EARNEASY: A MOBILE CROWD SOURCING APPLICATION

Software Project lab-3

Course Code: **Software Project Lab III (SE 801)**

EarnEasy:
A Mobile Crowd Sourcing Application

Submitted By

Pritom Kumar Das

BSSE 0919

Supervised By

Dr. Md. Shariful Islam

Professor
Institute of Information Technology
University of Dhaka

Submission Date: **23 February, 2021**

Institute of Information Technology

University Of Dhaka

LETTER OF TRANSMITTAL

23 February, 2020

BSSE 4th Year Exam Committee

Institute of Information Technology

University of Dhaka

Subject: Submission of mid report of "EarnEasy: A Mobile Crowdsourcing Application".

Dear Sir,

With due respect, I am pleased to submit the final report of Software project lab-III on "EarnEasy: A Mobile Crowdsourcing Application". I have tried my best to deliver a good report. However, it might lack perfection. So, I therefore, hope that you would be kind enough to accept my report and oblige thereby.

Sincerely yours,

Pritom Kumar Das

BSSE 0919

8th Semester

Bachelor of Science in Software Engineering

Institute of Information Technology

University of Dhaka.

LETTER OF ENDORSEMENT

February 23, 2021

BSSE 4th Year Exam Committee

Institute of Information Technology

University of Dhaka

Subject: Approval of the Report

This letter is to certify that, Pritom Kumar Das, BSSE 0919, student of Institute of Information Technology, University of Dhaka, has done "EarnEasy: A Mobile Crowdsourcing Application" under my supervision. I have gone through the report. All the information mentioned in this document is true.

I wish him every success in life and hope that he will continue his effort in the future.

SPL Supervisor

Dr. Md. Shariful Islam

Professor

Institute of Information Technology

University of Dhaka

ACKNOWLEDGEMENT

Firstly, I would like to thank the Almighty for helping me complete the final report.

Abstract

This document contains the software requirements and specifications, architectural design and user interface design, implementation details, testing and user manual of "EarnEasy: A Mobile Crowdsourcing Application". Mobile crowdsourcing is an emerging technology that has enormous potential in Bangladesh. It is a term that describes crowdsourcing activities that use information collected through smartphones or other mobile devices. Companies such as Pathao, Uber, Food Panda, Sohoz are tapping into this potential to proliferate their business. However, this is only a tiny portion of what can be done using mobile crowdsourcing. With mobile crowdsourcing we can tackle the issue of lack of communication between companies and users. The goal of our project is to help the companies to get the opinion of the crowd regarding the usage, storage, purchase of their products with a simple, reliable and trustworthy method. Thanks to various useful smartphone features, including reliable GPS, excellent cameras, as well as the easy availability of smartphones; users can work on crowdsourcing tasks without almost zero difficulties. The companies can provide any task for example stock checking, user feedback of items or surveys, just-in-time checking on product situations or other simple tasks that can help with the company's growth, prestige, popularity or profit to the users of the application. As a simple task it does not need any device other than a smartphone. The users will complete the task to earn the financial benefits that the companies offer. It will create a large body of working people and help mitigate the problem of unemployment in our country. In the era of Digital Bangladesh this project can become the gateway that connects both the companies and people. This way the companies and the users both can gain benefits through a mutual win-win situation. This document can be followed to develop a mobile crowd sourcing application.

Table Of Contents

List Of Figures

List of Tables

CHAPTER 1 : INTRODUCTION

This project is aimed to use Mobile crowdsourcing for providing financial benefits to end users and at the same time participating companies or organisations can benefit by seeking knowledge, goods, or services from a large body of people. EarnEasy is a software that will create an efficient , simple and reliable platform to connect the companies and people.

Crowdsourcing is the generalized act of outsourcing tasks, traditionally performed by employees or contractors, to a large group of the Internet population (the wise crowd) by means of an open call. With the great development of smartphones with rich built-in sensors and ratio interfaces, mixing smartphone-based mobile technologies and crowd sourcing offers vast computing resources, and leads to a new paradigm called Mobile Crowdsourcing (MCS). It involves obtaining work, information, or opinions from a large group of people who submit their data via the Internet, social media, and smartphone apps. It allows companies to farm out work to people anywhere in the country or around the world, which lets businesses tap into a vast array of skills and expertise without incurring the normal overhead costs of in-house employees. As an alternative to traditional financing options, crowdsourcing taps into the shared interest of a group, bypassing the conventional gatekeepers and intermediaries required to raise capital. So, this project will aim to create a crowdsourcing platform to connect companies with general populations.

This chapter describes the purpose, scope, assumption, and definitions of the "EarnEasy: A Mobile Crowdsourcing Application" System

1.1 PURPOSE

The purposes of this document are:

- Identify the requirements that have to be carried out as the part of the project.
- Form the baseline for construction of the proposed system.
- Help to reduce the development effort and reveal misunderstandings, and inconsistencies early in the development cycle when these problems are easier to correct.

1.2 SCOPE

The scope of the project is given below:

- This project will work on only android system.
- It will be developed and tested by flutter and android studio.
- The system will only work with image and text.
- The members will use smart phones to run the client side application.

1.3 ASSUMPTIONS

The assumptions of the project are:

- The underlying network is completely reliable.
- User will have reliable internet and GPS connection.

1.4 DEFINITIONS

This report uses the following terminologies in its description.

1.4.1 Mobile Crowdsourcing

Crowdsourcing is a business model or function that relies on a large group of users as third parties for outsourcing certain tasks. Mobile crowdsourcing is using smartphones as the medium of crowdsourcing strategies . The popular use of the internet makes communication and coordination progressively cheap , tasks that would have been impossible to communicate and coordinate before have become extremely easy to set up and coordinate.[2][3] Crowdsourcing can add significant value to a product or service, and can also generate valuable connections between the users and the company.[4] Crowdsourcing is the practice of engaging a 'crowd' or group for a common goal often innovation, problem solving, or efficiency. Crowdsourcing can take place on many different levels and across various industries.[5]

CHAPTER 2 : ELICITATION

2.1 INTRODUCTION

Requirements Elicitation is a part of requirements engineering that is the practice of gathering requirements from the users, customers and other stakeholders. Many difficulties were faced, like understanding the problems, making questions for the stakeholders, limited communication with the stakeholders due to a short amount of time and volatility. Though it is not easy to gather requirements within a very short time, these problems have been surpassed in an organized and systematic manner.

2.2 ELICITING REQUIREMENTS

The main task of this phase is to combine the elements of problem solving, elaboration, negotiation and specification. The collaborative working approach of the stakeholders is required to elicit the requirements. The following tasks were done for eliciting requirements-

- Quality Function Deployment
- Usage Scenarios
- Elicitation of work products

2.3 QUALITY FUNCTION DEPLOYMENT

Quality Function Deployment (QFD) is a structured approach to defining customer needs or requirements and translating them into specific plans. It turns subjective quality criteria into objective ones which can be used to design and manufacture the products. This methodology concentrates on customer satisfaction from the software engineering process. The following requirements are identified by QFD-

2.3.1 Normal Requirement

The normal requirements are generally the objectives and goals that are stated for a product or system during meetings with the user. The presence of these requirements fulfills users' satisfaction. These are the normal requirements for the project.

- Error free easy accessible system
- User-Friendly Interface
- Android application
- Effective System
- Separate android application for companies.
- Accessible via the Internet.
- Allow valid users to login and logout.
- Restrict access to functionality of the system based upon user roles
- Allow administrators of the system to change provided information and configure parameters of the system
- Allow valid users that log in to use exciting features of application.
- Allow Administrators to delete , modify, add or update any information regarding tasks.
- This application can be used by any android device with connection to internet.
- A dedicated server have to run all the time to help users get information.
- Maintain proper security for all the data.
- Maintain a database of all users and information.
- Allows user option to modify profile or not.
- Allows user option to change language.
- Allows user to contact with admin through email if needed
- Fast loading application.
- Allow valid users assign/find tasks.
- Sort tasks based on money/distance.

2.3.2 Expected Requirement

These requirements are intrinsic to the product or system and may be so elementary that the customer does not explicitly state them. Their absence will be a cause for significant dissatisfaction. Below the expected requirements for our project are briefly described.

- Payment system integration
- Error-free software
- Map facilities
- Super Admin monitors all activities
- Provide Search Facilities
- No ambiguous feature

- Data backup
- Sending notification to users if necessary.

