GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

A HYBRID APP USING FLUTTER FOR FARMER CROP AND E-COMMERCE MANAGEMENT USING GOOGLE CLOUD AND PROVIDING SECURITY TO CLOUD USING DEFFIE HELLMAN IN AWS

A Project Report of Capstone Project -2

Submitted by

Syed Mohd Gulam Baquer

(1613101774 / 16SCSE101811)

in partial fulfilment for the award of the degree of

Bachelor of Technology

IN

Computer Science and Engineering

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Mr. G Nagarajan, M.Tech.,

Assistant Professor

APRIL / MAY- 2020

GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report "**A HYBRID APP USING FLUTTER FOR FARMER CROP AND E-COMMERCE MANAGEMENT USING GOOGLE CLOUD AND PROVIDING SECURITY TO CLOUD USING DEFFIE HELLMAN IN AWS**" is the bonafide work of "**Syed Mohd Gulam Baquer (1613101774)**" who carried out the project work under my supervision

SIGNATURE OF HEAD

DR. MUNISH SABARWAL

PhD(Management), PhD(CS)

Professor & Dean,

School of Computing Science & Engineering

SIGNATURE OF SUPERVISOR

Mr. G. NAGARAJAN

M.tech

Assistant Professor,

School of Computing Science & Engineering

ABSTRACT

India, is known as the land of Agriculture for many years. Most of the population of our country is totally dependent on agriculture for their survival. In a period of the last two decades technology is expanding in rural areas with an exponential rate, although it is expanding many farmers are not aware of the advancements in agriculture. Most of the farmers do not have any idea about the rates of their crops and the market value of their crops and they sell their product in a very low rate than that of the actual market value of the crops and sometimes the intervention of the third party also leads to a low market value of the crops. In today's world, farmers get news through the newspaper. Many farmers do not get the news about the nearest market that is present in their region because of that they don't get any information about the farming schemes. In the end, they have to sell their products at a very nominal price. But with the advancement of technology and mobile computing and recent technological advances in Cloud Computing technology give us the path for developing and nurturing advanced services for remote area and monitoring in many industry areas and the agriculture sector. So our idea is to make a hybrid Application, which is a hybrid app that helps the farmers which particularly focusses on selling and buying the crops without the intervention of any third party. Not only the farmer can sell his crops but also he can be able to track the daily rates and news about the crops from all over India.

Cloud Computing provided platform for better utilisation of the resource spread across the world. Being a nascent field, it is crowded with many different problems that the engineers and scientist are working assiduously to eliminate. One of the main drawbacks with cloud is security. So, this project proposes a mechanism for secure file storage cloud using encryption and Diffie Hellman. The algorithm involves encrypting the file stored on the cloud and using Diffie Hellman for authenticating the user to decrypt the required file on the web app using flask framework in python.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

SYMBOL NAME	ABBREVIATION
DH	Diffie Hellman
AWS	Amazon web services
GC	Google Cloud Console
W	Widgets
IaaS	Infrastructure as a service
PaaS	Platform as a service
SaaS	Software as a service
HTML	Hypertext Markup Language
BaaS	Backend as a service
AES	Advanced Encryption Algo

CHAPTER 1
INTRODUCTION

India is the country of farmers and agriculture. About 70% of the population of India is depends on agriculture. One-third of our National income comes from the agriculture sector. The development in agriculture has a grate impact and much to do with the economy of our country. In the upcoming years, the agriculture sector will undergo very dramatic changes. The vast majority of Indian farmers, which includes small-scale producers are often unable to access the information and the correct pricing of their crops and this leads to low income and intervention of the third party in selling their own crops, not only those they are unaware of the news and other commodities of the agriculture which leads to unhealthy crops production and sometimes all their crops gets destroyed. These are the problems that lead to exponential suicide rates of Indian farmers from the last ten years.

The data regarding farming are available from many sources like printed media, audio and visual aids, newspapers, television internet, mobile, etc. but the formats and structures of data are not similar. In today's world, farmers get news through newspapers and television. But not every farmer has time to read a newspaper or they don't watch television as they don't have much time to sit and watch a television for some time. So because of that, they don't get any idea about the current values about the farming schemes, in the end, they have to sell their products at very low cost.

And because they get very little money, they end up taking a loan from the bank or any other person on interest.

With the advancement of mobile computing and smartphones and due to the lower price of smartphones, every home is having at least one single smartphone.

So our idea is to make an application for the farmers which helps him to sell his crops and can directly interact with the client without the intervention of any third party, not only this the farmer has been able to track the daily price of the crops and vegetables and news about the crops, fertilisers and the weather.

For doing all these things we are taking the help of the Google Firebase and Google Cloud, using Google Firebase API, all the information, data, and authentication of the farmer gets stored on Google Cloud and It's providing security by default so we don't need to worry about the security concerns. For fetching the daily pricing and daily updates about the crops we are taking the help of the Government of India (GOI) Farmer API.

Cloud security is one of the main concerns in the cloud computing domain. Storing personal and sensitive information on a third-party storage medium poses serious risks of data theft and data misuse by any person with malicious intent. The threat is so humongous that it has dissuaded governments and many other big organisations from migrating their operations on a cloud platform. The traditional methods of securing files and information are superfluous in the scenario of cloud. Extensive research and study is undergoing in this field to make cloud more secure and reliable. Among this behemoth instances of research, some of the methods that stand out include AES encryption and Diffie Hellman Key Exchange. The latter method is so powerful that it may take millions of years for even the most powerful computers of

current times to crack the code and reads the file. Our approach proposes a method that involves encrypting the file using any standard encryption technique and using Diffie Hellman for user authentication. In this way the files can be saved in a public domain securely without the threat of being used by any unauthorised person.

Diffie–Hellman key exchange (DH) is a method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols named after Whitfield Diffie and Martin Hellman. DH is one of the earliest practical examples of public key exchange implemented within the field of cryptography.

In public key crypto system, enciphering and deciphering are governed by distinct keys, E and D, such that computing D from E is computationally infeasible (e.g., requiring more than 10^{100} instructions). The enciphering key E can thus be publicly disclosed without compromising the deciphering key D. This was the main ideology behind Diffie-Hellman Key Exchange Protocol. Each user of the network can, therefore, place his enciphering key in a public directory. This enables any user of the system to send a message to any other user enciphered in such a way that only the intended receiver can decipher it. As such, a public key crypto system is a multiple access cipher. A private conversation can therefore be held between any two individuals regardless of whether they have ever communicated before. Each one sends messages to the other enciphered in the receiver's public enciphering key and deciphers the messages he receives using his own secret deciphering key. Diffie–Hellman key exchange establishes a shared secret between two parties that can be

used for secret communication for exchanging data over a public network. The above conceptual diagram illustrates the general idea of the key exchange by using colours instead of very large numbers.

The process begins by having the two parties, Alice and Bob, agree on an arbitrary starting colour that does not need to be kept secret in this example the colour is yellow. Each of them selects a secret colour that they keep to themselves – in this case, orange and blue-green. The crucial part of the process is that Alice and Bob each mix their own secret colour together with their mutually shared colour, resulting in orange-tan and light-blue mixtures respectively, and then publicly exchange the two mixed colours. Finally, each of the two mixes the colour he or she received from the partner with his or her own private colour. The result is a final colour mixture (yellow-brown in this case) that is identical to the partner's final colour mixture. If a third party listened to the exchange, it would be computationally difficult for this party to determine the secret colours. In fact, when using large numbers rather than colours, this action is computationally expensive for modern supercomputers to do in a reasonable amount of time. Cloud security is one of the main concerns in the cloud computing domain. Storing personal and sensitive information on a third-party storage medium poses serious risks of data theft and data misuse by any person with malicious intent. The threat is so humongous that it has dissuaded governments and many other big organizations from migrating their operations on a cloud platform. The traditional methods of securing files and information are superfluous in the scenario of cloud. Extensive research and study is undergoing in this field to make cloud more secure and reliable. Among this behemoth instances of research, some of

the methods that stand out include AES encryption and Diffie Hellman Key Exchange. The latter method is so powerful that it may take millions of years for even the most powerful computers of current times to crack the code and reads the file. Our approach proposes a method that involves encrypting the file using any standard encryption technique and using Diffie Hellman for user authentication. In this way the files can be saved in a public domain securely without the threat of being used by any unauthorised person.

CHAPTER 1.1

FLUTTER

Flutter is an open-source UI software development kit created by Google. It is used to develop applications for Android, iOS, Windows, Mac, Linux, Google Fuchsia and the web.

The first version of Flutter was known as codename "Sky" and ran on the Android operating system. It was unveiled at the 2015 Dart developer summit, with the stated intent of being able to render consistently at 120 frames per second. During the keynote of Google Developer Days in Shanghai, Google announced Flutter Release Preview 2 which is the last big release before Flutter 1.0. On December 4, 2018, Flutter 1.0 was released at the Flutter Live event, denoting the first "stable" version of the Framework. On December 11, 2019, Flutter 1.12 was released at the Flutter Interactive event.

The major components of Flutter include:

- Dart platform
- Flutter engine
- Foundation library
- Design-specific widgets

Dart platform

Flutter apps are written in the Dart language and make use of many of the language's more advanced features.

On Windows, macOS and Linux via the semi-official Flutter Desktop Embedding project, Flutter runs in the Dart virtual machine which features a just-in-time execution engine. While writing and debugging an app, Flutter uses Just In Time compilation, allowing for "hot reload", with which modifications to source files can be injected into a running application. Flutter extends this with support for stateful hot reload, where in most cases changes to source code can be reflected immediately in the running app without requiring a restart or any loss of state.This feature as implemented in Flutter has received widespread praise.

Release versions of Flutter apps are compiled with ahead-of-time (AOT) compilation on both Android and iOS, making Flutter's high performance on mobile devices possible.

Flutter engine

Flutter's engine, written primarily in C++, provides low-level rendering support using Google's Skia graphics library. Additionally, it interfaces with platform-specific SDKs such as those provided by Android and iOS.The Flutter Engine is a portable runtime for hosting Flutter applications. It implements Flutter's core libraries, including animation and graphics, file and network I/O, accessibility support, plugin architecture, and a Dart runtime and compile toolchain. Most developers will interact with Flutter via the Flutter Framework, which provides a modern, reactive framework, and a rich set of platform, layout and foundation widgets.

Foundation library

The Foundation library, written in Dart, provides basic classes and functions which are used to construct applications using Flutter, such as APIs to communicate with the engine.

Widgets

UI design in Flutter involves using composition to assemble / create "Widgets" from other Widgets. The trick to understanding this is to realise that any tree of components (Widgets) that is assembled under a single build() method is also referred to as a single Widget. This is because those smaller Widgets are also made up of even smaller Widgets, and each has a build() method of its own. This is how Flutter makes use of Composition.

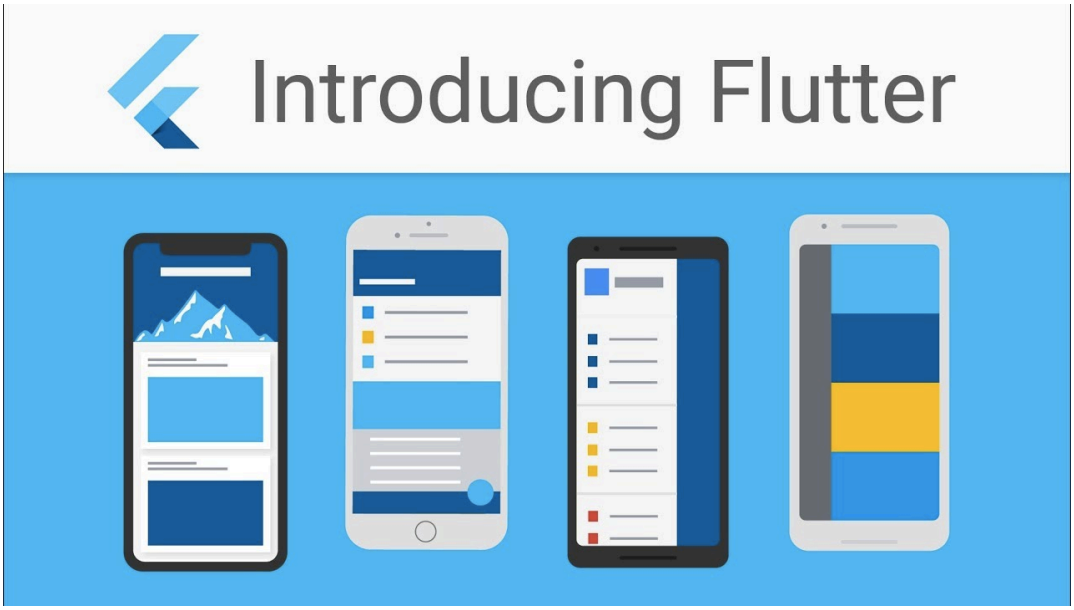
The docs say: " A widget is an immutable description of part of a user interface." A human being will tell you it's a Blueprint, which is a much easier way to think about it. However, one also needs to keep in mind there are many types of Widgets in Flutter, and you cannot see or touch all of them. Text is a Widget, but so is it’s TextStyle, which defines things like size, colour, font family and weight. There are Widgets that represent things, ones that represent characteristics (like TextStyle) and even others that do things, like FutureBuilder and StreamBuilder.

