it is to please a customer. We got to know the practical knowledge of working of Restaurants: what kind of software is used, how they make process faster at peak hours, how to serve fast and what is the difference between the thoughts of the management and the customers. The factors for success of Pizza Delivery are: value food, developing the local supply chain, customizing product offerings etc.

We have also faced several types of challenges while doing our project. Information sharing is the most critical problem faced by us. We are new to this android related project, for which we faced a lot of challenges related to learning about android studio. And for this project, the main challenge is, many customers are not willing to spare time to fill the questionnaire.

**Outcomes**

Degree of success of this project is to complete the project on time and submit it as it has been planned. One of the successes is to finish it on time and submit it as soon as possible to avoid any delay which will show that it was done from the day we were associated with the task till the time of submission. It will also show that it was done step by step as it was supposed to be.

Initially, we had planned to order both custom-made pizzas along with some non-pizza items. However, due to lack of time, we were just able to implement the custom-made pizza part. Also, we could not implement the variable amount for the various pizza toppings. By now, our project is able to place an order to any restaurant in Bashundhara Area at any time. A further improvement would involve the service for several restaurants all over the city. The system could be improved by enabling the order of other non-pizza items, by implementing the variable amount for pizza toppings and by improving the upgradation of the UI design. Thus, we can come to the statement that we have achieved the maximum project goals and objectives.

**CHAPTER 15**

**CONCLUSION**

This was all about our simple online pizza ordering system. Everyone may know the importance of this system in our day-to-day lives as we all prefer online ordering over ordering on phone.

This system greatly lightens the load on the restaurant’s end as the entire process of taking orders is automated. Once an order is placed on the app, it is entered into the database and then retrieved, in pretty much real-time. The restaurant can quickly go through the orders as they are placed and produce the necessary items with minimal delay and confusion.

This project is not difficult to use, operate and understand by the users. Design of this project is pretty and responsive so that user won’t find it difficult to understand, use and navigate. This project provides the simplest system for managing pizza orders.

**Chapter 16**

**Codes And Programs**

**ConfirmActivity :**

**package** com.example.pizzaorder;  
  
**import** android.app.Activity;  
**import** android.app.ProgressDialog;  
**import** android.content.DialogInterface;  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.support.annotation.NonNull;  
**import** android.view.View;  
**import** android.view.View.OnClickListener;  
**import** android.widget.Button;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.google.firebase.database.DataSnapshot;  
**import** com.google.firebase.database.DatabaseError;  
**import** com.google.firebase.database.DatabaseReference;  
**import** com.google.firebase.database.FirebaseDatabase;  
**import** com.google.firebase.database.ValueEventListener;  
  
**import** java.text.DateFormat;  
**import** java.text.SimpleDateFormat;  
**import** java.util.Calendar;  
  
**public class** ConfirmActivity **extends** Activity {  
  
 **static** String *size*, *crust*, *whole*, *left*, *right*, *phone*, *payment*, *temp*;  
 TextView **cSize**, **cCrust**, **cWhole**, **cLeft**, **cRight**, **cTotal**, **cPhone**, **cPayment**;  
 Button **ConfirmButton**, **BackButton**;  
  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_confirm***);  
  
 *size* = getIntent().getStringExtra(**"Size"**);  
 *crust* = getIntent().getStringExtra(**"Crust"**);  
 *whole* = getIntent().getStringExtra(**"Whole"**);  
 *left* = getIntent().getStringExtra(**"Left"**);  
 *right* = getIntent().getStringExtra(**"Right"**);  
 *phone* = getIntent().getStringExtra(**"Phone"**);  
 *payment* = getIntent().getStringExtra(**"Payment"**);  
  
 **ConfirmButton** = (Button) findViewById(R.id.***confirm***);  
 **BackButton** = (Button) findViewById(R.id.***back***);  
 **cSize** = (TextView) findViewById(R.id.***size***);  
 **cCrust** = (TextView) findViewById(R.id.***crust***);  
 **cWhole** = (TextView) findViewById(R.id.***whole***);  
 **cLeft** = (TextView) findViewById(R.id.***left***);  
 **cRight** = (TextView) findViewById(R.id.***right***);  
 **cTotal** = (TextView) findViewById(R.id.***total***);  
 **cPhone** = (TextView) findViewById(R.id.***cphone***);  
 **cPayment** = (TextView) findViewById(R.id.***payment***);  
  
 **cSize**.setText(*size*);  
 **cCrust**.setText(*crust*);  
 **cWhole**.setText(*whole*);  
 **cLeft**.setText(*left*);  
 **cRight**.setText(*right*);  
 **if**(*size*.equals(**"Small"**))  
 *temp* = **"400tk"**;  
 **else if**(*size*.equals(**"Medium"**))  
 *temp* = **"650Tk"**;  
 **else if**(*size*.equals(**"Large"**))  
 *temp* = **"800Tk"**;  
 **cTotal**.setText(**"Total amount: "** + *temp*);  
 **cPhone**.setText(*phone*);  
 **cPayment**.setText(*payment*);  
  
 **final** ProgressDialog dialog = **new** ProgressDialog(**this**);  
 **final** FirebaseDatabase database = FirebaseDatabase.*getInstance*();  
 **final** DatabaseReference myRef = database.getReference(**"order"**);  
  
 **ConfirmButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 dialog.setMessage(**"Please wait..."**);  
 dialog.setIndeterminate(**true**);  
 dialog.show();  
  
 myRef.addValueEventListener(**new** ValueEventListener() {  
  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot) {  
  
 dialog.dismiss();  
 DateFormat df = **new** SimpleDateFormat(**"EEE, d MMM yyyy, HH:mm"**);  
 String dateT = df.format(Calendar.*getInstance*().getTime());  
 Order order = **new** Order(*size*, *crust*, *whole*, *left*, *right*, *temp*, *payment*, dateT);  
 myRef.child(*phone*).setValue(order);  
 Toast.*makeText*(ConfirmActivity.**this**, **"Confirmed ! It will take 25minutes to deliver !"**, Toast.***LENGTH\_SHORT***).show();  
 Intent intent = **new** Intent(ConfirmActivity.**this**, HomeActivity.**class**);  
 startActivity(intent);  
 finish();  
 }  
  