2.3.3 Exciting Requirement

These requirements are for features that go beyond the customer's expectations and prove to be very satisfying when present

- Work scheduling based on user merit.
- Showing tasks based on zoom level.
- The user interface should provide appropriate error messages for invalid input or show message if search results are not found.

2.4 USAGE SCENARIO

EarnEasy is a mobile crowd sourcing application. This project will explore the huge untapped potential of mobile crowdsourcing in the context of our country. Our objective is to use mobile crowdsourcing to create an application that connects both the users and the companies. We want to create a workable, sustainable and maintainable crowdsourcing platform to create opportunities for both individuals and companies alike to use crowd engine.
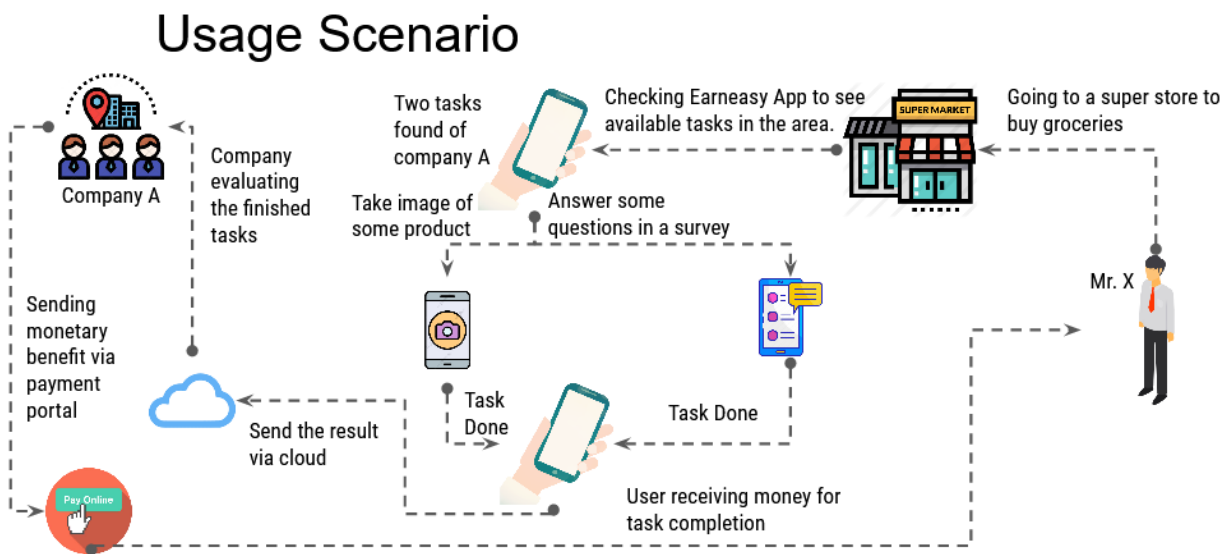

FIGURE 1 EARNEASY SERVICE USAGE SENARIO

5

Imagine a simple situation, Mr. X went to a grocery shop to buy some groceries for his family. He opened the EarnEasy app to check the tasks available in the area. He can accept tasks that require him to take some pictures of a particular product or talk to the shop employee about its price or other qualities. After doing some tasks he can upload the photos to the task provider via EarnEasy or answer a simple survey to best describe his experience. This way Mr. X earns some money in the midst of doing groceries. This is just one of the uses of mobile crowdsourcing. As the use of smartphones is becoming more and more popular, the practical use of mobile crowdsourcing becomes more apparent.

EarnEasy will connect the companies and the users. The companies can provide tasks or surveys with adequate payment, work description and location for the task. Through the app the users can search for the available tasks in any area or near their location. The users will need to get to the designated area to accept and start a task. The user location can be traced by using the GPS of the smart phone. Tasks will be very simple like taking some pictures of some products or completing a survey and answering some questionnaires. After completing the tasks, the user can send the necessary images via the mobile app. It will have certain time limit. The companies then can compare the documents with their requirements. If okay, the companies can send the money through the payment service of the application. Otherwise, they can give further instructions. The members can receive money through the payment portal and can send complaint in case of any delay or miscommunication. To maximize data connectivity, we will use Firestore servers. We will record all transactions to prevent any miscommunications. For better location service we will use Google Maps.API. Many services will be provided to the users of the application. Notification will be sent in case of any task, payment or account issues. There will be two major types of service based on the type of the user.

Company Features

The registered companies will be verified before they can use EarnEasy services. Companies that use our app will be entitled to many services. Such as -

Privilege of issuing tasks - Companies can issue any task that they want. They can issue the task multiple times in multiple places. They can also choose to restrict the tasks according to the level of members.

Count members in a given area - The companies can not see the location of an individual member but can see the number of members in a given area.

Task rights - Companies retain the rights to change, stop, delete or invalidate any task that they have issued.

Rate Member Performance - They can rate the member based on their task completion.

Member Features

Normal users or members will be entitled to many services. Such as -

Accept any task - Members can accept any task according to their choice.

Sort tasks - Members can sort tasks according to payment, area, difficulty, distance etc.

Anonymity - Members are anonymous to the companies. The companies can only see the level and rating of the members. Their personal information will be hidden.

Rate task - The users can also rate the task after completing it.

CHAPTER 3 : SCENARIO BASED MODELING

Scenario based modelling is an inexpensive rapid prototyping technique. This method is effective when systems are being built with the requirements vaguely known at the outset. Users are involved right from the start, to build prototypes evolving towards the final product. The users are also involved with the testing of the prototypes which is essential for the validation of requirements and help the users to gain an initial experience of the final system during the development itself. This method involves techniques which are applied by one or more professionals working alongside users who are expected to provide and specify their requirements at the beginning as well as evaluate and approve the system upon completion. The user (in a passive capacity) and the designer/builder (an active partner) cooperate to reach a working model where the means of communications are by the examination of preliminary models such as the initial narratives, paper models and graphical representations built to represent the final system functions.

This chapter describes Scenario Based Modeling of the EarnEasy: A Mobile Crowdsourcing Application system.

3.1 USE CASE DIAGRAM

A use case diagram is a graphic depiction of the interactions among the elements of a system. The purposes of use case diagrams are:

- Gathering requirements of a system.
- Getting an outside view of a system.
- Identifying external and internal factors influencing the system.
- Showing the interactions among actors.

The first step in writing a use case is to define the set of actors that will be involved in the story. Actors are of two types. They are:

- Primary Actors: Primary actors are the actors using the system to achieve a goal. They both consume data and produce information.
- Secondary Actors: Secondary actors are the actors that the system needs assistance from to achieve the primary actor's goal. They either consume data or produce information.

Once actors have been identified, use cases can be developed.

TABLE 1 USE CASE DIAGRAM

Level 0	Level 1	Level 2	:Level 3
EarnEasy System	Account Management	Sign up	
		Log out	
		Log in	
		Update	
		Verification	
		Delete	
	User Management	Company	Authenticate
			Fill up description
			Add tasks
			Create/Provide/Modify work
			Rate member's work
			Verify result
		Member	GPS check
			Start task
			Sort tasks
			Rate Task
			Task complete verification
			Send results
	Notification Management	Task	
		Payment	
		Account	
		Error	
	Payment Management	Payment portal	Send money
			Receive money
		Complaint	
	Search	Search tasks	
		Search member count	

Level 0: EarnEasy Mobile Crowdsourcing System

Primary actors: Company, Member.

Secondary actors: Admin, Database.