Complex widgets can be created by combining many simpler ones, and an app is actually just the largest Widget of them all (often called "MyApp"). The MyApp Widget contains all the other Widgets, which can contain even smaller Widgets, and together they make up your app.

However, the use of widgets is not strictly required to build Flutter apps. An alternative option, usually only used by people who like to control every pixel drawn to the canvas, is to use the Foundation library's methods directly. These methods can be used to draw shapes, text, and imagery directly to the canvas. This ability of Flutter has been utilised in a few frameworks, such as the open-source Flame game engine.

Design-specific widgets

The Flutter framework contains two sets of widgets which conform to specific design languages. Material Design widgets implement Google's design language of the same name, and Cupertino widgets implement Apple's iOS Human interface guidelines.



CHAPTER 1.2

GOOGLE FIREBASE

Firebase is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014. As of March 2020, the Firebase platform has 19 products, which are used by more than 1.5 million apps.

Firebase evolved from Envolve, a prior startup founded by James Tamplin and Andrew Lee in 2011. Envolve provided developers an API that enables the integration of online chat functionality into their websites. After releasing the chat service, Tamplin and Lee found that it was being used to pass application data that were not chat messages. Developers were using Envolve to sync application data such as game state in real time across their users. Tamplin and Lee decided to separate the chat system and the real-time architecture that powered it. They founded Firebase as a separate company in September 2011 and it launched to the public in April 2012.

Firebase's first product was the Firebase Real-time Database, an API that synchronizes application data across iOS, Android, and Web devices, and stores it on Firebase's cloud. The product assists software developers in building real-time, collaborative applications.

In May 2012, a month after the beta launch, Firebase raised $1.1 million in seed funding from venture capitalists Flybridge Capital Partners, Greylock Partners, Founder Collective, and New Enterprise Associates. In June 2013, the company

further raised $5.6 million in Series A funding from Union Square Ventures and Flybridge Capital Partners.

In 2014, Firebase launched two products. Firebase Hosting and Firebase Authentication. This positioned the company as a mobile backend as a service.

In October 2014, Firebase was acquired by Google. A year later, in October 2015, Google acquired Divshot, an HTML5 web-hosting platform, to merge it with the Firebase team.

In May 2016, at Google I/O, the company's annual developer conference, Firebase introduced Firebase Analytics and announced that it was expanding its services to become a unified backend-as-a-service (BaaS) platform for mobile developers. Firebase now integrates with various other Google services, including Google Cloud Platform, AdMob, and Google Ads to offer broader products and scale for developers. Google Cloud Messaging, the Google service to send push notifications to Android devices, was superseded by a Firebase product, Firebase Cloud Messaging, which added the functionality to deliver push notifications to both iOS and web devices. In January 2017, Google acquired Fabric and Crashlytics from Twitter to add those services to Firebase.

In October 2017, Firebase has launched Cloud Firestore, a real-time document database as the successor product to the original Firebase Realtime Database.

Services

Google Analytics

Google Analytics is a cost-free app measurement solution that provides insights on app usage and user engagement.

Develop

Firebase Cloud Messaging

Formerly known as Google Cloud Messaging (GCM), Firebase Cloud Messaging (FCM) is a cross-platform solution for messages and notifications for Android, iOS, and web applications, which as of 2016 can be used at no cost.

Firebase Authentication

Firebase Authentication is a service that can authenticate users using only client-side code. It supports social login providers Facebook, GitHub, Twitter and Google as well as other service providers like Google Play Games, Apple, Yahoo, and Microsoft. Additionally, it includes a user management system whereby developers can enable user authentication with email and password login stored with Firebase.

Firebase Realtime Database

Firebase provides a real-time database and back-end as a service. The service provides application developers an API that allows application data to be synchronized across clients and stored on Firebase's cloud. The company provides client libraries that enable integration with Android, iOS, JavaScript, Java, Objective-C, Swift and Node.js applications. The database is also accessible through a REST API and bindings for several JavaScript frameworks such as AngularJS, React, Ember.js and Backbone.js. The REST API uses the Server-Sent

Events protocol, which is an API for creating HTTP connections for receiving push notifications from a server. Developers using the realtime database can secure their data by using the company's server-side-enforced security rules.

Cloud Firestore

On January 31, 2019, Cloud Firestore was officially brought out of beta, making it an official product of the Firebase lineup. It is the successor to Firebase's original databasing system, Real-time Database, and allows for nested documents and fields rather than the tree-view provided in the Real-time Database.

Firebase Storage

Firebase Storage provides secure file uploads and downloads for Firebase apps, regardless of network quality, to be used for storing images, audio, video, or other user-generated content. It is backed by Google Cloud Storage.

Firebase Hosting

Firebase Hosting is a static and dynamic web hosting service that launched on May 13, 2014. It supports hosting static files such as CSS, HTML, JavaScript and other files, as well as support through Cloud Functions. The service delivers files over a content delivery network (CDN) through HTTP Secure (HTTPS) and Secure Sockets Layer encryption (SSL). Firebase partners with Fastly, a CDN, to provide the CDN backing Firebase Hosting. The company states that Firebase Hosting grew out of customer requests; developers were using Firebase for its real-time database but needed a place to host their content.

ML Kit

ML Kit is a mobile machine learning system for developers launched on May 8, 2018, in beta during the Google I/O 2018. ML Kit APIs feature a variety of features including optical character recognition, detecting faces, scanning barcodes, labelling images and recognising landmarks. It is currently available for iOS or Android developers. You may also import your own TensorFlow Lite models, if the given APIs are not enough.The APIs can be used on-device or on-cloud.

Stability

Crashlytics

Crash Reporting creates detailed reports of the errors in the app. Errors are grouped into clusters of similar stack traces and triaged by the severity of impact on app users. In addition to automatic reports, the developer can log custom events to help capture the steps leading up to a crash. Before acquiring Crashlytics, Firebase was using its own Firebase Crash Reporting.

Performance

Firebase Performance provides insights into an app's performance and the latencies the app's users experience.

Firebase Test Lab

Firebase Test Lab provides cloud-based infrastructure for testing Android and iOS apps in one operation. Developers can test their apps across a wide variety of devices and device configurations. Test results—including logs, videos, and screenshots—are

made available in the Firebase console. Even if a developer hasn't written any test code for their app, Test Lab can exercise the app automatically, looking for crashes. Test Lab for iOS is currently in beta.

Admob

Admob is a Google product that integrates with Firebase audience.

Grow

Firebase Dynamic Links

Dynamic Firebase links are smart URLs that dynamically change their behaviour to provide "the best available experience" across multiple platforms, including desktop web browsers, iOS, and Android, and in-depth links to mobile apps. Dynamic Links work in all app installs: if the user opens Dynamic Link on iOS or Android and the application is not installed, the user will be prompted to install the app first. Once installed, the application will start running and can access the link.

Fig:2 Google Firebase

CHAPTER 1.3

GOOGLE CLOUD

Google Cloud Platform (**GCP**), offered by Google, is a suite of cloud computing services that runs on the same infrastructure that Google uses internally for its end-user products, such as Google Search, Gmail and YouTube.Alongside a set of management tools, it provides a series of modular cloud services including computing, data storage, data analytics and machine learning.Registration requires a credit card or bank account details.

Google Cloud Platform provides infrastructure as a service, platform as a service, and serverless computing environments.

In April 2008, Google announced App Engine, a platform for developing and hosting web applications in Google-managed data centers, which was the first cloud computing service from the company. The service became generally available in November 2011. Since the announcement of the App Engine, Google added multiple cloud services to the platform.

Google Cloud Platform is a part of **Google Cloud**, which includes the Google Cloud Platform public cloud infrastructure, as well as **G Suite**, enterprise versions of Android and Chrome OS, and application programming interfaces (APIs) for machine learning and enterprise mapping services.

Products

Google lists over 90 products under the Google Cloud brand. Some of the key services are listed below.

Compute

- App Engine - Platform as a Service to deploy Java, PHP, Node.js, Python, C#, .Net, Ruby and Go applications.
- Compute Engine - Infrastructure as a Service to run Microsoft Windows and Linux virtual machines.
- Kubernetes Engine (GKE) or GKE on-prem offered as part of Anthos platform- Containers as a Service based on Kubernetes.
- Cloud Functions - Functions as a Service to run event-driven code written in Node.js, Python or Go.
- Cloud Run - Compute execution environment based on Knative. Offered as Cloud Run (fully managed) or as Cloud Run for Anthos.

Storage & Databases

- Cloud Storage - Object storage with integrated edge caching to store unstructured data.
- Cloud SQL - Database as a Service based on MySQL and PostgreSQL.
- Cloud Bigtable - Managed NoSQL database service.
- Cloud Spanner - Horizontally scalable, strongly consistent, relational database service.
- Cloud Datastore - NoSQL database for web and mobile applications.
- Persistent Disk - Block storage for Compute Engine virtual machines.
- Cloud MemoryStore - Managed in-memory data store based on Redis.

- Local SSD: High performance, transient, local block storage.
- Filestore: High performance file storage for Google Cloud users.

Networking

- VPC - Virtual Private Cloud for managing the software defined network of cloud resources.
- Cloud Load Balancing - Software-defined, managed service for load balancing the traffic.
- Cloud Armor - Web application firewall to protect workloads from DDoS attacks.
- Cloud CDN - Content Delivery Network based on Google's globally distributed edge points of presence.
- Cloud Interconnect - Service to connect a data center with Google Cloud Platform
- Cloud DNS - Managed, authoritative DNS service running on the same infrastructure as Google.
- Network Service Tiers - Option to choose Premium vs Standard network tier for higher-performing network.

Big Data

- BigQuery - Scalable, managed enterprise data warehouse for analytics.
- Cloud Dataflow - Managed service based on Apache Beam for stream and batch data processing.
- Cloud Dataproc - Big data platform for running Apache Hadoop and Apache Spark jobs.

- Cloud Composer - Managed workflow orchestration service built on Apache Airflow.
- Cloud Datalab - Tool for data exploration, analysis, visualization and machine learning. This is a fully managed Jupyter Notebook service.
- Cloud Dataprep - Data service based on Trifacta to visually explore, clean, and prepare data for analysis.
- Cloud Pub/Sub - Scalable event ingestion service based on message queues.
- Cloud Data Studio - Business intelligence tool to visualize data through dashboards and reports.

Cloud AI

- Cloud AutoML - Service to train and deploy custom machine, learning models. As of September 2018, the service is in Beta.
- Cloud TPU - Accelerators used by Google to train machine learning models.
- Cloud Machine Learning Engine - Managed service for training and building machine learning models based on mainstream frameworks.
- Cloud Job Discovery - Service based on Google's search and machine learning capabilities for the recruiting ecosystem.
- Dialogflow Enterprise - Development environment based on Google's machine learning for building conversational interfaces.
- Cloud Natural Language - Text analysis service based on Google Deep Learning models.
- Cloud Speech-to-Text - Speech to text conversion service based on machine learning.

- Cloud Text-to-Speech - Text to speech conversion service based on machine learning.
- Cloud Translation API - Service to dynamically translate between thousands of available language pairs
- Cloud Vision API - Image analysis service based on machine learning
- Cloud Video Intelligence - Video analysis service based on machine learning

Management Tools

- Stackdriver - Monitoring, logging, and diagnostics for applications on Google Cloud Platform and AWS.
- Cloud Deployment Manager - Tool to deploy Google Cloud Platform resources defined in templates created in YAML, Python or Jinja2.
- Cloud Console - Web interface to manage Google Cloud Platform resources.
- Cloud Shell - Browser-based shell command-line access to manage Google Cloud Platform resources.
- Cloud Console Mobile App - Android and iOS application to manage Google Cloud Platform resources.
- Cloud APIs - APIs to programmatically access Google Cloud Platform resources

Identity & Security

- Cloud Identity - Single sign-on (SSO) service based on SAML 2.0 and OpenID.
- Cloud IAM - Identity & Access Management (IAM) service for defining policies based on role-based access control.