 @Override  
 **public void** onCancelled(@NonNull DatabaseError databaseError) {  
  
 }  
 });  
 }  
 });  
 **BackButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 Intent intent = **new** Intent(ConfirmActivity.**this**, PizzaInfoActivity.**class**);  
 startActivity(intent);  
 finish();  
 }  
 });  
 }  
}

**HomeActivity :**

**package** com.example.pizzaorder;  
  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.TextView;  
  
**import** com.example.pizzaorder.common.Common;  
**import** com.google.firebase.FirebaseApp;  
**import** com.google.firebase.database.DatabaseReference;  
**import** com.google.firebase.database.FirebaseDatabase;  
  
**public class** HomeActivity **extends** AppCompatActivity {  
  
 TextView **Name**;  
 Button **ProfileButton**, **OrderButton**, **LogOutButton**;  
 FirebaseDatabase **database**;  
 DatabaseReference **myRef**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_home***);  
  
 FirebaseApp.*initializeApp*(**this**);  
  
 **database** = FirebaseDatabase.*getInstance*();  
 **myRef** = **database**.getReference(**"user"**);  
  
 **final** String uPhone = getIntent().getStringExtra(**"userPhone"**);  
  
 **ProfileButton** = (Button) findViewById(R.id.***profile***);  
 **LogOutButton** = (Button) findViewById(R.id.***logout***);  
 **OrderButton** = (Button) findViewById(R.id.***order***);  
 **Name** = (TextView) findViewById(R.id.***name***);  
 **Name**.setText(Common.*currentUser*.getName());  
  
 **ProfileButton**.setOnClickListener(**new** View.OnClickListener() {  
  
 **public void** onClick (View view){  
  
 Intent intent = **new** Intent(HomeActivity.**this**, ProfileActivity.**class**);  
 intent.putExtra(**"userPhone"**, uPhone);  
 startActivity(intent);  
 }  
 });  
  
 **OrderButton**.setOnClickListener(**new** View.OnClickListener() {  
  
 **public void** onClick (View view){  
  
 Intent intent = **new** Intent(HomeActivity.**this**, OrderActivity.**class**);  
 intent.putExtra(**"userPhone"**, uPhone);  
 startActivity(intent);  
 }  
 });  
  
 **LogOutButton**.setOnClickListener(**new** View.OnClickListener() {  
  
 **public void** onClick (View view){  
  
 Intent intent = **new** Intent(HomeActivity.**this**, MainActivity.**class**);  
 startActivity(intent);  
 finish();  
 }  
 });  
 }  
}

**ImageAdapter :**

**package** com.example.pizzaorder;  
**import** android.content.Context;  
**import** android.content.res.TypedArray;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.BaseAdapter;  
**import** android.widget.Gallery;  
**import** android.widget.ImageView;  
  
**public class** ImageAdapter **extends** BaseAdapter {  
 **int mGalleryItemBackground**;  
 **private** Context **mContext**;  
  
 **private** Integer[] **mImageIds** = {  
 R.drawable.anchovies,  
 R.drawable.bacon,  
 R.drawable.bananapepper,  
 R.drawable.blackolives,  
 R.drawable.chicken,  
 R.drawable.greenpeppers,  
 R.drawable.ham,  
 R.drawable.jalapenopeppers,  
 R.drawable.mozzarella,  
 R.drawable.mushrooms,  
 R.drawable.onion,  
 R.drawable.pepperoni,  
 R.drawable.pineapple,  
 R.drawable.sausage,  
 R.drawable.tomatoes,  
 };  
  
 **public** ImageAdapter(Context c) {  
 mContext = c;  
 TypedArray a = c.obtainStyledAttributes(R.styleable.HelloGallery);  
 mGalleryItemBackground = a.getResourceId(  
 R.styleable.HelloGallery\_android\_galleryItemBackground, 0);  
 a.recycle();  
 }  
  
 **public int** getCount() {  
 **return** mImageIds.length;  
 }  
  
 **public** Object getItem(**int** position) {  
 **return** position;  
 }  
  
 **public long** getItemId(**int** position) {  
 **return** position;  
 }  
  
 **public** View getView(**int** position, View convertView, ViewGroup parent) {  
 ImageView i = **new** ImageView(mContext);  
  
 i.setImageResource(mImageIds[position]);  
 i.setLayoutParams(**new** Gallery.LayoutParams(400, 400));  
 i.setScaleType(ImageView.ScaleType.FIT\_XY);  
 i.setBackgroundResource(mGalleryItemBackground);  
 **return** i;}}

**MainActivity :**

**package** com.example.pizzaorder;  
  
**import** android.support.annotation.NonNull;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.app.ProgressDialog;  
**import** android.content.Intent;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
  
**import** com.example.pizzaorder.common.Common;  
**import** com.google.firebase.FirebaseApp;  
**import** com.google.firebase.FirebaseError;  
**import** com.google.firebase.auth.FirebaseAuth;  
**import** com.google.firebase.database.DataSnapshot;  
**import** com.google.firebase.database.DatabaseError;  
**import** com.google.firebase.database.DatabaseReference;  
**import** com.google.firebase.database.FirebaseDatabase;  
**import** com.google.firebase.database.ValueEventListener;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 EditText **Phone**, **Password**;  
 Button **LogInButton**, **RegisterButton**;  
 ProgressDialog **dialog**;  
  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 FirebaseApp.*initializeApp*(**this**);  
 setContentView(R.layout.***activity\_main***);  
  
 **LogInButton** = (Button) findViewById(R.id.***login***);  
 **RegisterButton** = (Button) findViewById(R.id.***register***);  
 **Phone** = (EditText) findViewById(R.id.***phone***);  
 **Password** = (EditText) findViewById(R.id.***password***);  
  