Goal in context: The diagram shown in figure: 2 represents the whole EarnEasy Mobile Crowdsourcing System.
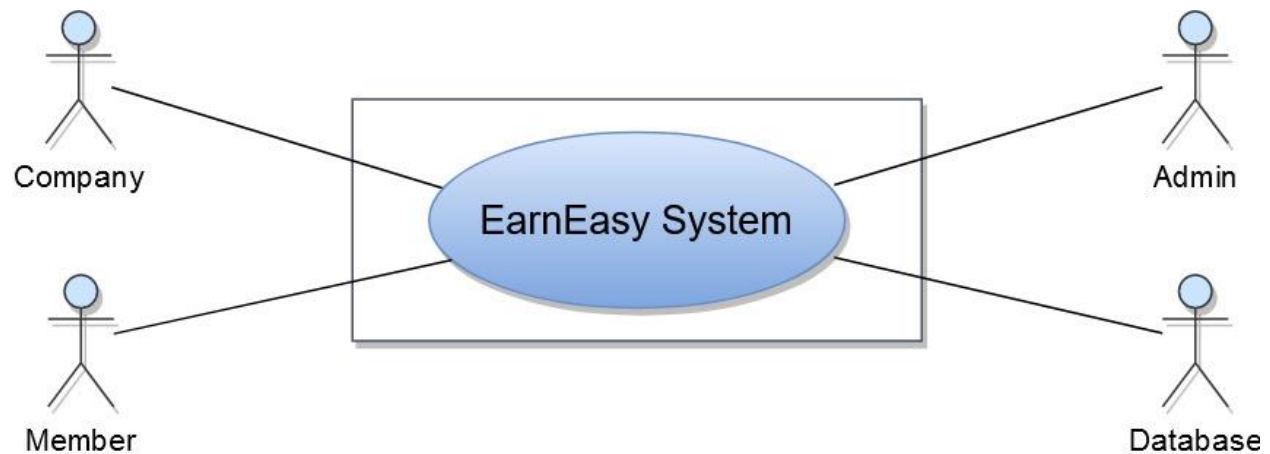
FIGURE 2 LEVEL 0 OF EARNEASY SYSTEM

Level 1: Modules of EarnEasy Mobile Crowdsourcing System

Primary actors: Company, Admin, Member.

Secondary actor: Admin, Company

Goal in context: The diagram shown in figure: 3 shows all the modules of the EarnEasy Mobile Crowdsourcing System.

EarnEasy Mobile Crowdsourcing System consists of 5 modules. They are:

Level 1.1: Account Management

Level 1.2: User Management

Level 1.3: Notification Management

Level 1.4: Payment Management

Level 1.5: Search

Level 1.1: Account Management

Primary actor: Company, Member.

Secondary actors: Admin, Database.

Goal in context: The diagram shown in figure: 4 refers to the details of the Account Management module of level 1.

FIGURE 4 LEVEL 1.1 OF EARNEASY SYSTEM

Actions and Replies

A1: User want to sign up/log in/ log out of system.

R1: The admin and database stores the values and provides verification.

A2: User wants to update/delete/change their profile.

R2: System stores those values.

Level 1.2: User Management

Primary actor: Company, Member.

Secondary actors: Admin, Database.

Goal in context: The diagram shown in figure: 5 refers to the details of the User Management module of level 1.
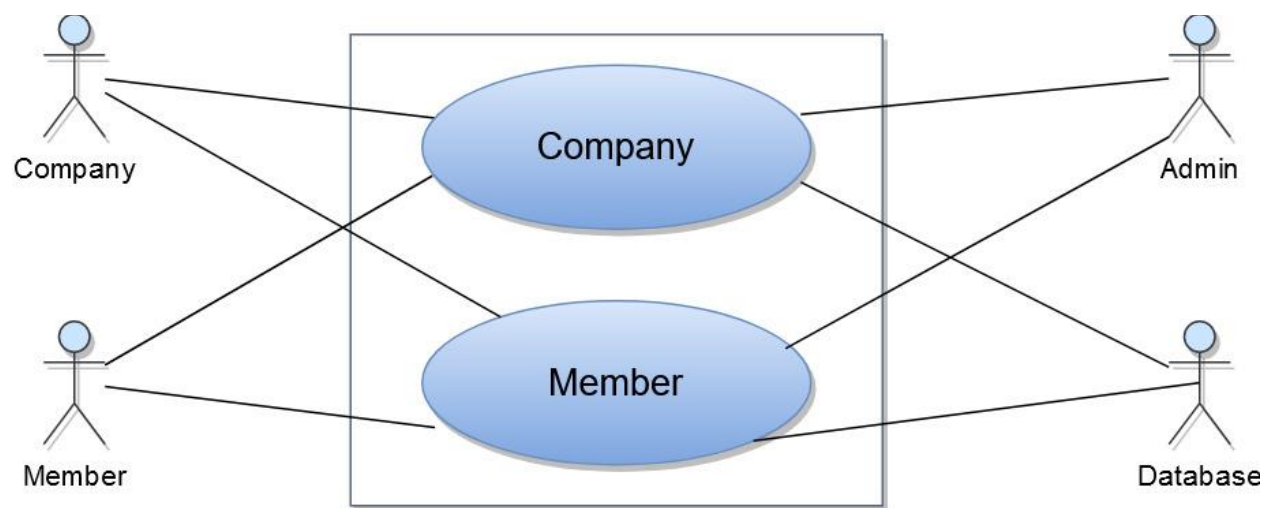
FIGURE 5 LEVEL 1.2 OF EARNEASY SYSTEM

Actions and Replies

A1: Company wants to create task that they need.

R1: The admin authenticates the company and the system stores the necessary data.

A2: Company wants to see the results send by the members.

R2: System sends the result of the particular task created by the company.

A3: The members wants to receive/accept tasks

R3: The system checks members location via GPS and provides the necessary information.

A4: The member completes the task.

R4: System verify the work and send it to the company.

13

A5: Member want to sort the tasks by distance/payment/title.

R5: System sort the available tasks.

Level 1.3: Notification Management

Primary actor: Company, Admin, Member.

Secondary actors: Company, Admin, Member, Database.

Goal in context: The diagram shown in figure: 6 refers to the details of the Notification Management module of level 1.
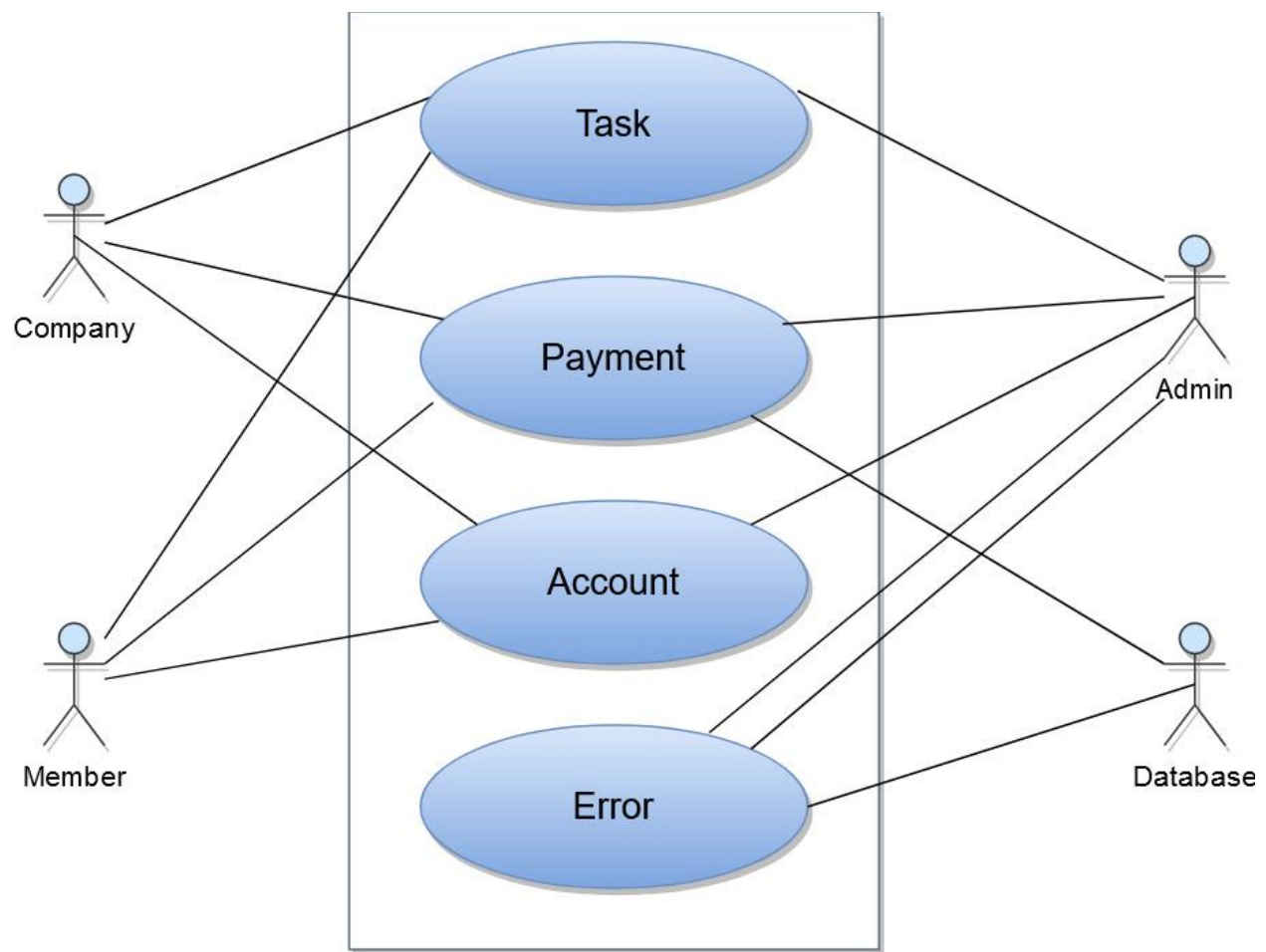
FIGURE 6 LEVEL 1.3 OF EARNEASY SYSTEM

Actions and Replies

A1: User wants to know if the given task is finished/evaluated/complete.