- Cloud Identity-Aware Proxy - Service to control access to cloud applications running on Google Cloud Platform without using a VPN.
- Cloud Data Loss Prevention API - Service to automatically discover, classify, and redact sensitive data.
- Security Key Enforcement - Two-step verification service based on a security key.
- Cloud Key Management Service - Cloud-hosted key management service integrated with IAM and audit logging.
- Cloud Resource Manager - Service to manage resources by project, folder, and organization based on the hierarchy.
- Cloud Security Command Center - Security and data risk platform for data and services running in Google Cloud Platform.
- Cloud Security Scanner - Automated vulnerability scanning service for applications deployed in App Engine.
- Access Transparency - Near real-time audit logs providing visibility to Google Cloud Platform administrators.
- VPC Service Controls - Service to manage security perimeters for sensitive data in Google Cloud Platform services.

IoT

- Cloud IoT Core - Secure device connection and management service for Internet of Things.
- Edge TPU - Purpose-built ASIC designed to run inference at the edge. As of September 2018, this product is in private beta.

- Cloud IoT Edge - Brings AI to the edge computing layer.

API Platform

- Maps Platform - APIs for maps, routes, and places based on Google Maps.
- Apigee API Platform - Lifecycle management platform to design, secure, deploy, monitor, and scale APIs.
- API Monetization - Tool for API providers to create revenue models, reports, payment gateways, and developer portal integrations.
- Developer Portal - Self-service platform for developers to publish and manage APIs.
- API Analytics - Service to analyze API-driven programs through monitoring, measuring, and managing APIs.
- Apigee Sense - Enables API security by identifying and alerting administrators to suspicious API behaviors.
- Cloud Endpoints - An NGINX-based proxy to deploy and manage APIs.
- Service Infrastructure - A set of foundational services for building Google Cloud products.

Regions and zones

As of Q1 2020, Google Cloud Platform is available in 22 regions and 61 zones. A region is a specific geographical location where users can deploy cloud resources. Each region is an independent geographic area that consists of zones. A zone is a deployment area for Google Cloud Platform resources within a region. Zones should be considered a single failure domain within a region. Most of the regions have three

or more zones. As of Q1 2020, Google Cloud Platform is available in the following regions and zones:

GCP Regions & Zones

Region Name	Launch Date	Location	Zones
us-west1	Q3, 2016	The Dalles, Oregon, USA	• us-west1-a • us-west1-b • us-west1-c
us-west2	Q3, 2018	Los Angeles, California, USA	• us-west2-a • us-west2-b • us-west2-c
us-west3	Q1, 2020	Salt Lake City, Utah, USA	• us-west3-a • us-west3-b • us-west3-c
us-central1		Council Bluffs, Iowa, USA	• us-central1-a • us-central1-b • us-central1-c • us-central1-f
us-east1	Q4, 2015	Moncks Corner, South Carolina, USA	• us-east1-b • us-east1-c • us-east1-d
us-east4	Q2, 2017	Ashburn, Virginia, USA	• us-east4-a • us-east4-b • us-east4-c
northamerica-northeast1	Q1, 2018	Montréal, Canada	• northamerica-northeast1-a • northamerica-northeast1-b • northamerica-northeast1-c

southamerica-east1	Q3, 2017	São Paulo, Brazil	• southamerica-east1-a • southamerica-east1-b • southamerica-east1-c
europe-west2	Q2, 2017	London, U.K.	• europe-west2-a • europe-west2-b • europe-west2-c
europe-west1		St. Ghislain, Belgium	• europe-west1-b • europe-west1-c • europe-west1-d
europe-west4	Q1, 2018	Eemshaven, Netherlands	• europe-west4-a • europe-west4-b • europe-west4-c
europe-west6	Q1, 2019	Zurich, Switzerland	• europe-west6-a • europe-west6-b • europe-west6-c
europe-west3	Q3, 2017	Frankfurt, Germany	• europe-west3-a • europe-west3-b • europe-west3-c
europe-north1	Q2, 2018	Hamina, Finland	• europe-north1-a • europe-north1-b • europe-north1-c
asia-south1	Q4, 2017	Mumbai, India	• asia-south1-a • asia-south1-b • asia-south1-c
asia-southeast1	Q2, 2017	Jurong West, Singapore	• asia-southeast1-a • asia-southeast1-b • asia-southeast1-c
asia-east2	Q3, 2018	Hong Kong	• asia-east2-a • asia-east2-b • asia-east2-c

asia-east1		Changhua County, Taiwan	• asia-east1-a • asia-east1-b • asia-east1-c
asia-northeast1	Q4, 2016	Tokyo, Japan	• asia-northeast1-a • asia-northeast1-b • asia-northeast1-c
asia-northeast2	Q2, 2019	Osaka, Japan	• asia-northeast2-a • asia-northeast2-b • asia-northeast2-c
asia-northeast3	Q1, 2020	Seoul, Korea	• asia-northeast3-a • asia-northeast3-b • asia-northeast3-c
australia-southeast1	Q3, 2017	Sydney, Australia	• australia-southeast1-a • australia-southeast1-b • australia-southeast1-c

Similarity to services by other cloud service providers

For those familiar with other notable cloud service providers, a comparison of similar services may be helpful in understanding Google Cloud Platform's offerings.

Google Cloud Platform	Amazon Web Services	Microsoft Azure	Oracle Cloud
Google Compute Engine	Amazon EC2	Azure Virtual Machines	Oracle Cloud Infra OCI
Google App Engine	AWS Elastic Beanstalk	Azure App Services	Oracle Application Container
Google Kubernetes Engine	Amazon Elastic Container Service for Kubernetes	Azure Kubernetes Service	Oracle Kubernetes Service
Google Cloud Bigtable	Amazon DynamoDB	Azure Cosmos DB	Oracle NoSQL Database
Google BigQuery	Amazon Redshift	Microsoft Azure DataWarehouse	Oracle Autonomous DataWarehouse
Google Cloud Functions	AWS Lambda	Azure Functions	Oracle Cloud Fn
Google Cloud Datastore	Amazon DynamoDB	Cosmos DB	Oracle NoSQL Database
Google Cloud Storage	Amazon S3	Azure Blob Storage	Oracle Cloud Storage OCI

CHAPTER 1.4
AMAZON WEB SERVICES(AWS)

Amazon Web Services (**AWS**) is a subsidiary of Amazon that provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered pay-as-you-go basis. In aggregate, these cloud computing web services provide a set of primitive abstract technical infrastructure and distributed computing building blocks and tools. One of these services is Amazon Elastic Compute Cloud, which allows users to have at their disposal a virtual cluster of computers, available all the time, through the Internet. AWS's version of virtual computers emulate most of the attributes of a real computer, including hardware central processing units (CPUs) and graphics processing units (GPUs) for processing; local/RAM memory; hard-disk/SSD storage; a choice of operating

systems; networking; and pre-loaded application software such as web servers, databases, and customer relationship management (CRM).

The AWS technology is implemented at server farms throughout the world, and maintained by the Amazon subsidiary. Fees are based on a combination of usage (known as a "Pay-as-you-go" model), the hardware/OS/software/networking features chosen by the subscriber, required availability, redundancy, security, and service options. Subscribers can pay for a single virtual AWS computer, a dedicated physical computer, or clusters of either. As part of the subscription agreement, Amazon provides security for subscribers' systems. AWS operates from many global geographical regions including 6 in North America.

In 2020, AWS comprised more than 212 services including computing, storage, networking, database, analytics, application services, deployment, management, mobile, developer tools, and tools for the Internet of Things. The most popular include Amazon Elastic Compute Cloud (EC2) and Amazon Simple Storage Service (Amazon S3). Most services are not exposed directly to end users, but instead offer functionality through APIs for developers to use in their applications. Amazon Web Services' offerings are accessed over HTTP, using the REST architectural style and SOAP protocol for older APIs and exclusively JSON for newer ones.

Amazon markets AWS to subscribers as a way of obtaining large scale computing capacity more quickly and cheaply then building an actual physical server farm. All services are billed based on usage, but each service measures usage in varying ways. As of 2017, AWS owns a dominant 34% of all cloud (IaaS, PaaS) while the next three

competitors Microsoft, Google, and IBM have 11%, 8%, 6% respectively according to Synergy Group.

The AWS platform was launched in July 2002. In its early stages, the platform consisted of only a few disparate tools and services. Then in late 2003, the AWS concept was publicly reformulated when Chris Pinkham and Benjamin Black presented a paper describing a vision for Amazon's retail computing infrastructure that was completely standardized, completely automated, and would rely extensively on web services for services such as storage and would draw on internal work already underway. Near the end of their paper, they mentioned the possibility of selling access to virtual servers as a service, proposing the company could generate revenue from the new infrastructure investment. In November 2004, the first AWS service launched for public usage: Simple Queue Service (SQS). Thereafter Pinkham and lead developer Christopher Brown developed the Amazon EC2 service, with a team in Cape Town, South Africa.

Amazon Web Services was officially re-launched on March 14, 2006, combining the three initial service offerings of Amazon S3 cloud storage, SQS, and EC2. The AWS platform finally provided an integrated suite of core online services, as Chris Pinkham and Benjamin Black had proposed back in 2003, as a service offered to other developers, web sites, client-side applications, and companies. Andy Jassy, AWS founder and vice president in 2006, said at the time that Amazon S3 (one of the first and most scalable elements of AWS) "helps free developers from worrying about where they are going to store data, whether it will be safe and secure, if it will be available when they need it, the costs associated with server maintenance, or whether

they have enough storage available. Amazon S3 enables developers to focus on innovating with data, rather than figuring out how to store it.".In 2016 Jassy was promoted to CEO of the division. Reflecting the success of AWS, his annual compensation in 2017 hit nearly $36 million.

In 2014, AWS launched its partner network entitled APN (AWS Partner Network) which is focused on helping AWS-based companies grow and scale the success of their business with close collaboration and best practices.[17]

To support industry-wide training and skills standardization, AWS began offering a certification program for computer engineers, on April 30, 2013, to highlight expertise in cloud computing.

In January 2015, Amazon Web Services acquired Annapurna Labs, an Israel-based microelectronics company reputedly for US$350–370M.

James Hamilton, an AWS engineer, wrote a retrospective article in 2016 to highlight the ten-year history of the online service from 2006 to 2016. As an early fan and outspoken proponent of the technology, he had joined the AWS engineering team in 2008.

In January 2018, Amazon launched an autoscaling service on AWS.

In November 2018, AWS announced customized ARM cores for use in its servers. Also in November 2018, AWS is developing ground stations to communicate with customer's satellites.

Growth and profitability

In November 2010, it was reported that all of Amazon.com's retail sites had migrated to AWS. Prior to 2012, AWS was considered a part of Amazon.com and so its revenue was not delineated in Amazon financial statements. In that year industry watchers for the first time estimated AWS revenue to be over $1.5 billion.

In April 2015, Amazon.com reported AWS was profitable, with sales of $1.57 billion in the first quarter of the year and $265 million of operating income. Founder Jeff Bezos described it as a fast-growing $5 billion business; analysts described it as "surprisingly more profitable than forecast". In October, Amazon.com said in its Q3 earnings report that AWS's operating income was $521 million, with operating margins at 25 percent. AWS's 2015 Q3 revenue was $2.1 billion, a 78% increase from 2014's Q3 revenue of $1.17 billion. 2015 Q4 revenue for the AWS segment increased 69.5% y/y to $2.4 billion with 28.5% operating margin, giving AWS a $9.6 billion run rate. In 2015, Gartner estimated that AWS customers are deploying 10x more infrastructure on AWS than the combined adoption of the next 14 providers.

In 2016 Q1, revenue was $2.57 billion with net income of $604 million, a 64% increase over 2015 Q1 that resulted in AWS being more profitable than Amazon's North American retail business for the first time. In the first quarter of 2016, Amazon experienced a 42% rise in stock value as a result of increased earnings, of which AWS contributed 56% to corporate profits.[32]

AWS had $17.46 billion in annual revenue in 2017. By end of 2018, the number had grown to $25.65 billion.