 **final** ProgressDialog dialog = **new** ProgressDialog(**this**);  
  
 **final** FirebaseDatabase database = FirebaseDatabase.*getInstance*();  
 **final** DatabaseReference myRef = database.getReference(**"user"**);  
  
 **LogInButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 dialog.setMessage(**"Loging in please wait..."**);  
 dialog.setIndeterminate(**true**);  
 dialog.show();  
  
 myRef.addValueEventListener(**new** ValueEventListener() {  
  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot) {  
  
 **if** (dataSnapshot.child(**Phone**.getText().toString()).exists()) {  
 dialog.dismiss();  
 User user = dataSnapshot.child(**Phone**.getText().toString()).getValue(User.**class**);  
 **if** (user.getPassword().equals(**Password**.getText().toString())) {  
  
 Intent intent = **new** Intent(MainActivity.**this**, HomeActivity.**class**);  
 intent.putExtra(**"userPhone"**, **Phone**.getText().toString());  
 Common.*currentUser* = user;  
 startActivity(intent);  
 finish();  
  
 } **else** {  
 Toast.*makeText*(MainActivity.**this**, **"Wrong password !"**, Toast.***LENGTH\_SHORT***).show();  
 **Phone**.getText().clear();  
 **Password**.getText().clear();  
 }  
 } **else** {  
 dialog.dismiss();  
 Toast.*makeText*(MainActivity.**this**, **"Phone number does not exists !"**, Toast.***LENGTH\_SHORT***).show();  
 **Phone**.getText().clear();  
 **Password**.getText().clear();  
 }  
 }  
  
 @Override  
 **public void** onCancelled(@NonNull DatabaseError databaseError) {  
  
 }  
 });  
 }  
 });  
 **RegisterButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 Intent intent = **new** Intent(MainActivity.**this**, RegisterActivity.**class**);  
 startActivity(intent);  
  
 }  
 });  
 }  
}

**NewPizzaActivity :**

**package** com.example.pizzaorder;  
**import static** android.provider.BaseColumns.***\_ID***;  
**import static** com.example.pizzaorder.PizzaData.***SIZE***;  
**import static** com.example.pizzaorder.PizzaData.***TABLE\_NAME***;  
**import static** com.example.pizzaorder.PizzaData.***TOPPINGS\_LEFT***;  
**import static** com.example.pizzaorder.PizzaData.***TOPPINGS\_RIGHT***;  
**import static** com.example.pizzaorder.PizzaData.***TOPPINGS\_WHOLE***;  
  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
**import** android.app.Activity;  
**import** android.app.AlertDialog;  
**import** android.content.ContentValues;  
**import** android.content.DialogInterface;  
**import** android.content.Intent;  
**import** android.database.Cursor;  
**import** android.os.Bundle;  
**import** android.view.Menu;  
**import** android.view.MenuInflater;  
**import** android.view.MenuItem;  
**import** android.view.View;  
**import** android.view.View.OnClickListener;  
**import** android.widget.AdapterView;  
**import** android.widget.AdapterView.OnItemClickListener;  
**import** android.widget.Gallery;  
**import** android.widget.RadioButton;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.example.pizzaorder.R;  
  
**public class** NewPizzaActivity **extends** Activity **implements** OnClickListener {  
  
 **public** RadioButton **wholeRadio**;  
 **public** RadioButton **leftRadio**;  
 **public** RadioButton **rightRadio**;  
 **public** TextView **wholeText**;  
 **public** TextView **leftText**;  
 **public** TextView **rightText**;  
 **public** ArrayList<String> **wList** = **new** ArrayList<String>();  
 **public** ArrayList<String> **lList** = **new** ArrayList<String>();  
 **public** ArrayList<String> **rList** = **new** ArrayList<String>();  
 **public** ArrayList<String> **topingList** = **new** ArrayList<String>();  
 **int id** = 999;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_new\_pizza***);  
 createToppingList();  
  
 Bundle bundle = **this**.getIntent().getExtras();  
 **if** (bundle != **null** && bundle.size() > 0)  
 **id** = bundle.getInt(***\_ID***);  
  
 **wholeRadio** = (RadioButton) findViewById(R.id.***whole***);  
 **leftRadio** = (RadioButton) findViewById(R.id.***left***);  
 **rightRadio** = (RadioButton) findViewById(R.id.***right***);  
 **wholeText** = (TextView) findViewById(R.id.***whole\_text***);  
 **leftText** = (TextView) findViewById(R.id.***left\_text***);  
 **rightText** = (TextView) findViewById(R.id.***right\_text***);  
  
  
 Gallery gallery = (Gallery) findViewById(R.id.***gallery***);  
 gallery.setAdapter(**new** ImageAdapter(**this**));  
  