R1: System sends the necessary notification message.

A2: User wants to know the payment conditions

R2: System sends the necessary notification message.

A3: User wants to know account situation.

R3: System sends the necessary notification message.

A4: User wants to know the reason for errors. (If occurs).

R4: System sends the necessary notification message.

Level 1.4: Payment Management

Primary actor: Company, Member.

Secondary actors: Admin.

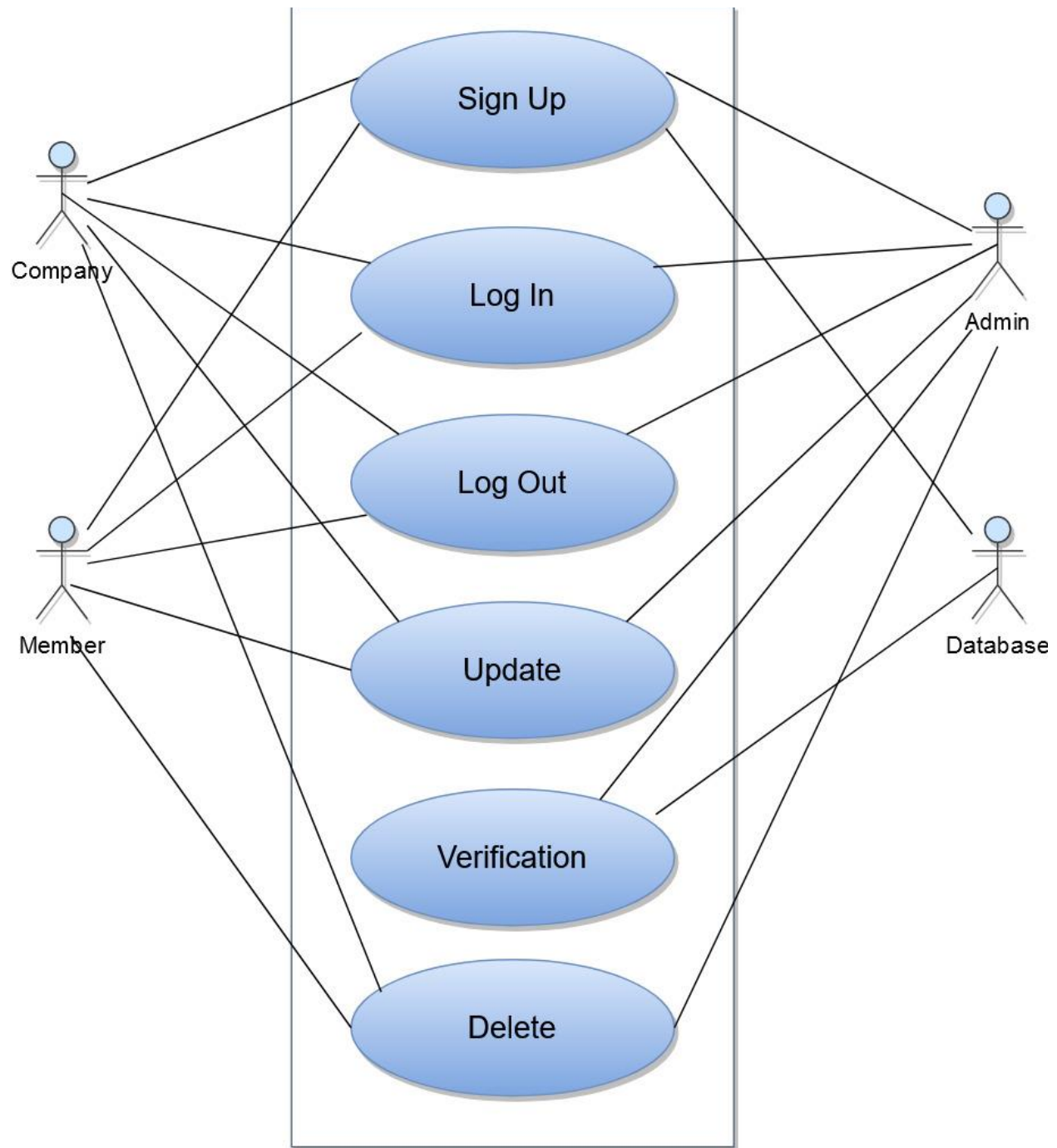**Goal in context:** The diagram shown in figure: 7 refers to the details of the Payment Management module of level 1.
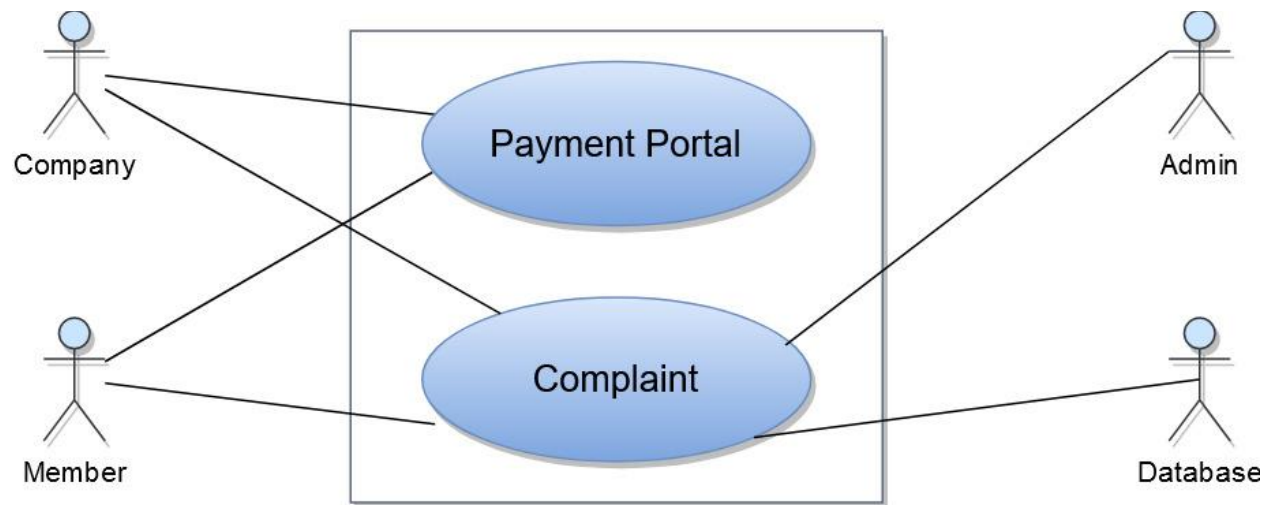
FIGURE 7 LEVEL 1.4 OF EARNEASY SYSTEM

Actions and Replies

A1: Member wants to receive payment for completed task

R1: Company sends the payment via payment portal.

A2: Company wants to send payment for completed task

R2 Member receives the payment via payment portal.

Level 1.5: Search

Primary actor: Company, Member.

Secondary actors: Database.