In 2019, AWS continued to surpass its parent company in terms of profitability; Amazon reported 20% growth in sales while AWS reported 37% growth in 2019. In 2019, AWS alone accounted for 12% of Amazon's revenue (up from 11% in 2018).

Customer base

- On March 14, 2006, Amazon said in a press release: "More than 150,000 developers have signed up to use Amazon Web Services since its inception."
- In November 2012, AWS hosted its first customer event in Las Vegas.
- On May 13, 2013, AWS was awarded an Agency Authority to Operate (ATO) from the U.S. Department of Health and Human Services under the Federal Risk and Authorization Management Program.
- In October 2013, it was revealed that AWS was awarded a $600M contract with the CIA.
- During August 2014, AWS received Department of Defense-Wide provisional authorization for all U.S. Regions.
- During the 2015 re:Invent keynote, AWS disclosed that they have more than a million active customers every month in 190 countries, including nearly 2,000 government agencies, 5,000 education institutions and more than 17,500 nonprofits.
- On April 5, 2017, AWS and DXC Technology (formed from a merger of CSC and HPE's Enterprise Services Business) announced an expanded alliance to increase access of AWS features for enterprise clients in existing data centers.

Notable customers include NASA, the Obama presidential campaign of 2012, and Netflix.

In 2019, it was reported that more than 80% of Germany's listed DAX companies use AWS.

In August 2019, the U.S. Navy said it moved 72,000 users from six commands to an AWS cloud system as a first step toward pushing all of its data and analytics onto the cloud.

Significant service outages

- On April 20, 2011, AWS suffered a major outage. Parts of the Elastic Block Store (EBS) service became "stuck" and could not fulfill read/write requests. It took at least two days for service to be fully restored.
- On June 29, 2012, several websites that rely on Amazon Web Services were taken offline due to a severe storm in Northern Virginia, where AWS' largest data center cluster is located.
- On October 22, 2012, a major outage occurred, affecting many sites such as Reddit, Foursquare, Pinterest, and others. The cause was a memory leak bug in an operational data collection agent.
- On December 24, 2012, AWS suffered another outage causing websites such as Netflix to be unavailable for customers in the Northeastern United States. AWS cited their Elastic Load Balancing (ELB) service as the cause.
- On February 28, 2017, AWS experienced a massive outage of S3 services in its Northern Virginia region. A majority of websites which relied on AWS S3 either hung or stalled, and Amazon reported within five hours that AWS was fully online again. No data has been reported to have been lost due to the outage. The outage was caused by a human error made while debugging, that

resulted in removing more server capacity than intended, which caused a domino effect of outages.

Availability and topology

As of 2019, AWS has distinct operations in 22 geographical "regions": 7 in North America, 1 in South America, 5 in Europe, 1 In Middle-East and 8 in Asia Pacific.

AWS has announced 5 new regions that will be coming online in Italy, South Africa, Spain, Osaka and Indonesia.

Each region is wholly contained within a single country and all of its data and services stay within the designated region. Each region has multiple "Availability Zones", which consist of one or more discrete data centers, each with redundant power, networking and connectivity, housed in separate facilities. Availability Zones do not automatically provide additional scalability or redundancy within a region, since they are intentionally isolated from each other to prevent outages from spreading between Zones. Several services can operate across Availability Zones (e.g., S3, DynamoDB) while others can be configured to replicate across Zones to spread demand and avoid downtime from failures.

As of December 2014, Amazon Web Services operated an estimated 1.4 million servers across 28 availability zones. The global network of AWS Edge locations consists of 54 points of presence worldwide, including locations in the United States, Europe, Asia, Australia, and South America.

In 2014, AWS claimed its aim was to achieve 100% renewable energy usage in the future. In the United States, AWS's partnerships with renewable energy providers

include Community Energy of Virginia, to support the US East region; Pattern Development, in January 2015, to construct and operate Amazon Wind Farm Fowler Ridge; Iberdrola Renewables, LLC, in July 2015, to construct and operate Amazon Wind Farm US East; EDP Renewables North America, in November 2015, to construct and operate Amazon Wind Farm US Central; and Tesla Motors, to apply battery storage technology to address power needs in the US West (Northern California) region.

CHAPTER .15

DIFFIE HELLMAN

Diffie–Hellman key exchange (DH) is a method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols named after Whitfield Diffie and Martin Hellman. DH is one of the earliest practical examples of public key exchange implemented within the field of cryptography.

In public key cryptosystem, enciphering and deciphering are governed by distinct keys, E and D, such that computing D from E is computationally infeasible (e.g., requiring more than 10^{100} instructions). The enciphering key E can thus be publicly disclosed without compromising the deciphering key D. This was the main ideology behind Diffie-Hellman Key Exchange Protocol. Each user of the network can, therefore, place his enciphering key in a public directory. This enables any user of the system to send a message to any other user enciphered in such a way that only the intended receiver can decipher it. As such, a public key cryptosystem is a multiple access cipher. A private conversation can therefore be held between any two individuals regardless of whether they have ever communicated before. Each one sends messages to the other enciphered in the receiver's public enciphering key and deciphers the messages he receives using his own secret deciphering key. Diffie–Hellman key exchange establishes a shared secret between two parties that can be used for secret communication for exchanging data over a public network. The above conceptual diagram illustrates the general idea of the key exchange by using colours

instead of very large numbers.

The process begins by having the two parties, Alice and Bob, agree on an arbitrary starting colour that does not need to be kept secret in this example the color is yellow. Each of them selects a secret color that they keep to themselves – in this case, orange and blue-green. The crucial part of the process is that Alice and Bob each mix their own secret color together with their mutually shared color, resulting in orange-tan and light-blue mixtures respectively, and then publicly exchange the two mixed colours. Finally, each of the two mixes the color he or she received from the partner with his or her own private color. The result is a final color mixture (yellow-brown in this case) that is identical to the partner's final color mixture. If a third party listened to the exchange, it would be computationally difficult for this party to determine the secret colors. In fact, when using large numbers rather than colors, this action is computationally expensive for modern supercomputers to do in a reasonable amount of time.

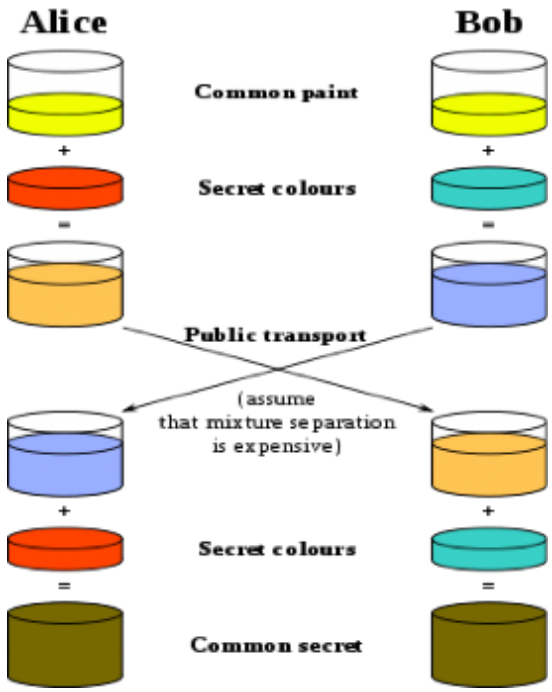


Figure 1: Illustration of idea behind Diffie-Hellman

CHAPTER 1.6

PRIME NUMBER

A prime number (or a prime) is a natural number greater than 1 that cannot be formed by multiplying two smaller natural numbers.

The only user-defined pre-existing parameter in the Diffie-Hellman protocol is the selection of prime number. The prime number p should be large enough to defend against the known attacks against it. The most efficient attack is NFS (attack on the network file system); that has been used against numbers on the order of 2^768 (a 232-digit number). It would appear wise to pick a p that's considerably bigger than that; around 1024 bits at a minimum, and more realistically at least 1536 bits. Another property about p is that p−1 should have a large prime factor q, and one should know what the factorization of p−1 is. If we pick a random prime p, and a random generator g, well, we’re probably secure, but we won't be certain (and we might leak a few bits of the private exponent if the order of your random g happens to have some small factors).

CHAPTER 1.7

METHOD

Follow the mathematical implementation of Diffie Hellman key exchange protocol. *p* is a prime number.

g is a primitive root modulo of *p*

1. Alice and Bob agree to use a modulus *p* = 23 and base g = 5

2. Alice gets her private key (key which she should not share with anyone) generated as 4. 3. Thus, public key generated for Alice shall be 5^4%23 = 625%23 = 4

4. Bob gets his private key (key which he should not share with anyone) generated as 3. 5. Thus, public key generated for Bob shall be 5^3%23 = 125%23 = 10

6. Now, Alice gets the public key of Bob and generates a secret key. i.e. p

=> (10^4) % 23 => 10000 % 23 => 18

7. On the other side, Bob also uses a similar method to generate a secret key i.e.

=> (4^3) % 23 => 64 % 23 => 18

Thus, it is proven that mathematically, Alice and Bob generate the same key without each one of them knowing other one's private key. This is the implementation of Diffie-Hellman Key Exchange Protocol.

CHAPTER 1.8

ENCRYPTION

Encryption is widely used on the internet to protect user information being sent between a browser and a server, including passwords, payment information and other personal information that should be considered private. Organizations and individuals also commonly use encryption to protect sensitive data stored on computers, servers and mobile devices like phones or tablets. There are various encryption techniques that are present some of which are:

A) Triple DES

B) RSA

C) Blowfish

D) Two fish

E) AES

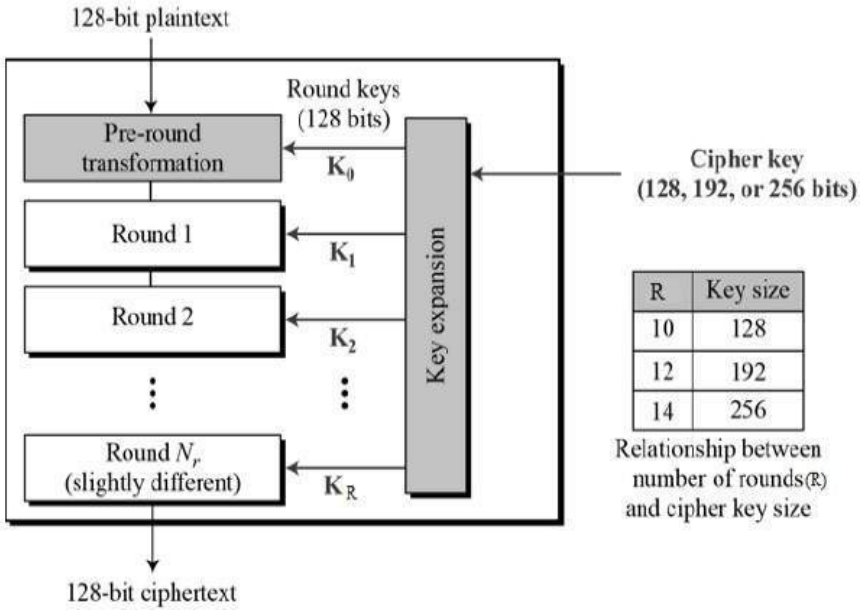
The technique that we have used in our project is AES and it is described below.

CHAPTER 1.9

ADVANCED ENCRYPTION ALGORITHM

The more popular and widely adopted symmetric encryption algorithm nowadays is the Advanced Encryption Standard (AES). It is found to be at least six time faster than triple DES. A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback, but it was found to be slow.

The AES has hree fixed 128-bit block ciphers with cryptographic key sizes of 128, 192 and 256 bits. Key size is unlimited, whereas the block size maximum is 256 bits. The AES design is based on a substitution-permutation network (SPN) and does not use the Data Encryption Standard (DES) Feistel network. The diagram below shows the implementation of AES encryption technique.



R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

Figure 2: The schematic of AES structure

CHAPTER 1.10

PYTHON

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespaces. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative and procedural, and has a large and comprehensive standard library.

Such large and comprehensive standard libraries along with the documented support helped us to choose python as the language which we used to build both, the stand alone as well as web-based application.

Below is a brief description of the frameworks and libraries extensively used to build the desired framework.