 gallery.setOnItemClickListener(**new** OnItemClickListener() {  
 @SuppressWarnings(**"rawtypes"**)  
 **public void** onItemClick(AdapterView parent, View v, **int** position, **long** id) {  
 **if** (**wList**.isEmpty())  
 **wList**.clear();  
 **if** (**lList**.isEmpty())  
 **lList**.clear();  
 **if** (**rList**.isEmpty())  
 **rList**.clear();  
 **if** (**wholeRadio**.isChecked()) { *// Whole Pizza* **if** (**wList**.contains(**topingList**.get(position))) {  
 displayMessage(position, **" removed"**);  
 **wList**.remove(**wList**.indexOf(**topingList**.get(position)));  
 **if** (**wList**.isEmpty())  
 **wholeText**.setText(**""**);  
 **else  
 wholeText**.setText(editString(**wList**));  
 } **else if** (**rList**.contains(**topingList**.get(position))) {  
 **rList**.remove(**rList**.indexOf(**topingList**.get(position)));  
 **wList**.add(**topingList**.get(position));  
 **rightText**.setText(editString(**rList**));  
 **wholeText**.setText(editString(**wList**));  
 displayMessage(position, **" added"**);  
 } **else if** (**lList**.contains(**topingList**.get(position))) {  
 **lList**.remove(**lList**.indexOf(**topingList**.get(position)));  
 **wList**.add(**topingList**.get(position));  
 **leftText**.setText(editString(**lList**));  
 **wholeText**.setText(editString(**wList**));  
 displayMessage(position, **" added"**);  
 } **else** {  
 displayMessage(position, **" added"**);  
 **wList**.add(**topingList**.get(position));  
 **wholeText**.setText(editString(**wList**));  
 }  
 } **else if** (**leftRadio**.isChecked()) { *// Left side of pizza.* **if** (**lList**.contains(**topingList**.get(position))) {  
 displayMessage(position, **" removed"**);  
 **lList**.remove(**lList**.indexOf(**topingList**.get(position)));  
 **if** (**lList**.isEmpty())  
 **leftText**.setText(**""**);  
 **else  
 leftText**.setText(editString(**lList**));  
 } **else if** (**rList**.contains(**topingList**.get(position))) { *// Checking if the same topping is on the right side of pizza.* displayMessage(position, **" added to whole pizza"**);  
 **rList**.remove(**rList**.indexOf(**topingList**.get(position)));  
 **rightText**.setText(editString(**rList**));  
 **wList**.add(**topingList**.get(position));  
 **wholeText**.setText(editString(**wList**));  
 } **else if** (**wList**.contains(**topingList**.get(position))) {  
 displayMessage(position, **" have already been added to the whole pizza"**);  
 } **else** {  
 displayMessage(position, **" added"**);  
 **lList**.add(**topingList**.get(position));  
 **leftText**.setText(editString(**lList**));  
 }  
 } **else** { *// Right side of pizza.* **if** (**rList**.contains(**topingList**.get(position))) {  
 displayMessage(position, **" removed"**);  
 **rList**.remove(**rList**.indexOf(**topingList**.get(position)));  
 **if** (**rList**.isEmpty())  
 **rightText**.setText(**""**);  
 **else  
 rightText**.setText(editString(**rList**));  
 } **else if** (**lList**.contains(**topingList**.get(position))) { *// Checking if the same topping is on the left side of pizza.* displayMessage(position, **" added to whole pizza"**);  
 **lList**.remove(**lList**.indexOf(**topingList**.get(position)));  
 **leftText**.setText(editString(**lList**));  
 **wList**.add(**topingList**.get(position));  
 **wholeText**.setText(editString(**wList**));  
 } **else if** (**wList**.contains(**topingList**.get(position))) {  
 displayMessage(position, **" have already been added to the whole pizza"**);  
 } **else** {  
 displayMessage(position, **" added"**);  
 **rList**.add(**topingList**.get(position));  
 **rightText**.setText(editString(**rList**));  
 }  
 }  
 }  
  
 });  
  
 View AddToCartButton = findViewById(R.id.***add\_to\_cart\_button***);  
 AddToCartButton.setOnClickListener(**this**);  
 View CancelButton = findViewById(R.id.***cancel\_button***);  
 CancelButton.setOnClickListener(**this**);  
 }  
  
 @Override  
 **public void** onClick(View v) {  
 **switch** (v.getId()) {  
 **case** R.id.***add\_to\_cart\_button***:  
 String toppingsWhole = **""**;  
 String toppingsLeft = **""**;  
 String toppingsRight = **""**;  
 **if** (**wList**.size() > 0) {  
 toppingsWhole = toppingsWhole + editString(**wList**);  
 }  
 **if** (**lList**.size() > 0) {  
 toppingsLeft = toppingsLeft + editString(**lList**);  
 }  
 **if** (**rList**.size() > 0) {  
 toppingsRight = toppingsRight + editString(**rList**);  
 }  
 **final** String size = getIntent().getStringExtra(**"Size"**);  
 **final** String crust = getIntent().getStringExtra(**"Crust"**);  
 **final** String phone = getIntent().getStringExtra(**"Phone"**);  
 Intent intent = **new** Intent(NewPizzaActivity.**this**, PizzaInfoActivity.**class**);  
 intent.putExtra(**"pSize"**, size);  
 intent.putExtra(**"pCrust"**, crust);  
 intent.putExtra(**"pWhole"**, toppingsWhole);  
 intent.putExtra(**"pLeft"**, toppingsLeft);  
 intent.putExtra(**"pRight"**, toppingsRight);  
 intent.putExtra(**"uPhone"**, phone);  
 startActivity(intent);  
 **break**;  
 **case** R.id.***cancel\_button***:  
 Intent intent1 = **new** Intent(NewPizzaActivity.**this**, OrderActivity.**class**);  
 startActivity(intent1);  
 finish();  
 **break**;  
 }  
  
 }  
  
 **private void** createToppingList() {  
 **topingList**.add(**"Anchovies"**);  
 **topingList**.add(**"Bacon"**);  
 **topingList**.add(**"Banana Peppers"**);  
 **topingList**.add(**"Black Olives"**);  
 **topingList**.add(**"Chicken"**);  
 **topingList**.add(**"Green Peppers"**);  
 **topingList**.add(**"Ham"**);  
 **topingList**.add(**"Jalapeno Peppers"**);  
 **topingList**.add(**"Extra Cheese"**);  
 **topingList**.add(**"Mushrooms"**);  
 **topingList**.add(**"Onion"**);  
 **topingList**.add(**"Pepperoni"**);  
 **topingList**.add(**"Pineapple"**);  
 **topingList**.add(**"Sausage"**);  
 **topingList**.add(**"Roma Tomatoes"**);  
 }  
  
 **private void** displayMessage(**int** position, String message) {  
 Toast.*makeText*(NewPizzaActivity.**this**, **topingList**.get(position) + message, Toast.***LENGTH\_SHORT***).show();  
 }  
  
 **private** String editString(List<String> list) {  
 String toppings = **""**;  
 String withOutComma;  
 **for** (String item : list) {  
 toppings += item + **", "**;  
 }  
 **if** (toppings.equals(**""**))  
 withOutComma = **""**;  
 **else** withOutComma = toppings.substring(0, toppings.length() - 2);  
 **return** withOutComma;  
 }  
  