**Goal in context:** The diagram shown in figure: 8 refers to the details of the Search module of level 1.

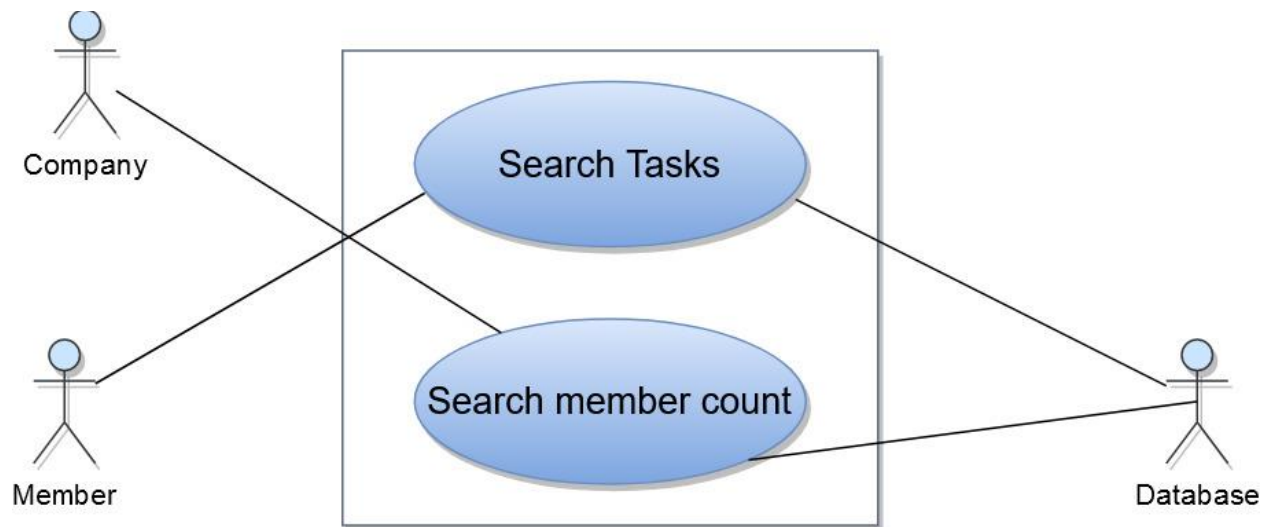*FIGURE 8 LEVEL 1.5 OF EARNEASY SYSTEM*

Actions and Replies

A1: Member wants to search for tasks in a given area.

R1: System performs the search according to mobile zoom level.

A2: Company wants to know the number of members in a given area.

R2: System sends the result according to mobile zoom level.

CHAPTER 4 : DATA-BASED MODELING

This chapter presents ER diagram and schema tables of the EarnEasy: A Mobile Crowdsourcing Application System.

4.1 DATABASE

We will use firestore nosql cloud database as our data storing option. NoSQL or the so-called Not-Only SQL database stores the unstructured data in JSON format and provides a unique data storage and access mechanism that is quite different from the tabular relations in RDBMS. These unstructured databases are widely known for their non-relational and schema less data model, improved performance and scalability factors which are always an issue with relational database systems. The NoSQL databases are basically developed to meet the requirements of the modern cloud-based decentralized apps and are a good solution as compared to the relational databases in many ways.

We have decided to use this for the following reasons-

4.1.1 Continuous Availability

The various relational databases may show up modern to high availability for the data transactions while this is much better with the NoSQL databases which excellently show up continuous availability to cope up with different sorts of data transactions at any point of time and in difficult situations.

4.1.2 Low Latency Rate

It is easy and less time consuming to derive data from the unstructured data models in the Not-Only SQL databases. Hence, response times with these databases are fast enough and can handle the most intense operations for the applications.

4.1.3 Easy Scalability

With NoSQL, it's easy to scale database for current and future requirements. These databases can easily handle data partitioning across multiple servers to meet the increasing data storage requirements. It's quite affordable to scale a NoSQL database in which inexpensive hardware or servers can be utilized for the purpose as compared to the relational databases which require expensive hardware solutions to meet up the scalability issues arising out of growing needs.

4.1.4 Ability to handle changes

The schema-less structure of the NoSQL databases helps it cope up easily with the changes coming with time. There is a universal index provided for structure, values and text found in the data and hence, it's easy for the organizations to cope with the changes immediately using this information.

4.2 ER DIAGRAM

Although we are using noSql database we still need to know the relationship between the data entities. Figure 9 shows ER diagram of the EarnEasy: A Mobile Crowdsourcing Application System.
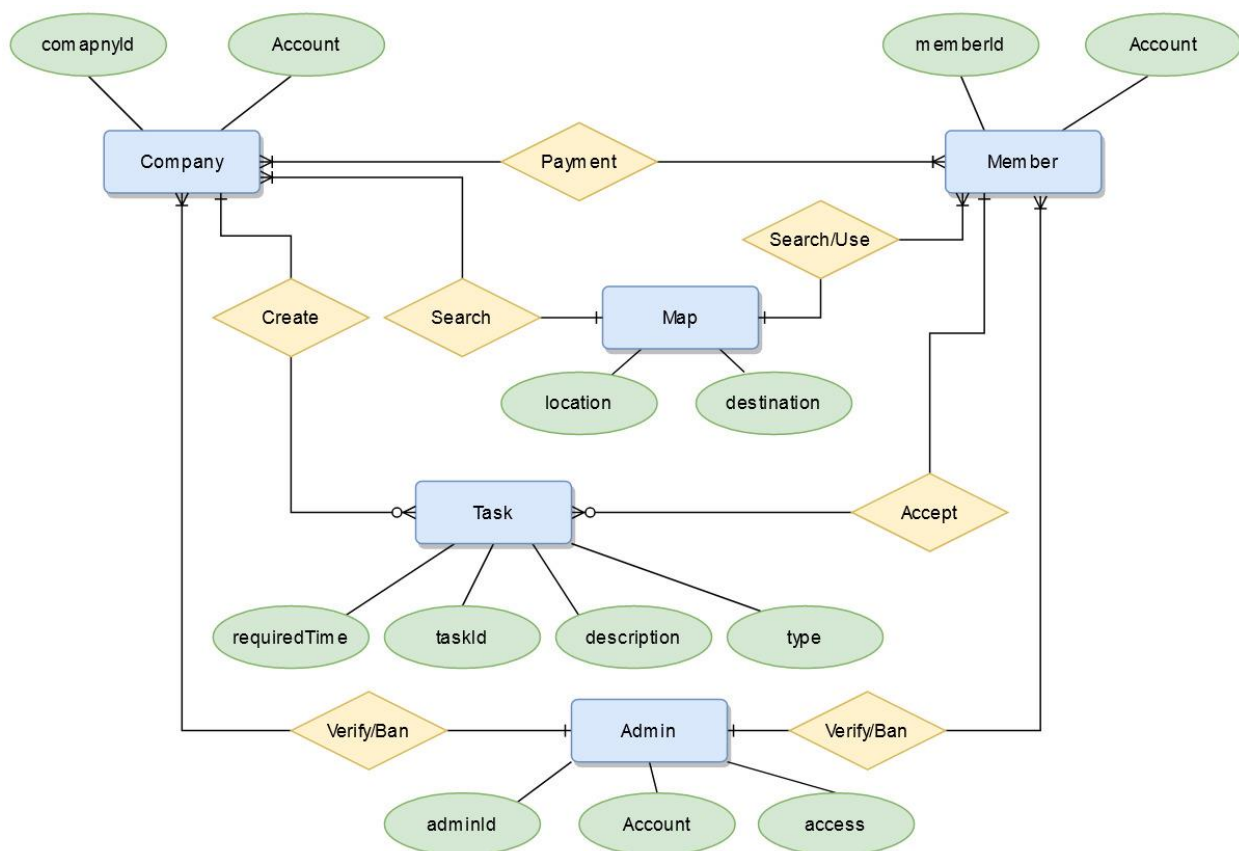
FIGURE 9 ER DIAGRAM OF EARNEASY SYSTEM

CHAPTER 5 : CLASS BASED MODEL

This chapter describes the class based modeling of the EarnEasy: A Mobile Crowdsourcing Application System.