CHAPTER 1.11

PYTHON FLASK

Flask is a BSD licensed microframework for Python based on Werkzeug and Jinja 2. “Micro” does not mean that your whole web application must fit into a single Python file (although it certainly can), nor does it mean that Flask is lacking in functionality. The “micro” in microframework means Flask aims to keep the core simple but extensible. Flask won’t make many decisions for you, such as what database to use. Those decisions that it does make, such as what templating engine to use, are easy to change. Everything else is up to you, so that Flask can be everything you need and nothing you don’t.

By default, Flask does not include a database abstraction layer, form validation or anything else where different libraries already exist that can handle that. Instead, Flask supports extensions to add such functionality to your application as if it was implemented in Flask itself. Numerous extensions provide database integration, form validation, upload handling, various open authentication technologies, and more. Flask may be “micro”, but it’s ready for production use on a variety of needs. Flask has many configuration values, with sensible defaults, and a few conventions when getting started. By convention, templates and static files are stored in subdirectories within the application’s Python source tree, with the names templates and static respectively. While this can be changed, you usually don’t have to, especially when getting started.

CHAPTER 1.12

ELASTIC CLOUD COMPUTE [EC2]

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers.

Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change. Amazon EC2 changes the economics of computing by allowing you to pay only for capacity that you actually use. Amazon EC2 provides developers the tools to build failure resilient applications and isolate them from common failure scenarios. For compute, AWS' main offering is its EC2 instances, which can be tailored with a large number of options. It also provides related services such as Elastic Beanstalk for app deployment, the EC2 Container service, AWS Lambda and Autoscaling. In general terms, prices are roughly comparable, especially since AWS shifter from by-the-hour to by-the-second pricing for its EC2 and EBS services in 2017. This economic pricing, reliable and secure infrastructure and vast online support enabled us to use Amazon to host and deploy the service on it is IaaS platform.

CHAPTER 1.13

Proposed Model

1. HYBRID APP

User Module The User Module is for the registration of the farmers and the client. There will be two registration pages one for the farmer and the other for the client. After sign up the user will be directed to the login page from where he needs to log in and lands to the home page of the app. Farmer and Client both will have a different home screen where they will have a feature to share crops details and can be able to see the daily prices and notification about the crops

Home Screen Home Screen is the main page of this Application. this module shows some category options to the user to their choice. By clicking someone option a user gets the main page of that particular category which is easy to use by the user. As we already told, there will be two separate home screens for the farmer as well as for the client. In Farmer home screen It shows three types of options

- Customer Notification/Details
- Sell Crops
- Daily Rate Update and Daily News Update by GOI
- Framer Details
- Buy Crops

- Share and see details related to crops

CustomerNotification/Details In this module farmer can see the details of the client who requested or looking for the crops in his nearby areas, so the customer details will be shared with the farmer and from there the farmer can contact the client directly without any intervention of any other person all these details will get stored on the google cloud using google firebase database and API.

Sell Crops The App provides a special section for selling the crops, in this sections a farmer can update the details of the crops which he wants to sell and it will get store on our cloud and also get reflected in the farmer notification and the buy crops section of the client home screen. The client can fetch the details of the crops and the person associated with the crops and deal directly with the farmer

Daily Rate Update and Crops News In the farmer section there will be a special sections, for the daily rate of the crops and the news related to the crops, we are tracking these details from a government of India API reflect these details using HTTP request using Alamo fire and JSON. The Code and Implementation of the Hybrid App can be found here

2. Storing File using Diffie Hellman on AWS

Diffie-Hellman: The Diffie-Hellman algorithm was one of the asymmetric key exchange algorithm. This algorithm is mostly used for key exchange. Although symmetric algorithms are fast and secure, but keeping the key secure and exchange is always a problem. You have to make sure to get the private key to all systems. This algorithm helps with this. The Diffie-Hellman algorithm establishes a secure

communication channel. This channel is used to exchange a private key. And then this private key is used to do symmetric encryption between the two systems.

The web application of the project is used for secure storage on the cloud. The major steps included in the web application are as follows:

- The first step for the user is to register on our web app . On registering the user would be given an automatically generated private key that would be used by the user for transactions and the file storage. For security purposes the private key of the user is not stored in the database.

- On selecting the upload file option the user is taken to another page that allows the user to submit the encrypted file.

- Clicking the file directory option on the page takes the user to the above page which displays all the different files stored on the cloud.

- Selecting different options like download-public key lists the public keys of the registered users which the user can download and use for authenticating the user when someone tries to open his uploaded file.

- Clicking on register user tab take the user to a page where the user registers himself with the platform.

For this purpose we are using amazon web services to host a server and that server will generate a public key pair and with the help of putty and putty generation that

file is going to convert into a private key and using that private key we encrypt our file using Diffie Hellman Algorithm.

CHAPTER 2

LITERATURE REVIEW

A. The Modern Farming Techniques using Android Application. The main awareness of this work is focused on Indian farmers as it addresses the key problems of getting the market and wether updates of different products, and information about the rain and also provides multiple language support. Annually, such loss exceeds 40% in total. So, the paper presented here suggest various ways in which a farmer can utilise mobile on their handsets using application to assist them for relatively better cultivation and merchandise

B. Mahafarm is an app which talks about Information and Communication Technology (ICT) in agriculture is an emerging field which is focusing on the enhancement of agriculture sector and rural development in India. Using innovation is a key and vital measure in the rural domain. The advancement of ICT can be utilised for providing accurate and timely relevant information and services related to agriculture to the farmers, thereby facilitating an environment for remunerative agriculture. This paper describes a mobile application for farmers which would exhaustively help them in their farming

C. E-Agro Android Application this paper talks about software application is basically for sustainable development of farmers. Many times farmer is confused to take decisions regarding selection of fertiliser, pesticide and time to do particular farming actions. To avoid this problem this application is very helpful. Fertiliser schedule of every form of crop can get registered. Based on

sowing date of crop, farmer will get reminders about application of fertiliser, herbicide as per schedule, pesticide for diseases alerts and weather alerts if particular crop exceeds its favourable temperature range. Crop suggestion are given supported Soil kind, geographical location. Farmer will get real time crop rates to get more benefit. This system combines fashionable net and mobile communication systems with GPS for economical and sleek farming. This paper presents the introduction, theories and deep analysis of DBMS, use of Smartphone in agriculture. This papers developed on brief study of some common problems faced by the farmers across the nation.

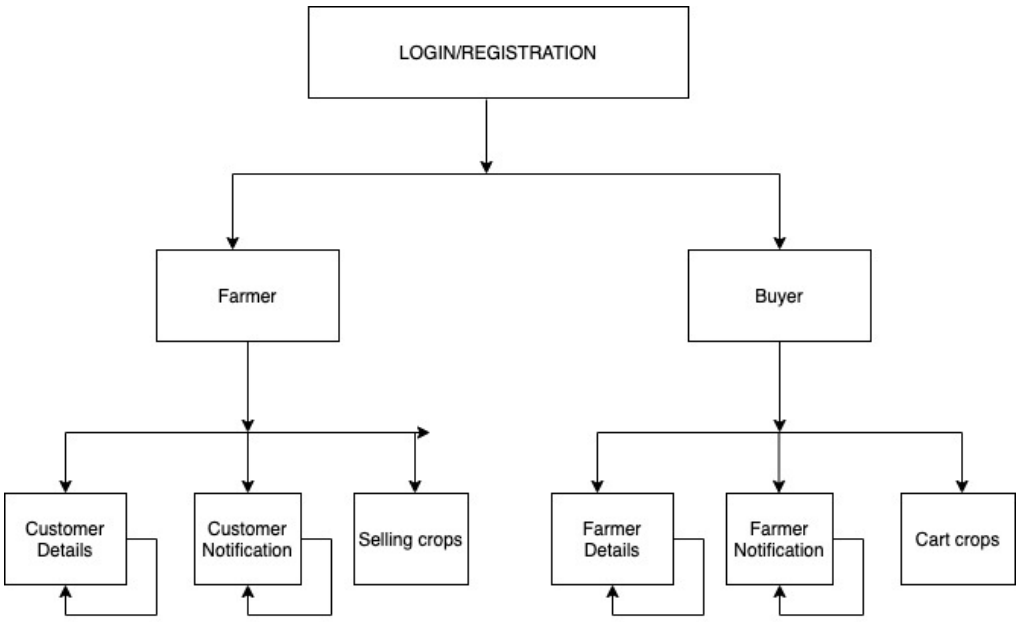
D. Android Application Agriculture Based.This paper talks about AgriCom is an android based application which is providing information to farmers regarding different crops and farming practices and other agricultural products. It is dynamic and interactive to take in the feedback and other input from the end users and can guide people regarding the different procedures that needs to be adopted .rThis project shows a simulation of live environment which takes different aspects of farming into consideration such as market demand and supply , production, forecast, fertiliser preferences etc.

E. Security in cloud by using Diffie Hellman Protocol. This paper focuses on how one can overcome the security concern faced by the user using Diffie Hellman protocol. It basically tries to eliminate the insecurity faced by the user for his data is on cloud server and under the control of the cloud service provider. A secure and trusted channel is provided with the help of Diffie

Hellman protocol such that no one except the user can access the data without his permission. Cloud computing has a very impactful scope. Multiple data scheme of encryption can be considered for maintaining security and liability of the data. Homomorphic encryption seems to be very effective but needs further study and consideration.

CHAPTER 2.1

Implementation and architecture diagram

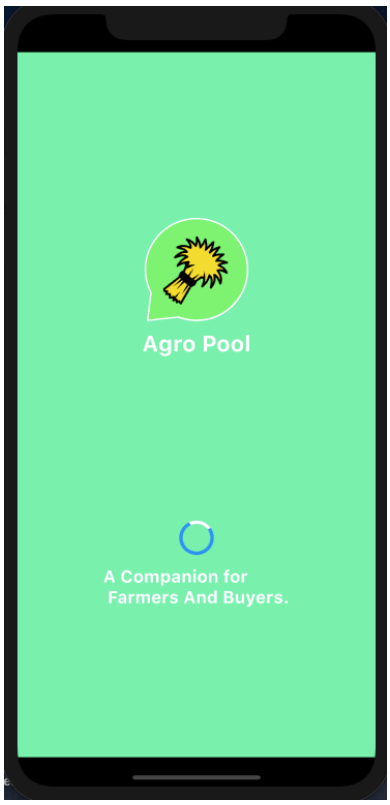


CHAPTER 2.2

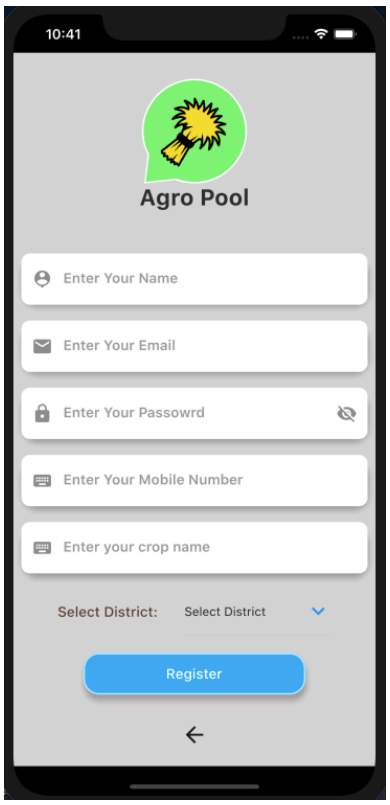
Output / Screen Shots

1. Hybrid App

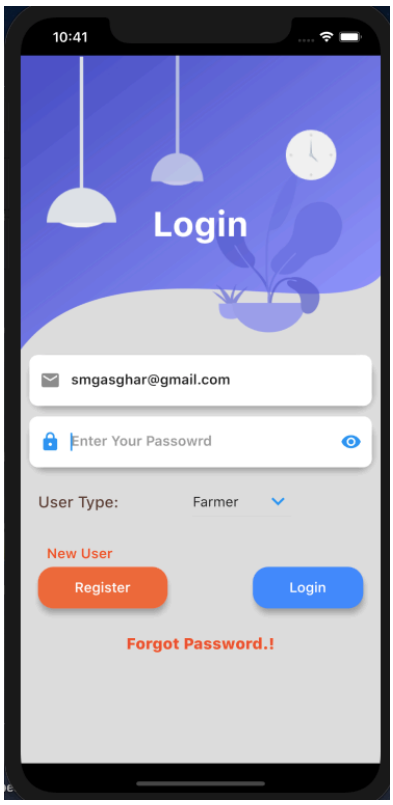
The Video Demo of the app can be found here