 **private void** setArrayList(ArrayList<String> list, String[] toppings) {  
 **for** (**int** i = 0; i < toppings.**length**; i++) {  
 list.add(toppings[i].trim());  
 }  
 }  
  
}

**Order :**

**package** com.example.pizzaorder;  
**public class** Order {  
 **private** String **Size**;  
 **private** String **Crust**;  
 **private** String **WholeTop**;  
 **private** String **LeftTop**;  
 **private** String **RightTop**;  
 **private** String **Payment**;  
 **private** String **Total**;  
 **private** String **Time**;  
  
 **public** Order(String size, String crust, String whole, String left, String right, String total, String payment, String time) {  
 **Size** = size;  
 **Crust** = crust;  
 **WholeTop** = whole;  
 **LeftTop** = left;  
 **RightTop** = right;  
 **Total** = total;  
 **Payment** = payment;  
 **Time** = time;  
 }  
  
  
 **public void** setTime(String time) {  
 **Time** = time;  
 }  
  
  
 **public** String getTime() {  
 **return Time**;  
 }  
  
 **public** String getSize() {  
 **return Size**;  
 }  
  
 **public** String getCrust() {  
 **return Crust**;  
 }  
  
 **public** String getWholeTop() {  
 **return WholeTop**;  
 }  
  
 **public** String getLeftTop() {  
 **return LeftTop**;  
 }  
  
 **public** String getRightTop() {  
 **return RightTop**;  
 }  
  
 **public** String getPayment() {  
 **return Payment**;  
 }  
  
 **public** String getTotal() {  
 **return Total**;  
 }  
  
 **public void** setSize(String size) {  
 **Size** = size;  
 }  
  
 **public void** setCrust(String crust) {  
 **Crust** = crust;  
 }  
  
 **public void** setWholeTop(String wholeTop) {  
 **WholeTop** = wholeTop;  
 }  
  
 **public void** setLeftTop(String leftTop) {  
 **LeftTop** = leftTop;  
 }  
  
 **public void** setRightTop(String rightTop) {  
 **RightTop** = rightTop;  
 }  
  
 **public void** setPayment(String payment) {  
 **Payment** = payment;  
 }  
  
 **public void** setTotal(String total) {  
 **Total** = total;  
 }  
  
 **public** Order() {  
  
 }  
  
}

**OrderActivity :**

**package** com.example.pizzaorder;  
**import** android.app.AlertDialog;  
**import** android.app.ListActivity;  
**import** android.content.DialogInterface;  
**import** android.content.Intent;  
**import** android.database.Cursor;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.AdapterView;  
**import** android.widget.TextView;  
  
**public class** OrderActivity **extends** ListActivity **implements** View.OnClickListener{  
  
 **public boolean hasShown** = **false**;  
 **public** TextView **Total**;  
 **static** String *usPhone*;  
 @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_order***);  
 getListView().setChoiceMode(1);  
  
  
 *usPhone* = getIntent().getStringExtra(**"userPhone"**);  
 getListView().setOnItemClickListener(**new** AdapterView.OnItemClickListener() {  
  
 **public void** onNothingSelected(AdapterView arg0) {  
  
 }  
  
 **public void** onItemClick(AdapterView arg0, View arg1, **int** arg2, **long** arg3) {  
  
 }  
  
 });  
  
 View getNewPizzaButton = findViewById(R.id.***newPizza***);  
 getNewPizzaButton.setOnClickListener(**this**);  
 View getCheckoutButton = findViewById(R.id.***checkout***);  
 getCheckoutButton.setOnClickListener(**this**);  
 **Total** = (TextView) findViewById(R.id.***total***);  
 }  
  
 @Override  
 **public void** onClick(View v) {  
  
 **switch** (v.getId()) {  
 **case** R.id.***newPizza***:  
 **if** (!**hasShown**) {  
 openHowToDialog();  
 **hasShown** = **true**;  
 } **else** {  
 openSizeSelectionDialog();  
 }  
 **break**;  
 **case** R.id.***checkout***:  
 **if** (getListView().getCount() != 0)  
 checkOutDialog();  
 **else** {  
 **new** AlertDialog.Builder(**this**).setTitle(**"Info"**).setMessage(**"You must order a pizza, before you can checkout."**).setCancelable(**false**)  
 .setNeutralButton(**"OK"**, **new** DialogInterface.OnClickListener() {  
  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 openSizeSelectionDialog();  
 }  
 }).show();  
 }  
 **break**;  
 }  
  
 }  
  
 **public void** openHowToDialog() {  
 **new** AlertDialog.Builder(**this**).setTitle(R.string.***how\_to\_title***).setMessage(R.string.***how\_to\_text***).setCancelable(**false**)  
 .setNeutralButton(**"OK"**, **new** DialogInterface.OnClickListener() {  
  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 openSizeSelectionDialog();  
 }  
 }).show();  
 }  
  
 **public void** openSizeSelectionDialog() {  
 **new** AlertDialog.Builder(**this**).setTitle(R.string.***pizza\_size***).setItems(R.array.***pizza\_size***, **new** DialogInterface.OnClickListener() {  
  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 String size;  
 **if** (which == 0)  
 size = **"Small"**;  
 **else if** (which == 1)  
 size = **"Medium"**;  
 **else** size = **"Large"**;  
 openCrustSelectionDialog(size);  
 }  
 }).show();  
 }  
  
 **public void** openCrustSelectionDialog(**final** String size) {  
 **new** AlertDialog.Builder(**this**).setTitle(R.string.***crust\_selection***).setItems(R.array.***pizza\_crust***, **new** DialogInterface.OnClickListener() {  
  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 String crust;  
 **if** (which == 0)  
 crust = **"Thin"**;  
 **else if** (which == 1)  
 crust = **"Thick"**;  
 **else if** (which == 2)  
 crust = **"Deep"**;  
 **else** crust = **"Stuffed"**;  
  