5.1 ANALYSIS CLASSES

After identifying nouns from scenario, I have filtered nouns belonging to solution domain using General Classification (External entities, Things, Events, Roles, Organizational units, Places, and Structures). Nouns selected as potential class were filtered using Selection Criteria (Retained information, Needed services, Multiple attributes, Common attributes, Common operations, and Essential requirements). After performing analysis on potential classes, I have found the following analysis classes:

TABLE 2 ANALYSIS CLASSES

Serial No	Class name	Attributes	Methods
1	User (Company)	companyId, Account	addTask(), createAccount()
2	User (Member)	memberId, Account	startTask(), createAccount()
3	Admin	Authenticate,	verifyAccount()
4	Map	Location	showTasks(), searchMembers()
5	Account	AccountType,	showAccount()
6	Message	Message body, type	showMessage()
7	Task	taskType, description, isRequired	showTask(), changeTaskType()
8	Database	Store, update, retrieve	storeData(), updateData(), retriveInformation()
9	System	GeneratedQuery, Send, Retrieve	generateSearchQuery(), performRoutineCheck()

5.2 CRC CARD

The class cards of the analysis classes are given below:

TABLE 3 USER (COMPANY) CRC CARD

User (Company)	
Attribute	**Methods**
companyName email level companyID companyAddress certification writeAccess city zipCode photoUrl phoneNumber taskCompletedList	• viewMap() • viewInformation() • viewAccount() • updateAccount() • search() • addTask() • modifyTask() • sendPayment() • signIn() • signOut() • deleteAccount()
Responsibilities	**Collaborator**
Add Task	Task, Admin
Search member count	Database, System
Send payment	Member
Receive task Result	Database. Member
Create Account	Account, Admin
Rate member	Database, System

TABLE 4 USER(MEMBER) CRC CARD

User (Member)	
Attribute	**Methods**
userID userName email level address type writeAccess city occupation zipCode photoUrl phoneNumber gender employmentStatus taskCompletedList	• viewMap() • viewInformation() • viewAccount() • updateAccount() • search() • startTask() • completeTask() • receivePayment() • sendComplaint() • signIn() • signOut() • deleteAccount()
Responsibilities	**Collaborator**
Start/Complete Task	Task, Admin, Company
Search tasks in an area	Database, System
Receive payment	Company
send task result	Database, System
Create Account	Account, Admin
Rate task	Database, System

TABLE 5 USER(ADMIN) CRC CARD

User (Admin)	
Attribute	**Methods**
adminID adminName password email level address type writeAccess photoUrl phoneNumber	• viewMap() • viewInformation() • viewAccount() • updateAccount() • searchUser() • banCompany() • banMember() • verifyUser() • signIn() • signOut() • deleteAccount()
Responsibilities	**Collaborator**
Verify company	Database, System
Search user	Database, System
Ban User	System, Database
Create Account	Account, Admin

TABLE 6 MAP CRC CARD

Map	
Attribute	**Methods**
Location ID Destination Latitude Longitude Area	• showMap() • showTaskLocation() • viewLocation(latlong) • searchTasks() • viewUserlocation() • changeZoomLevel()
Responsibilities	**Collaborator**
Show map with task location	Database, System
Search task location	Database, System
Change zoom level	System
View user location	System

TABLE 7 ACCOUNT CRC CARD

Account	
Attribute	**Methods**
ID Name Address Type writePermission	• createAccount() • update Account() • delete Account() • checkPermission()
Responsibilities	**Collaborator**
Create/Update/delete Account	Database
Check User permission	Database

23

TABLE 8 MESSAGE CRC CARD

Message	
Attribute	**Methods**
MessageID Description ErrorMessage	• showMessage() • sendMessage() • notify()
Responsibilities	**Collaborator**
Notify user	Database, System
Show message in case of any event	System
Send message as complaint	Member, Admin

TABLE 9 TASK CRC CARD

Task	
Attribute	**Methods**
taskId type taskDescription optionList isCompleted require startTime endTime requiredTime access companyId importantNote	• addTask() • addTaskType() • createTask() • saveCompanyId() • searchTaskById() • checkAccessLevel() • checkRequiredTime() • checkCompletionStatus() • isRequired()
Responsibilities	**Collaborator**
Check access level	System
Provide task time	Database, Company
Search task by Id	Database, System

TABLE 10 DATABASE CRC CARD

Database	
Attribute	**Methods**
store update retrieve	• store() • update() • getInformation() • SetInforamtion() • retrieve() • viewInformation()
Responsibilities	**Collaborator**
Store Information	Admin
update Information	Admin
Retrieve information	User
Perform routine check	System

TABLE 11 SYSTEM CRC CARD

System	
Attribute	**Methods**
GeneratedQuery Send Retrieve	• generateSearchQuery() • send() • retrieve() • showInfo() • formatInfo() • checkUserLocationByGPS() • routineDatabaseCheck()
Responsibilities	**Collaborator**
Generate search key to database query	database
Send information to database	admin
Format information to show to user	user
Check User Location By GPS	Member

25

5.3 UML

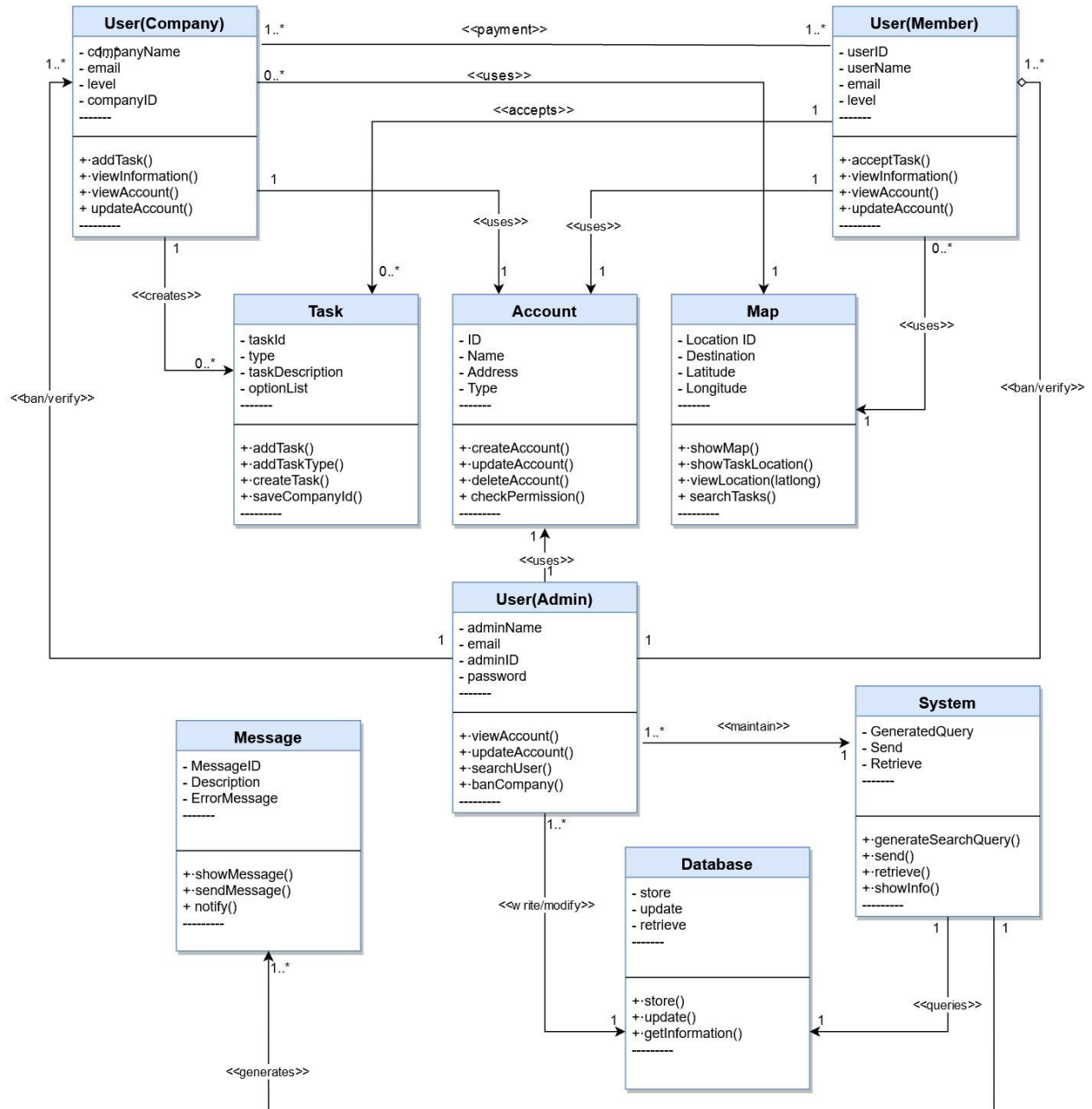This is the UML class diagram of EarnEasy System.