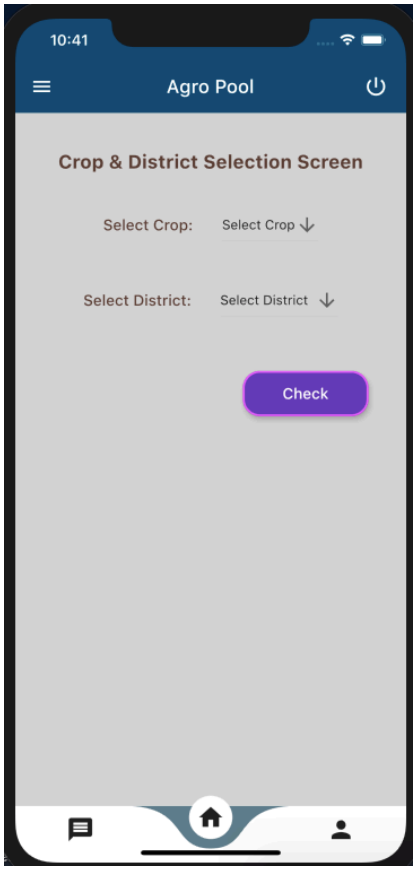
(Image: Splash Screen)



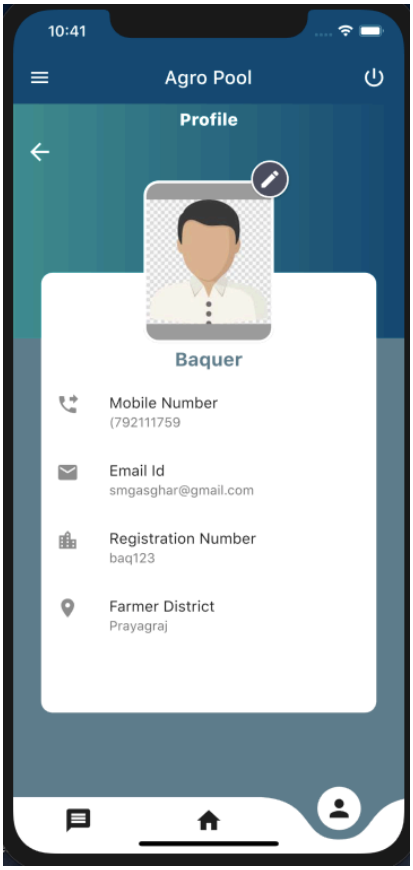
(Image: Registration)



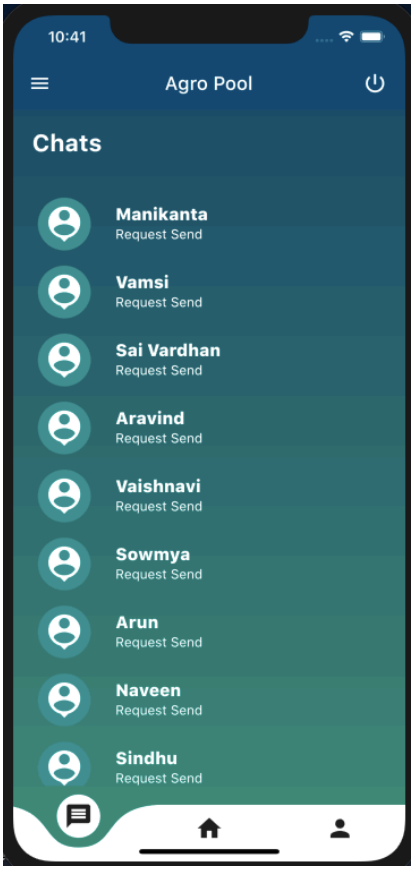
(Image: Login)



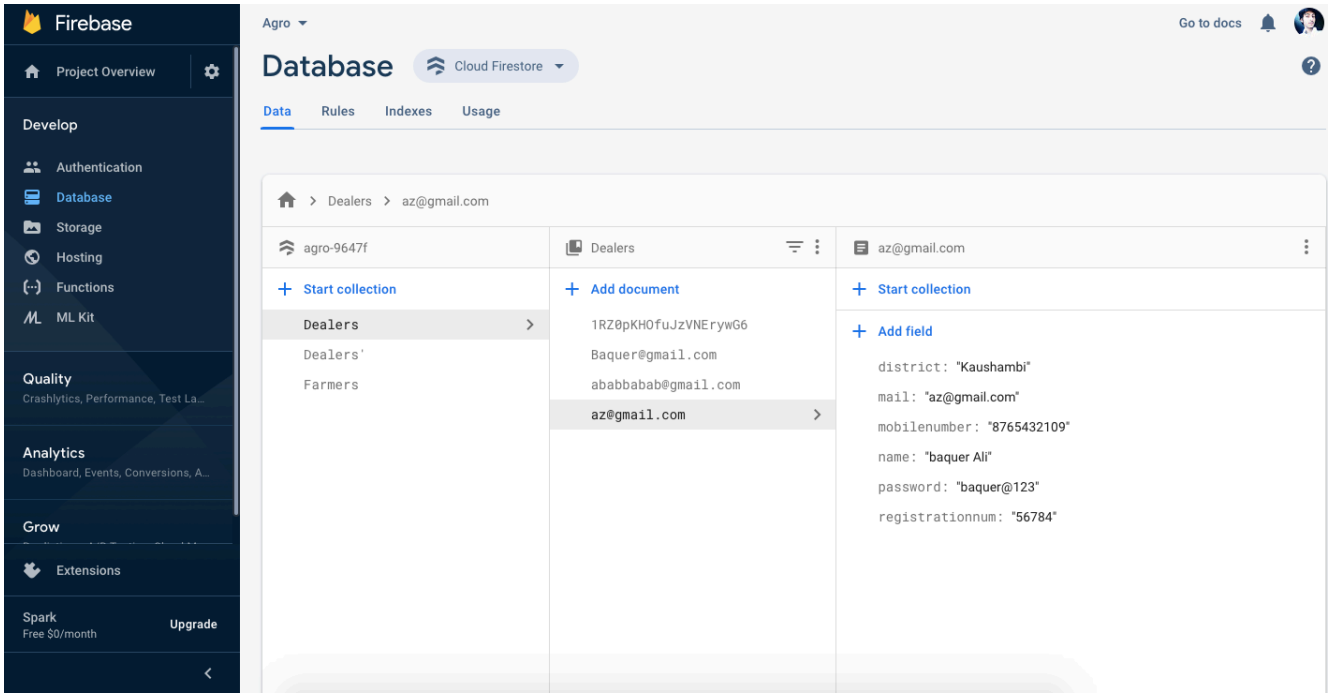
(Image:Home).

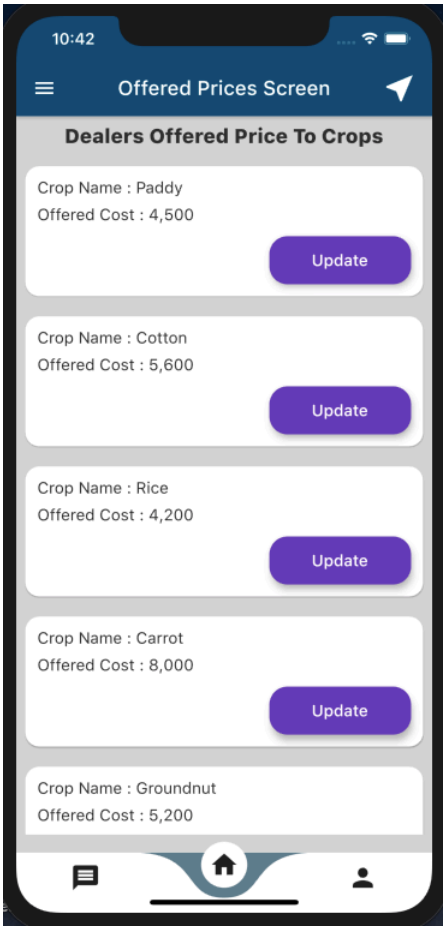


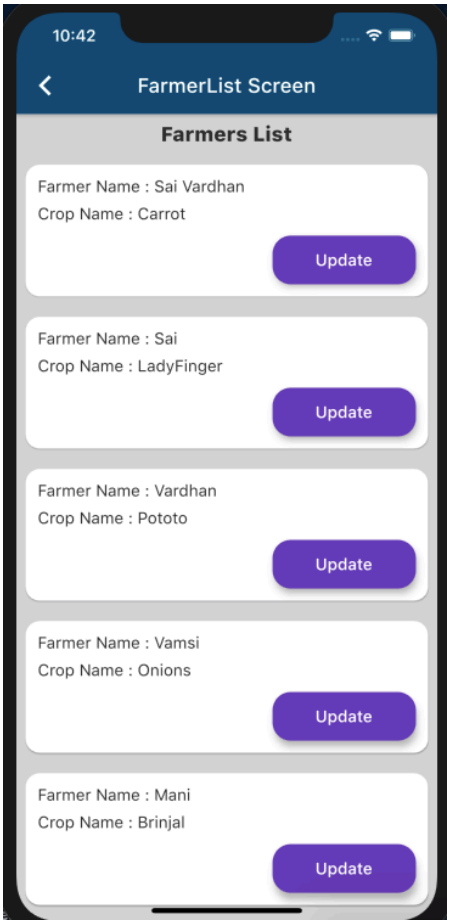
(Image: Profile)

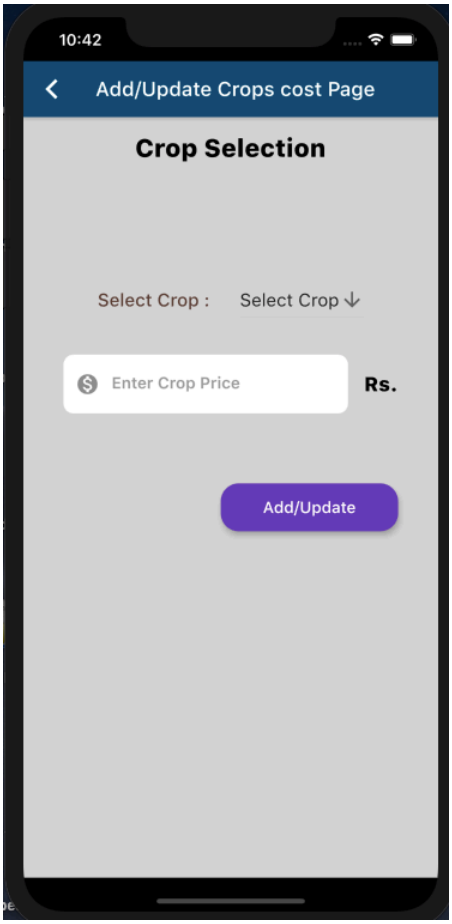


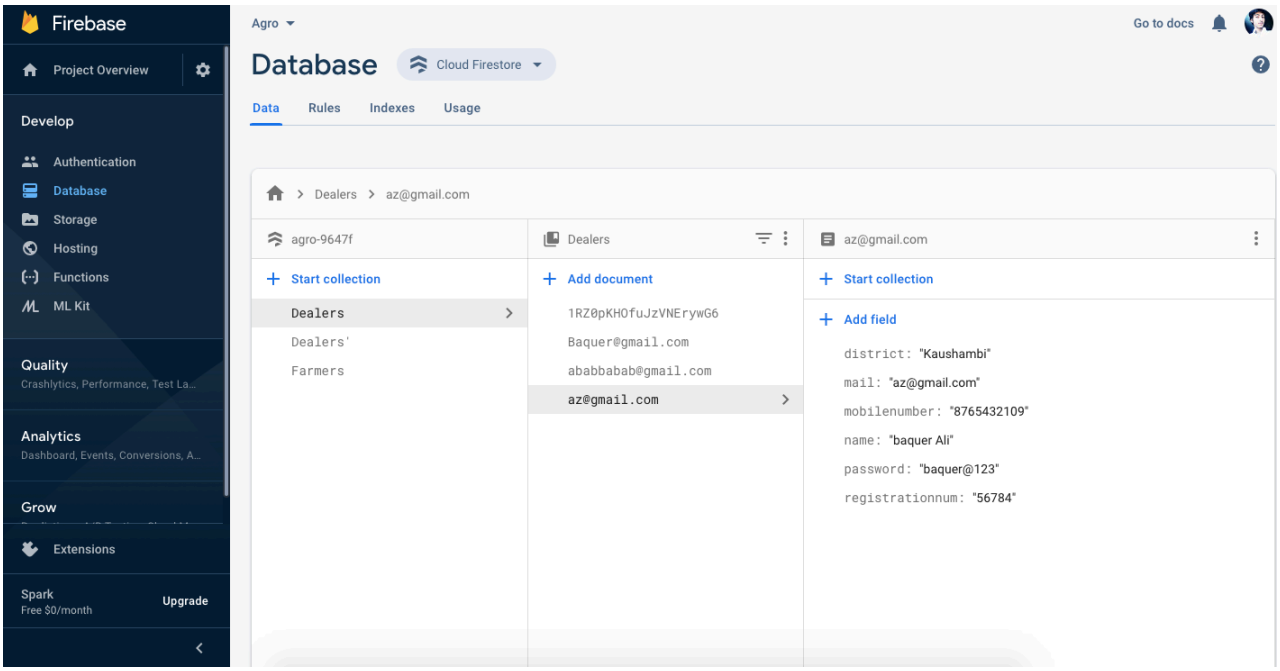
(Image: Chat)