 String sz = size;  
 Intent intent = **new** Intent(OrderActivity.**this**, NewPizzaActivity.**class**);  
 intent.putExtra(**"Size"**, sz);  
 intent.putExtra(**"Crust"**, crust);  
 intent.putExtra(**"Phone"**, *usPhone*);  
 startActivity(intent);  
  
 }  
 }).show();  
 }  
  
 **public void** checkOutDialog() {  
  
 }  
}

**PizzaData:**

**package** com.example.pizzaorder;  
  
**import** android.provider.BaseColumns;  
  
**public interface** PizzaData **extends** BaseColumns {  
 **public static final** String ***TABLE\_NAME*** = **"pizza"**;  
  
 *// Columns in the Events database* **public static final** String ***SIZE*** = **"size"**;  
 **public static final** String ***CRUST*** = **"crust"**;  
 **public static final** String ***TOPPINGS\_WHOLE*** = **"toppingsWhole"**;  
 **public static final** String ***TOPPINGS\_LEFT*** = **"toppingsLeft"**;  
 **public static final** String ***TOPPINGS\_RIGHT*** = **"toppingsRight"**;  
}

**PizzaInfoActivity :**

**package** com.example.pizzaorder;  
**import** android.app.Activity;  
**import** android.app.AlertDialog;  
**import** android.content.DialogInterface;  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.text.Editable;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.View.OnClickListener;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.TextView;  
  
**import** java.util.ArrayList;  
  
**public class** PizzaInfoActivity **extends** Activity {  
  
 TextView **Size**, **Crust**, **Whole**, **Left**, **Right**, **Total**;  
 Button **PayButton**, **BackButton**;  
 **static** String *ppsize*, *ppcrust*, *ppwhole*, *ppleft*, *ppright*, *uuphone*, *paym*;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_pizza\_info***);  
  
 *ppsize* = getIntent().getStringExtra(**"pSize"**);  
 *ppcrust* = getIntent().getStringExtra(**"pCrust"**);  
 *ppwhole* = getIntent().getStringExtra(**"pWhole"**);  
 *ppleft* = getIntent().getStringExtra(**"pLeft"**);  
 *ppright* = getIntent().getStringExtra(**"pRight"**);  
 *uuphone* = getIntent().getStringExtra(**"uPhone"**);  
  
 **PayButton** = (Button) findViewById(R.id.***payM***);  
 **BackButton** = (Button) findViewById(R.id.***cancel***);  
 **Size** = (TextView) findViewById(R.id.***size***);  
 **Crust** = (TextView) findViewById(R.id.***crust***);  
 **Whole** = (TextView) findViewById(R.id.***whole***);  
 **Left** = (TextView) findViewById(R.id.***left***);  
 **Right** = (TextView) findViewById(R.id.***right***);  
 **Total** = (TextView) findViewById(R.id.***stotal***);  
  
 **Size**.setText(*ppsize*);  
 **Crust**.setText(*ppcrust*);  
 **Whole**.setText(*ppwhole*);  
 **Left**.setText(*ppleft*);  
 **Right**.setText(*ppright*);  
 String temp = **null**;  
 **if**(*ppsize*.equals(**"Small"**))  
 temp = **"400tk"**;  
 **else if**(*ppsize*.equals(**"Medium"**))  
 temp = **"650Tk"**;  
 **else if**(*ppsize*.equals(**"Large"**))  
 temp = **"800Tk"**;  
 **Total**.setText(**"Total amount: "** + temp);  
 **PayButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 AlertDialog.Builder alertDialogBuilder = **new** AlertDialog.Builder(PizzaInfoActivity.**this**);  
 alertDialogBuilder.setTitle(**"Select Payment Method"**).setItems(R.array.***payment***, **new** DialogInterface.OnClickListener() {  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 **if**(which == 0)  
 bkashTxnDialog();  
 **else** {  
 *paym* = **"Cash On Delivery"**;  
 Intent intent = **new** Intent(PizzaInfoActivity.**this**, ConfirmActivity.**class**);  
 intent.putExtra(**"Size"**, *ppsize*);  
 intent.putExtra(**"Crust"**, *ppcrust*);  
 intent.putExtra(**"Whole"**, *ppwhole*);  
 intent.putExtra(**"Left"**, *ppleft*);  
 intent.putExtra(**"Right"**, *ppright*);  
 intent.putExtra(**"Phone"**, *uuphone*);  
 intent.putExtra(**"Payment"**, *paym*);  
 startActivity(intent);  
 finish();  
 }  
 }  
 }).show();  
  
 }  
  
 **private void** bkashTxnDialog() {  
 LayoutInflater li = LayoutInflater.*from*(PizzaInfoActivity.**this**);  
 View promptsView = li.inflate(R.layout.***prompts***, **null**);  
  
 AlertDialog.Builder alertDialogBuilder = **new** AlertDialog.Builder(PizzaInfoActivity.**this**);  
  
 alertDialogBuilder.setView(promptsView);  
  
 **final** EditText userInput = (EditText) promptsView  
 .findViewById(R.id.***editText***);  
  
 alertDialogBuilder.setCancelable(**false**);  
 alertDialogBuilder.setPositiveButton(**"OK"**,  
 **new** DialogInterface.OnClickListener() {  
 **public void** onClick(DialogInterface dialog, **int** id) {  
 *paym* = **"Bkash TxnId: "** + userInput.getText().toString();  
 Intent intent = **new** Intent(PizzaInfoActivity.**this**, ConfirmActivity.**class**);  
 intent.putExtra(**"Size"**, *ppsize*);  
 intent.putExtra(**"Crust"**, *ppcrust*);  
 intent.putExtra(**"Whole"**, *ppwhole*);  
 intent.putExtra(**"Left"**, *ppleft*);  
 intent.putExtra(**"Right"**, *ppright*);  
 intent.putExtra(**"Phone"**, *uuphone*);  
 intent.putExtra(**"Payment"**, *paym*);  
 startActivity(intent);  
 finish();  
 }  
 });  
 alertDialogBuilder.setNegativeButton(**"Cancel"**,  
 **new** DialogInterface.OnClickListener() {  
 **public void** onClick(DialogInterface dialog, **int** id) {  
 dialog.cancel();  
 }  
 });  
  