FIGURE 10 UML OF EASREASY SYSTEM

CHAPTER 6 : ARCHITECTURAL DESIGN

As architectural design begins, the software to be developed must be put into context—that is, the design should define the external entities (other systems, devices, people) that the software interacts with and the nature of the interaction. This information can generally be acquired from the requirements model and all other information gathered during requirements engineering. Once context is modeled and all external software interfaces have been described, you can identify a set of architectural archetypes.

This chapter describes architectural context diagram of the EarnEasy: A Mobile Crowdsourcing Application System.

6.1 ARCHITECTURAL CONTEXT DIAGRAM

At the architectural design level, a software architect uses an architectural context diagram (ACD) to model the manner in which software interacts with entities external to its boundaries. systems that interoperate with the target system (the system for which an architectural design is to be developed) are represented as

● Superordinate systems

● Subordinate systems

● Peer-level systems

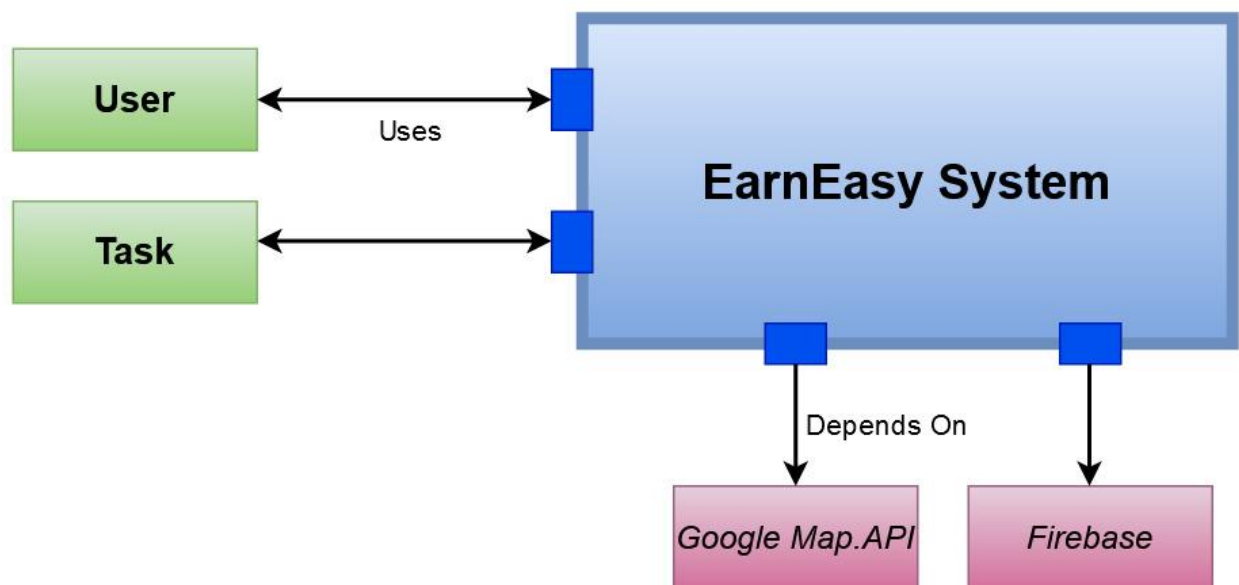The following diagram represents the software in context

FIGURE 11 ARCHITECTURAL CONTEXT DIAGRAM OF EARNEASY SYSTEM

6.2 REFINE THE ARCHITECTURE INTO COMPONENTS

As the software architecture is refined into components, the structure of the system begins to emerge. The analysis classes introduced in software requirement modeling represent entities within the application domain that must be addressed within the software architecture. Hence, the application domain is one source for the derivation and refinement of components. Another source is the infrastructure domain. The architecture must accommodate many infrastructure components that enable application components but have no business connection to the application domain.

The interfaces depicted in the architecture context diagram imply one or more specialized components that process the data that flows across the interface

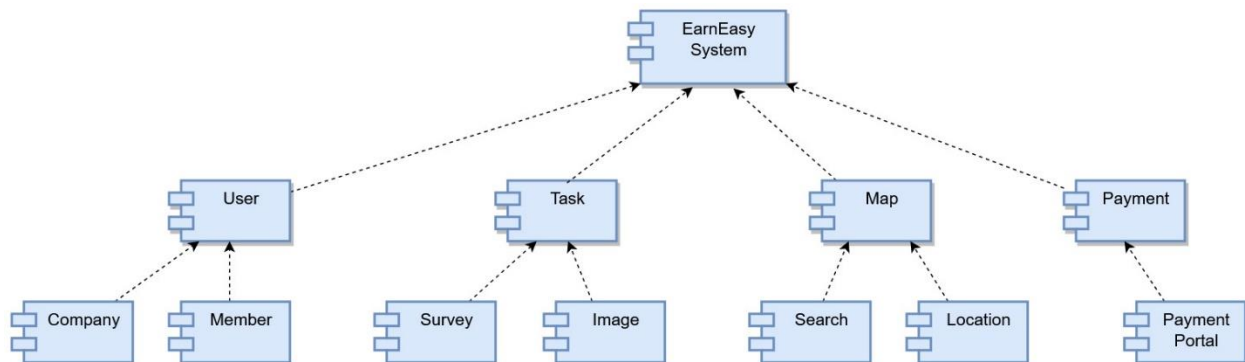For the proposed android application the following components can be introduced:

FIGURE 12 COMPONETS OF THE EARNEASY SYSTEM

CHAPTER 7 : MAPPING REQUIREMENTS TO SOFTWARE ARCHITECTURE

The requirements gathered in the project planning is mapped with the software architecture.
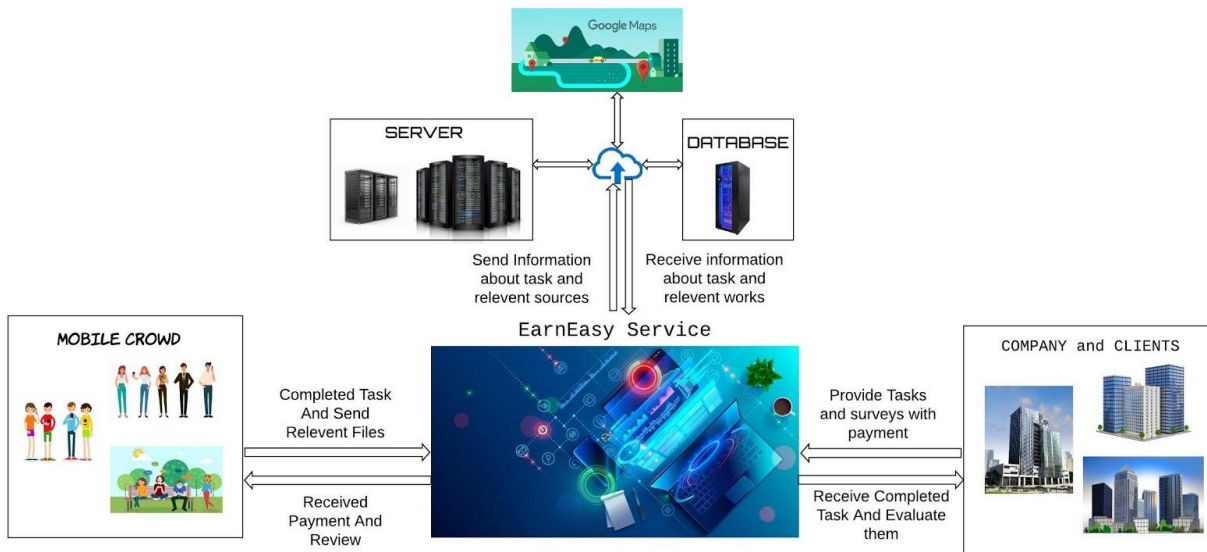
FIGURE 13 ARCHITECTURE OF EARNEASY SYSTEM

The following features are mapped into the architecture of the project form requirement analysis.

All time internet connectivity: By using googles firstore database we can provide all time connection with the server and as well as data analysis.

Platform between company and members: Using flutter to develop the mobile application has enabled us to use smartphone as a crowdsourcing media. And it in turn helps to create the platform.

Payment: Payment between company and members are carried through payment portals such as sslcommerz, bkash, sohoz etc

Task creation and acceptance: Creation and acceptance of task is achieved by using different roles of users. Such as company and member.

Super admin support: Super admin can check all activities and can ban any user that creates problem with firebase architecture. This in turn makes problem solving easier.