10:42
Offered Prices Screen
Dealers Offered Price To Crops
Crop Name : Paddy
Offered Cost : 4,500
Update
Crop Name : Cotton
Offered Cost : 5,600
Update
Crop Name : Rice
Offered Cost : 4,200
Update
Crop Name : Carrot
Offered Cost : 8,000
Update
Crop Name : Groundnut
Offered Cost : 5,200


10:42
FarmerList Screen
Farmers List
Farmer Name : Sai Vardhan
Crop Name : Carrot
Update
Farmer Name : Sai
Crop Name : LadyFinger
Update
Farmer Name : Vardhan
Crop Name : Pototo
Update
Farmer Name : Vamsi
Crop Name : Onions
Update
Farmer Name : Mani
Crop Name : Brinjal
Update


10:42
Add/Update Crops cost Page
Crop Selection
Select Crop :
Select Crop
Enter Crop Price
Rs.
Add/Update


Firebase
Project Overview
Develop
Authentication
Database
Storage
Hosting
Functions
ML Kit
Quality
Crashlytics, Performance, Test La...
Analytics
Dashboard, Events, Conversions, A...
Grow
Extensions
Spark
Free $0/month
Upgrade
Agro
Go to docs
Database
Cloud Firestore
Data
Rules
Indexes
Usage
Dealers
az@gmail.com
agro-9647f
Start collection
Dealers
Dealers'
Farmers
Add document
1RZ0pKHOfuJzVNErywG6
Baquer@gmail.com
ababbabab@gmail.com
az@gmail.com
Start collection
Add field
district: "Kaushambi"
mail: "az@gmail.com"
mobilenumber: "8765432109"
name: "baquer Ali"
password: "baquer@123"
registrationnum: "56784"

CHAPTER 2.3

2. Diffie Hellman In AWS

The Video Demo of the app can be found here

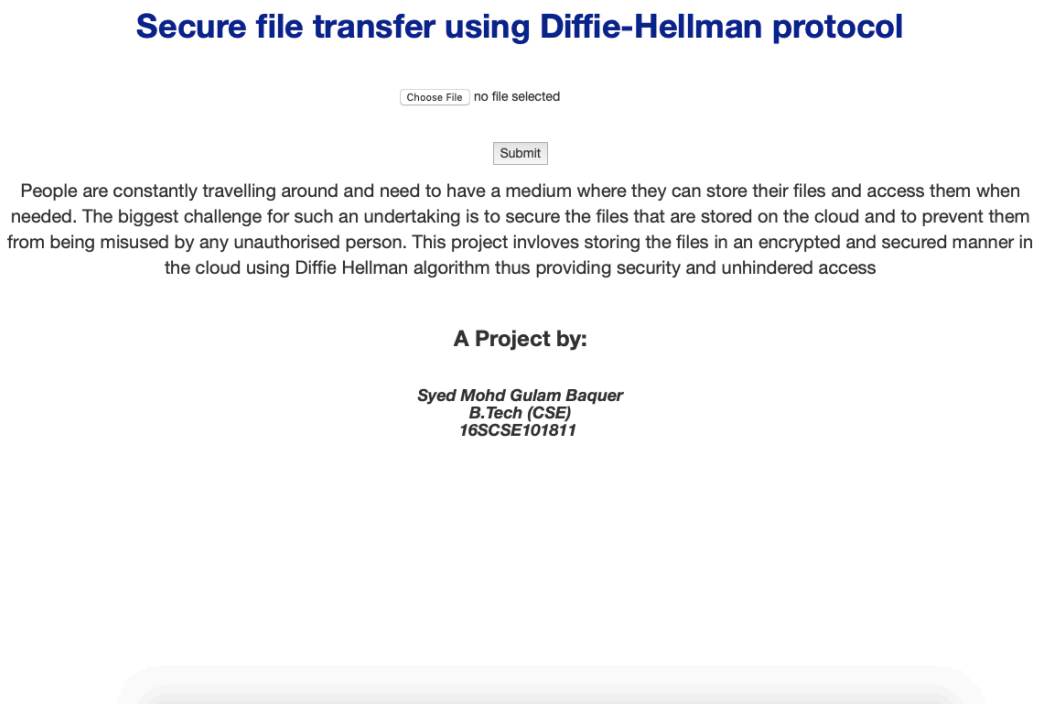
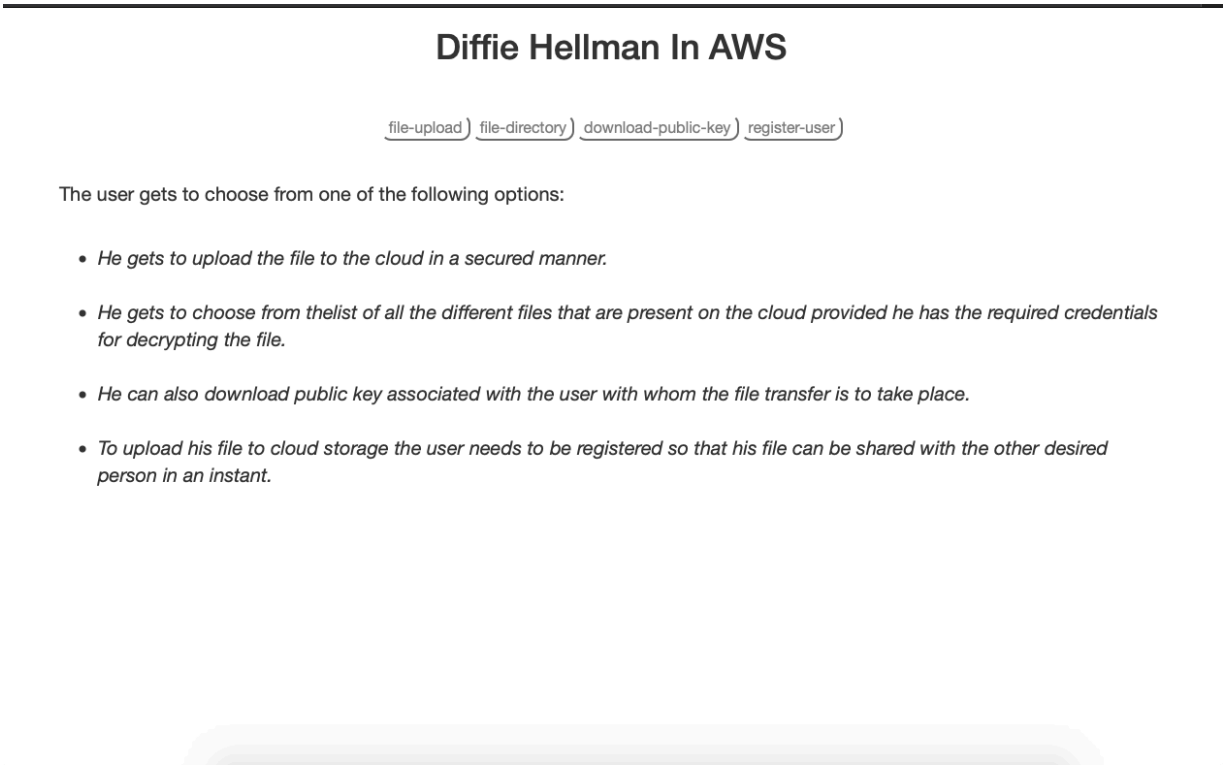