 AlertDialog alertDialog = alertDialogBuilder.create();  
  
 alertDialog.show();  
 }  
 });  
  
 }  
}

**ProfileActivity :**

**package** com.example.pizzaorder;  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.TextView;  
  
**import** com.example.pizzaorder.common.Common;  
**import** com.google.firebase.FirebaseApp;  
**import** com.google.firebase.database.DatabaseReference;  
**import** com.google.firebase.database.FirebaseDatabase;  
  
**public class** ProfileActivity **extends** AppCompatActivity {  
  
 TextView **Name**, **Phone**, **Address**, **House**, **Road**, **Block**;  
 Button **EditButton**, **BackButton**;  
 FirebaseDatabase **database**;  
 DatabaseReference **myRef**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_profile***);  
  
 FirebaseApp.*initializeApp*(**this**);  
  
 **database** = FirebaseDatabase.*getInstance*();  
 **myRef** = **database**.getReference(**"user"**);  
  
 **final** String usPhone = getIntent().getStringExtra(**"userPhone"**);  
  
 **EditButton** = (Button) findViewById(R.id.***edit***);  
 **BackButton** = (Button) findViewById(R.id.***back***);  
 **Name** = (TextView) findViewById(R.id.***name***);  
 **Phone** = (TextView) findViewById(R.id.***phone***);  
 **Address** = (TextView) findViewById(R.id.***address***);  
 **House** = (TextView) findViewById(R.id.***house***);  
 **Road** = (TextView) findViewById(R.id.***road***);  
 **Block** = (TextView) findViewById(R.id.***block***);  
  
 **Name**.setText(Common.*currentUser*.getName());  
 **Phone**.setText(usPhone);  
 **Address**.setText(Common.*currentUser*.getAddress());  
 **House**.setText(Common.*currentUser*.getHouse());  
 **Road**.setText(Common.*currentUser*.getRoad());  
 **Block**.setText(Common.*currentUser*.getBlock());  
  
 **EditButton**.setOnClickListener(**new** View.OnClickListener() {  
  
 **public void** onClick (View view){  
  
 Intent intent = **new** Intent(ProfileActivity.**this**, ProfileEditActivity.**class**);  
 intent.putExtra(**"userPhone"**, usPhone);  
 startActivity(intent);  
  
 }  
 });  
  
 **BackButton**.setOnClickListener(**new** View.OnClickListener() {  
  
 **public void** onClick (View view){  
  
 Intent intent = **new** Intent(ProfileActivity.**this**, HomeActivity.**class**);  
 intent.putExtra(**"userPhone"**, usPhone);  
 startActivity(intent);  
 }  
 });  
  
 }  
}

**ProfileEditActivity :**

**package** com.example.pizzaorder;  
  
**import** android.app.ProgressDialog;  
**import** android.content.Intent;  
**import** android.support.annotation.NonNull;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
  
**import** com.example.pizzaorder.common.Common;  
**import** com.google.firebase.FirebaseApp;  
**import** com.google.firebase.database.DataSnapshot;  
**import** com.google.firebase.database.DatabaseError;  
**import** com.google.firebase.database.DatabaseReference;  
**import** com.google.firebase.database.FirebaseDatabase;  
**import** com.google.firebase.database.ValueEventListener;  
  
**public class** ProfileEditActivity **extends** AppCompatActivity {  
  
 EditText **Name**, **Phone**, **Address**, **House**, **Road**, **Block**, **Pass**;  
 Button **UpdateButton**, **BackButton**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_profile\_edit***);  
  
 FirebaseApp.*initializeApp*(**this**);  
  
 **final** ProgressDialog dialog = **new** ProgressDialog(**this**);  
 FirebaseDatabase database = FirebaseDatabase.*getInstance*();  
 **final** DatabaseReference myRef = database.getReference(**"user"**);  
 **final** String urPhone = getIntent().getStringExtra(**"userPhone"**);  
  
 **UpdateButton** = (Button) findViewById(R.id.***update***);  
 **BackButton** = (Button) findViewById(R.id.***cancel***);  
 **Name** = (EditText) findViewById(R.id.***name***);  
 **Phone** = (EditText) findViewById(R.id.***phone***);  
 **Address** = (EditText) findViewById(R.id.***address***);  
 **House** = (EditText) findViewById(R.id.***house***);  
 **Road** = (EditText) findViewById(R.id.***road***);  
 **Block** = (EditText) findViewById(R.id.***block***);  
 **Pass** = (EditText) findViewById(R.id.***pass***);  
  
 **Name**.setText(Common.*currentUser*.getName());  
 **Phone**.setText(urPhone);  
 **Address**.setText(Common.*currentUser*.getAddress());  
 **House**.setText(Common.*currentUser*.getHouse());  
 **Road**.setText(Common.*currentUser*.getRoad());  
 **Block**.setText(Common.*currentUser*.getBlock());  
 **Pass**.setText(Common.*currentUser*.getPassword());  
  
 **UpdateButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 dialog.setMessage(**"Please wait..."**);  
 dialog.setIndeterminate(**true**);  
 dialog.show();  
  
 myRef.addValueEventListener(**new** ValueEventListener() {  
  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot) {  
  
 **if** (dataSnapshot.child(**Phone**.getText().toString()).exists()) {  
  
 dialog.dismiss();  
 User user = **new** User(**Address**.getText().toString(), **Block**.getText().toString(), **House**.getText().toString(), **Name**.getText().toString(), **Pass**.getText().toString(), **Road**.getText().toString());  
 myRef.child(**Phone**.getText().toString()).setValue(user);  
 Toast.*makeText*(ProfileEditActivity.**this**, **"Successful !"**, Toast.***LENGTH\_SHORT***).show();  
 Intent intent = **new** Intent(ProfileEditActivity.**this**, ProfileActivity.**class**);  
 intent.putExtra(**"userPhone"**, urPhone);  
 startActivity(intent);  
 }  
 }  
  