Anonymity of company and members: Companies are anonymous from the members as their information is not shown to the members. Same way the members achieve anonymity because the

companies can not directly view their profile. They can view selected information such as rating and levels.

Location and distance: Using google maps.api has enabled us to have a powerful map features in the application. It has also made it easy to implement other features such as searching in a area, getting user current location and other map related features.

Heuristic function: With this architectures we are able to get enough data from users to make a suitable heuristic function for assigning task to member and as well as just in time information sharing.

CHAPTER 8 : Methodology

In this project the various project requirements and other features are implemented through many processes. For simplicity we are dividing the whole project into the following two parts.

- Android application development
- Heuristic function development

8.1 ANDROID APPLICATION DEVELOPMENT

For developing the android application we divide the application into two part. The company part and member part.

8.1.1 Android Application Company

It is developed with flutter language. In this part to log in the company must gain a certificate from our EarnEasy system. We are using firestore as our background application. Companies can issue task here and can check the members count in a given area. It can also issue message to admin in case of any problems. The identity of companies will remain anonymous to the members and members can not access this feature.

8.1.2 Android Application Member

It is developed with flutter language. In this part the members can become members by using google or email login to see the full list of features available for them. There is no need for certificate from our EarnEasy system. We are using firestore as our background application. Members can apply and accept task here and can see all task in a given area according to their levels. They can also issue complaint to admin in case of any problems. The identity of members will remain anonymous to the companies because companies can only see the information provided by EarnEasy system such as user level and ratings.

8.1.3 Problems Faced During The Android Application Development Process

- The problem of using crowdsourcing is solved by using smart phones. Because it is highly available to most people.
- Always have internet connectivity is solved by using firebase firestore cloud database. It is horizontally scalable and uses googles optimization.
- Location finding is solved using mobile GPS.
- UI and UX design is implemented with development in flutter. As flutter is highly customizable and useful in rapid development. It is also used to provide maximum coverage with both IOS and Android implementation.
- Creating and storing tasks implemented with firestore .

- Searching task in a specific area is solved using geohashing and geolocation. Geohashing is taking the whole earth as a grid to store information of location. We use flutter GeoFlutterFire library to provide the solution. GeoFlutterFire uses the Firebase Firestore Database for data storage, allowing query results to be updated in realtime as they change. GeoFlutterFire selectively loads only the data near certain locations, keeping applications light and responsive, even with extremely large datasets
- To send the images we use asynchronous function to check the task in memory. And send all the results once the task is completed by user.
- Save memory from big size image we use a heuristic to convert and reduce the size of the images before sending to the data storage. We use flutter image compressor library to tackle the issue.
- For image task the members can take the photos of the products on selves and answer the questions on survey.
- The members can see all the tasks in a selected area. To maximize efficiency the task will be selected according to the zoom level.
- All images taken can be send later if the user has problem with internet connection. But have to finish in time.

8.2 HEURISTIC FUNCTION DEVELOPMENT

For an application like EarnEasy to prosper and flourish it always has think about the user experience and user satisfaction. For that to happen the core component of task assign between multiple members has to be developed with utmost care and prudence. For this we are using a heuristic model to rate the members and accept task according to their level.

To get the task the members apply for the task and wait to be assigned. To assign the task we take in user relative distance, user task completion history, current task usage, availability, local time, work involvement user level, rating by companies, any case of previous misconduct, number of works done, educational level, work time history and marital status. With these we create a heuristic function to create the likelihood of the members success in the task. Higher performance means high chance to get more valued tasks.

This part is still under process and likely to add more parameters to further improvement.

CHAPTER 9 : Testing

In this project, we test the system for admin, company and member interaction. In this section, we use black box testing on the system to find any bugs or problems.

9.1 Plan Identifier

On this project we have two android application for company and members. We use firestore noSql database. So we see various data input and output.

- Android Application (Company) Test Plan
- Android Application (Member) Test Plan

9.2 Introduction

This test plan has been developed for "EarnEasy: A Mobile Crowdsourcing Application". The whole system is created to test on android application with android studio and firebase firecast module for fast connectivity.

9.3 Features to be Tested

The following features are tested:

TABLE 12 FEAUTRES TO BE TESTED

Application	Features
Android Application (Company)	✓ Authentication ✓ Payment portal ✓ Search member count ✓ Add/Create task with various inputs ✓ Add/Create different types of task ✓ Account
Android Application (Member)	✓ Authentication ✓ Payment portal ✓ Search task ✓ Accept task with various situation ✓ Account create/remove/modify

9.4 Features not to be tested

All the features of the system are tested.

9.5 Approach

I used end to end automated (integration) testing technique to test the system as a black box.

9.6 Item Pass/Fail Criteria

If actual output of a test case does not match with expected output of the test case, the test case is considered as failed. 100% of all test cases should pass. No failed case should be crucial to the end-user's ability to use the application.

9.7 Test Deliverables

I will deliver test plan document, test case and test report.

9.8 Scheduling

Scheduling is given below with different part of Quality Assurance (QA) and duration –

TABLE 13 SCHEDULE OF TEST PLAN

QA	Duration
Test plan	2 days
Testing	3 days

9.9 Planning Risks and Contingencies

The following scenarios are considered as risks for the project:

- Delay in requirements engineering.

- Delay in developing.

- Modification in development technology.

TABLE 14 TEST CASES ANDROID APPLICATION (COMPANY)

Android Application (Company)						
Test Case ID	Scenario	Steps	Input	Expected output	Actual Output	Result
1	Authenticate using fake account	Trying to log in as company with fake credential	No input	An error message will show as company is not registered	Same as expected output	Passed
2	Not providing payment after receiving task result	Log in as company, then not paying after receiving task result	No Input	Payment not provided. Then an message showing the payment that is missed.	Same as expected output	Passed
3	Create / Add task with impossible parameter	Log in as company, then create/ add task with impossible parameter such as wrong location or negative payment or impossible deadline	Task creation parameters.	Provide meaningful parameter and an error message providing the location of the problem.	Same as expected output	Passed
4	Delete account after receiving work	Log in as company, then deleting account after receiving task result	No input	Send email with the warning printed and inform admin to take legal approach against the company	Same as expected output	Passed

TABLE 15 TEST CASES ANDROID APPLICATION (MEMBER)

			Android Application (Member)			
Test Case ID	Scenario	Steps	Input	Expected output	Actual Output	Result
5	Authenticate using fake account	Trying to log in as member with fake credential	No input	An error message will show to log in with valid email	Same as expected output	Passed
6	Accept task but not completing it.	Log in as member, then not completing accepted task	No input	Warning message will be send and decrease it members level	Same as expected output	Passed
7	Take on too many tasks in mutually distant areas in the same time frame	Log in as member, then accept too many tasks in mutually distant areas such as some task in Uttara and some task in Azimpur at the same time	No input	Provide warning to the user when accepting task and decrease user level if task is not completed.	Same as expected output	Passed
8	Tapping task completed option without completing task properly.	Log in as member, then tapping task is completed without finishing the required tasks.	Insuffic ient inputs for task	Provide warning message that task is not completed and decrease user level if task is not completed	Same as expected output	Passed

CHAPTER 10 : CONCLUSION

It was so much challenging to prepare a final report for the first time. I think that this report has been written in an easy-to-read way as well as with full information required to have a good concept over the idea. The reader of should easily understand the information of the report.

REFERENCES

1. A Review of Mobile Crowdsourcing Architectures and Challenges: Towards Crowd-Empowered Internet-of-Things . *Published in: IEEE Access , 07 December 2018*

2. A Survey on Security, Privacy, and Trust in Mobile Crowdsourcing . *Published in: IEEE Internet of Things Journal (Volume: 5 , Issue: 4 , Aug. 2018)*

3. CrowdEIM: Crowdsourcing Emergency Information Management Tasks to the Mobile Social Media Users . *Published in: 2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL) . Date of Conference: 2-6 June 2019*

4. Thebault-Spieker, Terveen, & Hecht. Avoiding the South Side and the Suburbs: The Geography of Mobile Crowdsourcing Markets. *Published - Feb 28 , 2015*

5. CrowdDBS: A Crowdsourced Brightness Scaling Optimization for Display Energy Reduction in Mobile Video *Published in: IEEE Transactions on Mobile Computing (Volume: 17 , Issue: 11 , Nov. 1 2018)*