Image: Home Page



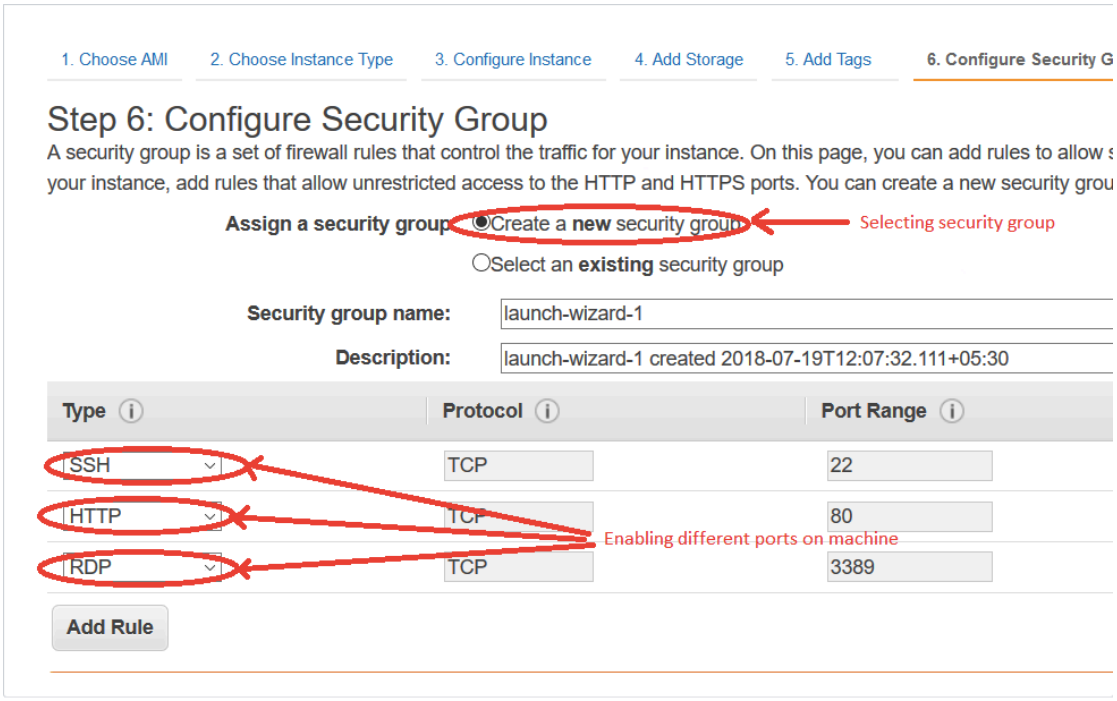


Image: Configure

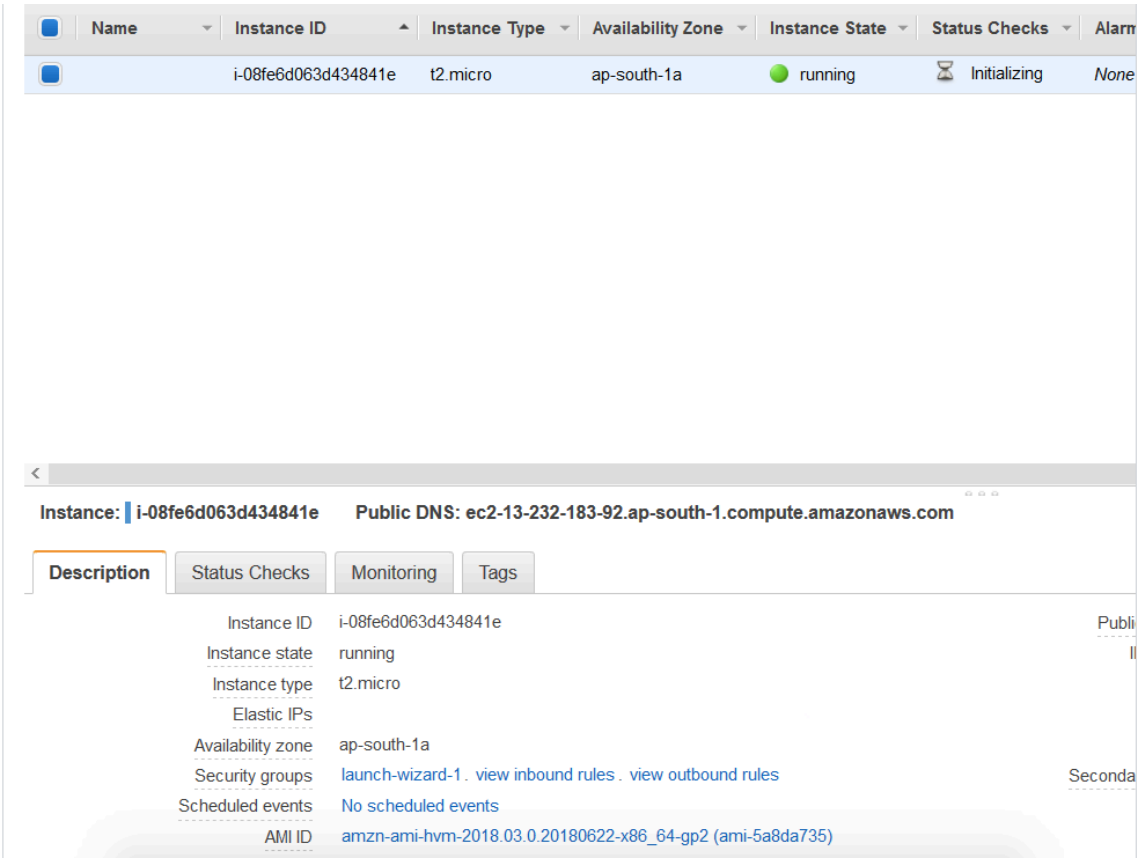


Image: Server Authentication

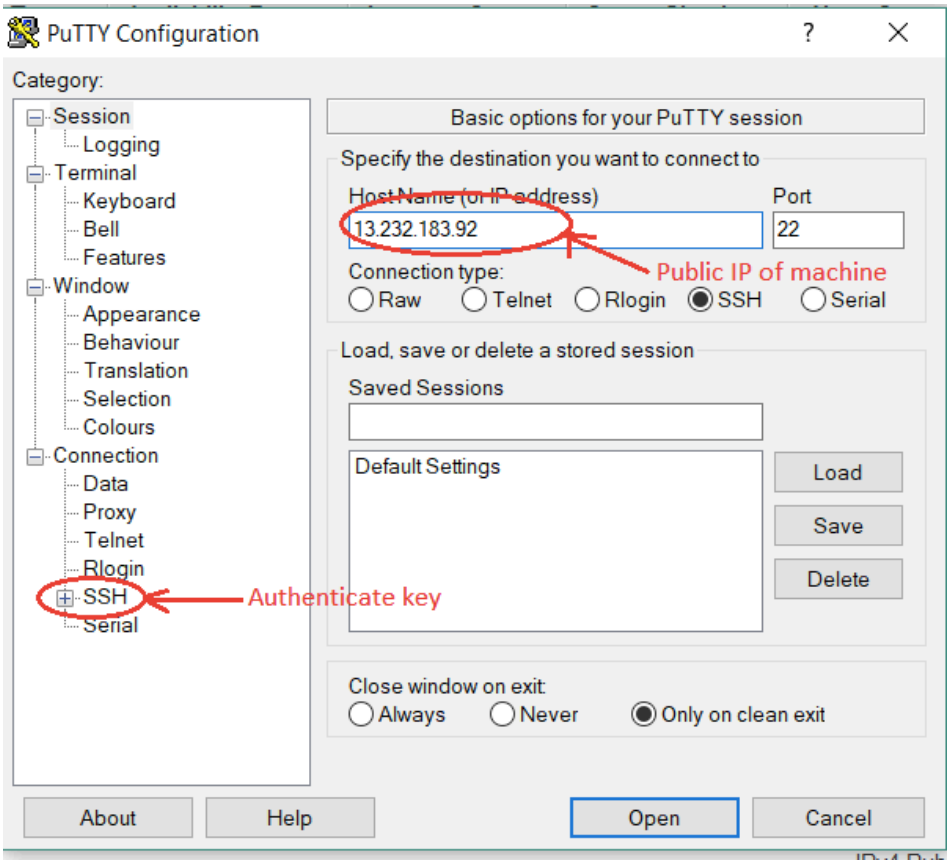
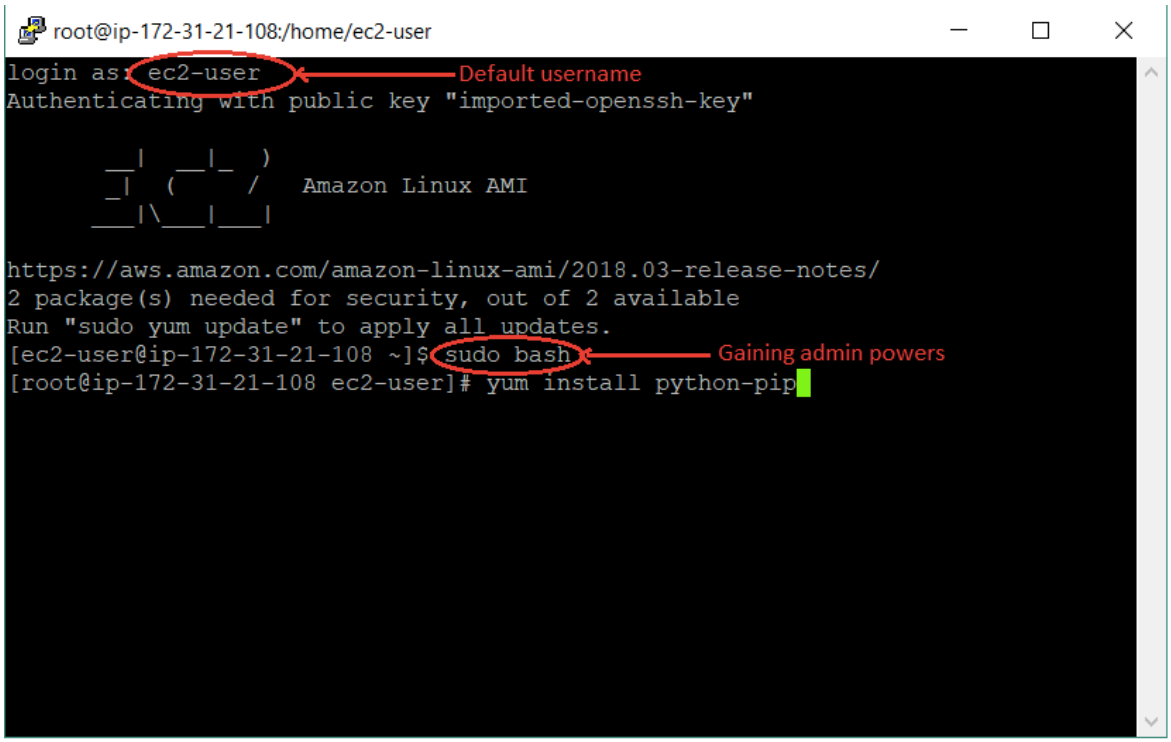


Image: Putty



(Image: terminal)

CHAPTER 2.4

Conclusion/ Future Enhancements

The farmers will derive greater benefit when they can make better decisions about where and whom to sell their crops after getting market prices for and also he get to know about the daily updates and news about the crops which will helps him in yielding large amount of healthy crops

And the second proposed model aims to address the problem of secure file storage on the cloud. This method is a basic implementation of the proposed methodology that can be improvised and customised according to the needs. It proposes uses Diffie Hellman to provide double layer of security to the files that are stored on the AWS Cloud.

In Future I will try to add the weather forecast and news about the crops in the app as due to Covid - 19 all the government API are not working and most of them are in no use, In future I will try to make this enhancements possible.

And in the second part I will try to add more extension file to upload as of now it is supporting only the text and png file so in future I will add other extensions as well.

CHAPTER 2.5

REFERENCES

1. Flutter Application Development - https://flutter.dev/
2. Krishi Ville-Android based solution for Indian agriculture. Authors-Manav Singhal Kshitij Verma, Anupam Shukla. ABV-Indian Inst. of Inf. Technol. & Manage., Gwalior, India. Advanced Networks and Telecommunication Systems (ANTS), 2011. IEEE fifth International Conference on Digital Object symbol ten. 1109/ANTS. 2011.636865. Publication Year:2011
3. N.K. Mishra 'FAO /AFMA/ Myanmar on improving Agriculture Marketing', Journal on Agricultural Marketing Information System. 2003, Vol 15, issue no 4, pp .no 2-4.
4. Yan Bo and Bu Yibi, 'Agricultural Marketing System in China', Journal on Agricultural Marketing Information System, 2003, vol 15, issue no 4, pp.no 33-37.
5. Brithal, P. S., Jha, A. K. and Singh, H. (), "Linking Farmers to promote for top worth Agricultural Commodities", Agricultural economic science analysis Review, 2007, Vol. 20, pp.no. 425-439.
6. Dhankar, G. H.,'Development of Internet Based Agricultural Marketing System in India' Agricultural Marketing, 2003, vol 4, pp no. 7-16.
7. Pathak N, "Contribution of Agriculture to the Development of Indian Economy", The Journal of Indian Management and strategy, 2009 vol 14, issue no 1, pp.no 52- 56.

8. Shakeel-Ul-Rehman, M. Selvaraj and M. Syed Ibrahim “Indian Agricultural Marketing- A Review”, Asian Journal of Agriculture and Rural Development, 2012 Vol. 2, No.1, pp.no. 69-75.

9. Xiaolan Fu and Shaheen Akter, ‘Impact of Mobile Telephone on the Quality and Speed of Agricultural Extension Services Delivery: Evidence from the Rural e-services Project in India’ International Conference on Agriculture social scientist, 2012, issue no 2, pp.no. 1-32

10. Saurabha A,Ghogare, Priyanka M Monga 2015 ‘E- Agriculture Introduction and Figuration of its Application’ International Journal of Advanced Research in Computer Science and Software Engineering,2006, vol 5, issue no 1, ppno.44–47. 2012.

11. Monika Jadhav, Vishakha Jagtap, Shankar M Patil ‘Android Application for farmer’ International Research Journal of Engineering and Technology, volume 6, Issue 04, pp.no. 56-72

12. abRuj S, NaDiffe Hellman A, Stomernovic I. DACC: distributed access management in clouds. 2011 International Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11, IEEE pc Society, 2011:91-98.

13. M. Klems, A. Lenk, J. Nimis, T. Sandholm and S. Tai. “What’s within the Cloud? Associate in nursing subject area Map of the Cloud Landscape.” IEEE Xplore, pp23-31, june.2009.

14. S. Kamara and K. Lauter. Scientific discipline cloud storage. In money Cryptography and information Security (FC’10),volume 6054 of LNCS, pages136strong KEY AGREEMENT supported PUBLIC KEY

AUTHENTICATION" Proceedings of the fourteenth International Conference on money Cryptography and information Security, Tene-rife,Spain,LNCS6052,pp. 383-390,Jan 2010.

15. M.Joshi, YS Moudgil "SECURE CLOUD STOARGE" International Journal of applied science 2011, ijcscn.com.
16. Yaoxue Zhang and Yuezhi Zhou_ "Transparent Computing: Spatio-Temporal Extension on von Neumann design for Cloud Services" TSINGHUA SCIENCE AND TECHNOLOGY, ISSN 1007-0214l l02/12l lpp10-21 Volume eighteen, Number 1, Feb 2013.
17. Linlin Wu and Rajkumar Buyya "Service Level Agreement (SLA) in Utility Computing Systems" Cloud Computing and Distributed Systems (CLOUDS) Laboratory Department of applied science and package Engineering The University of Melbourne, Australia.
18. Kawser Wazed Nafi, Tonny Shekha Kar, Sayed Anisul Hoque, Dr. M. M. A Hashem "A Newer User Authentication, File coding and Distributed Server based mostly Cloud Computing security architecture" (IJACSA) International Journal of Advanced applied science and Applications,Vol. 3, No. 10, 2012.
19. Abdul Muttalib Khan, Mohd. Haroon Khan, Dr. Shish Ahmad "Security in Cloud by Diffie Hellman Protocol" International Journal of Engineering and Innovative Technology (IJEIT) Volume 4, Issue 5, November 2014