 @Override  
 **public void** onCancelled(@NonNull DatabaseError databaseError) {  
  
 }  
 });  
 }  
 });  
  
 **BackButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 Intent intent = **new** Intent(ProfileEditActivity.**this**, ProfileActivity.**class**);  
 intent.putExtra(**"userPhone"**, urPhone);  
 startActivity(intent);  
  
 }  
 });  
 }  
}

**RegisterActivity :**

**package** com.example.pizzaorder;  
  
**import** android.app.ProgressDialog;  
**import** android.content.Intent;  
**import** android.support.annotation.NonNull;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
  
**import** com.google.firebase.database.DataSnapshot;  
**import** com.google.firebase.database.DatabaseError;  
**import** com.google.firebase.database.DatabaseReference;  
**import** com.google.firebase.database.FirebaseDatabase;  
**import** com.google.firebase.database.ValueEventListener;  
  
**public class** RegisterActivity **extends** AppCompatActivity {  
  
 EditText **Phone**, **Name**, **Address**, **House**, **Block**, **Road**, **Password**;  
 Button **RegisterButton**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_register***);  
  
 **Phone** = (EditText) findViewById(R.id.***phone***);  
 **Name** = (EditText) findViewById(R.id.***name***);  
 **Address** = (EditText) findViewById(R.id.***address***);  
 **House** = (EditText) findViewById(R.id.***house***);  
 **Road** = (EditText) findViewById(R.id.***road***);  
 **Block** = (EditText) findViewById(R.id.***block***);  
 **Password** = (EditText) findViewById(R.id.***password***);  
 **RegisterButton** = (Button) findViewById(R.id.***register***);  
  
 **final** ProgressDialog dialog = **new** ProgressDialog(**this**);  
 **final** FirebaseDatabase database = FirebaseDatabase.*getInstance*();  
 **final** DatabaseReference myRef = database.getReference(**"user"**);  
  
 **RegisterButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 dialog.setMessage(**"Please wait..."**);  
 dialog.setIndeterminate(**true**);  
 dialog.show();  
  
 myRef.addValueEventListener(**new** ValueEventListener() {  
  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot) {  
  
 **if** (dataSnapshot.child(**Phone**.getText().toString()).exists()) {  
  
 dialog.dismiss();  
 **Phone**.getText().clear();  
 **Name**.getText().clear();  
 **Address**.getText().clear();  
 **House**.getText().clear();  
 **Road**.getText().clear();  
 **Block**.getText().clear();  
 **Password**.getText().clear();  
 }  
 **else** {  
 dialog.dismiss();  
 User user = **new** User(**Address**.getText().toString(), **Block**.getText().toString(), **House**.getText().toString(), **Name**.getText().toString(), **Password**.getText().toString(), **Road**.getText().toString());  
 myRef.child(**Phone**.getText().toString()).setValue(user);  
 Toast.*makeText*(RegisterActivity.**this**, **"Successful !"**, Toast.***LENGTH\_SHORT***).show();  
 Intent intent = **new** Intent(RegisterActivity.**this**, MainActivity.**class**);  
 startActivity(intent);  
 }  
 }  
  
 @Override  
 **public void** onCancelled(@NonNull DatabaseError databaseError) {  
  
 }  
 });  
 }  
 });  
  
 }  
}

**User :**

**package** com.example.pizzaorder;  
  
**public class** User {  
  
 **private** String **Address**;  
 **private** String **Block**;  
 **private** String **House**;  
 **private** String **Name**;  
 **private** String **Password**;  
 **private** String **Road**;  
  
 **public** User(String address, String block, String house, String name, String password, String road) {  
 **Address** = address;  
 **Block** = block;  
 **House** = house;  
 **Name** = name;  
 **Password** = password;  
 **Road** = road;  
 }  
  
 **public** User() {  
  
 }  
  
 **public** String getAddress() {  
 **return Address**;  
 }  
  
 **public** String getBlock() {  
 **return Block**;  
 }  
  
 **public** String getHouse() {  
 **return House**;  
 }  
  
 **public** String getName() {  
 **return Name**;  
 }  
  
 **public** String getPassword() {  
 **return Password**;  
 }  
  
 **public** String getRoad() {  
 **return Road**;  
 }  
  
 **public void** setAddress(String address) {  
 **Address** = address;  
 }  
  
 **public void** setBlock(String block) {  
 **Block** = block;  
 }  
  
 **public void** setHouse(String house) {  
 **House** = house;  
 }  
  
 **public void** setName(String name) {  
 **Name** = name;  
 }  
  
 **public void** setPassword(String password) {  
 **Password** = password;  
 }  
  
 **public void** setRoad(String road) {  
 **Road** = road;  
 }  
}

**REFERENCES**

* Landau, Peter. “Project Scope 101 - ProjectManager.com”, *ProjectManager.com*, 1 Feb. 2019, Redirecting from <https://www.projectmanager.com/blog/project-scope>.
* “Project Proposal Template Sample”, *MyCourses*, Redirecting from <https://mycourses.aalto.fi/mod/page/view.php?id=26348>.
* Waliaula, Brian. “ONLINE ORDERING SYSTEM PROJECT PROPOSAL.” *Academia.edu - Share Research*, Redirecting from <https://www.academia.edu/4935972/ONLINE_ORDERING_SYSTEM_PROJECT_PROPOSAL>.
* Soder, Chuck. "Online Ordering System Will Get Bigger Slice of Case Students' Pie." *Crane's Cleveland Business News*. 14 May 2007. From <https://www.crainscleveland.com/article/20070514/SUB/70511028/online-ordering-system-will-get-bigger-slice-of-case-students-pie>
* Shanker, Deena. "Online food delivery ordering is about to overtake phone ordering in the US”. *Quartz*. 10 Jan 2016. Retrieved from <https://qz.com/452609/online-food-delivery-ordering-is-about-to-overtake-phone-ordering-in-the-us/>