FACULTY OF ENGINEERING AND TECHNOLOGY

A report
on

"SECURE WEB-DEVELOPMENT IN CLOUD PARADIGM"

Submitted in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING
(CLOUD TECHNOLOGY AND INFORMATION SECURITY)

Submitted by

Jatin Mehrotra.
16BT6CT006

Herish Chovatiya.
16BT6CT019

Mahir Gada.
16BT6CT020

Hrishikesh Kakkad.
16BT6CT021

Under the guidance of

Prof. Santhosh S.
Assistant Professor

Faculty of Engineering & Technology

Jain (Deemed-To-Be University)

Department Computer Science & Engineering

Jain Global Campus, Kanakapura Taluk - 562112

Ramanagara District, Karnataka, India

2019-2020

JAIN DEEMED-TO-BE UNIVERSITY | FACULTY OF ENGINEERING AND TECHNOLOGY

Department of Computer Science & Engineering

Jain Global campus, Kanakapura Taluk
Ramanagara District - 562112
Karnataka, India

CERTIFICATE

This is to certify that the project work titled **"SECURE WEB-DEVELOPMENT IN CLOUD PARADIGM"** is carried out by **Jatin Mehrotra. (16BT6CT006), Herish Chovatiya. (16BT6CT019), Mahir Gada. (16BT6CT020), Hrishikesh Kakkad (16BT6CT021),** a bonafide students of Bachelor of Technology at the Faculty of Engineering & Technology, Jain (Deemed-to-be University), Bangalore in partial fulfillment for the award of degree Bachelor of Technology in Computer Science & Engineering (Cloud Technology and Information Security), during the Academic year **2019-2020**.

Prof. Santhosh S Nair **Guide** Assistant Professor Dept. of CSE, Faculty of Engineering & Technology, Jain University Date:	**Dr. Devaraj Verma C** Head of the Department, Dept. of CSE, Faculty of Engineering & Technology, Jain University Date:	**Dr. Hariprasad S A** Director, Faculty of Engineering & Technology, Jain University Date:

Name of the Examiner	Signature of Examiner
1.	
2.	

DECLARATION

We, **Jatin Mehrotra. (16BT6CT006), Herrish Chovatiya. (16BT6CT019), Mahir Gada. (16BT6CT020), Hrishikesh Kakkad. (16BT6CT021),** are students of eighth semester B. Tech in **Computer Science & Engineering (Cloud Technology and Information Security)**, at Faculty of Engineering & Technology, **Jain (Deemed-To-Be University)**, hereby declare that the project work titled **"Secure Web-Development In Cloud Paradigm"** has been carried out by us and submitted in partial fulfilment for the award of degree in **Bachelor of Technology in Computer Science & Engineering (Cloud Technology and Information Security)** during the academic year **2019-2020**. Further, the matter presented in the project has not been submitted previously by anybody for the award of any degree or any diploma to any other University, to the best of our knowledge and faith.

Signature

Jatin Meherotra
USN : 16BT6CT006

Herish Chovatiya
USN : 16BT6CT019

Mahir Gada
USN : 16BT6CT020

Hrishikesh Kakkad
USN : 16BT6CT021

Place : Bangalore
Date :

ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of a large number of individuals who have been responsible for the successful completion of this project work.

First, we take this opportunity to express our sincere gratitude to ***Faculty of Engineering & Technology, Jain (Deemed-to-be University),*** *for providing us with a great opportunity to pursue our Bachelor's Degree in this institution.*

In particular we would like to thank ***Dr. Hariprasad S A****,* ***Director****,* ***Faculty of Engineering & Technology****,* ***Jain (Deemed-to-be University),*** *for his constant encouragement and expert advice.*

It is a matter of immense pleasure to express our sincere thanks to ***Dr. Kuldeep Sharma****,* ***Dean****,* ***School of Computer Science & Engineering****,* ***Jain (Deemed-to-be University),*** *for providing right academic guidance that made our task possible.*

It is a matter of immense pleasure to express our sincere thanks to ***Dr. Devaraj Verma C****,* ***Head of the department****,* ***Computer Science & Engineering****,* ***Jain (Deemed-to-be University),*** *for providing right academic guidance that made our task possible.*

We would like to thank our guide ***Santhosh S. Professor****,* ***Dept. of Computer Science & Engineering****,* ***Jain (Deemed-to-be University),*** *for sparing his valuable time to extend help in every step of our project work, which paved the way for smooth progress and fruitful culmination of the project.*

We would like to thank our Project Coordinator ***Prof. Adarsha S P*** *and all the staff members of Computer Science & Engineering for their support.*

We are also grateful to our family and friends who provided us with every requirement throughout the course.

We would like to thank one and all who directly or indirectly helped us in completing the Project work successfully.

Signature of Students

TABLE OF CONTENTS

Certificate	
Declaration	
Acknowledgement	
Table of Content	
Abstract	iv
List of Figures	v
List of Tables	v
Nomenclature used	vi
Chapter 1	01
1. INTRODUCTION	01
1.1 Overview	04
1.2 Problem Definition	05
1.3 Objectives	06
1.4 Methodology	07
1.5 Hardware and Software tools used	09
Chapter 2	11
2. LITERATURE SURVEY	11
2.1. Related Work	12
2.2. Existing System	13
2.3. Limitation of existing system	14
2.4. Proposed System	15
Chapter 3	16
3. SYSTEM DESIGN	16
3.1. ARCHITECTURE	
3.2. SEQUENCE DIAGRAM	
Chapter 4	20

4. TOOL DESCRIPTION 20

4.1.HARDWARE REQUIREMENTS 20

4.2.SOFTWARE REQUIREMENTS 21

Chapter 5 23

5. IMPLEMENTATION

Chapter 6 26

6. RESULTS 26-31

6.1 DISCUSSION: DIFFERENCES

CONCLUSIONS AND FUTURE SCOPE 32

REFERENCES 33

Appendix

ABSTRACT

The Cloud Industry has undergone many changes and updates since its inception and people are opting in more for the cloud services. After the development of 'cloud driven as-a-Service', people are extensively adopting cloud. This leaves the conventional and traditional methods of software development in a state of void. That said, some beginners and startups face difficulties adopting cloud since with recent upgrades the cloud infrastructure is becoming complex. The Internet provides multiple vague solutions for entirely undirected questions on the same. Developers are confused even more so.

To answer this need, 'Secure Web-Development in Cloud Paradigm' introduces a set of guidelines which facilitates development on the cloud. This improves app development productivity and at the same time minimizes the risk of project failure while trying to deliver a new safety aware SDLC in Cloud. The testing of this guide is done by developing an industry-standard web-app following the given steps. Discussed app is in the constant development phase as the discussion proceeds.

LIST OF FIGURES

Fig. No.	Description of the figure	Page No.
3.1	*Architecture and flow proposed by the guide*	16
3.2	*A general sequence flow used in the architecture*	19
5	*An abstracted view of implementation for the app development routine.*	8
6.1	*Application Screenshot- 1*	26
6.2	*Application Screenshot- 2*	26
6.3	*Application Screenshot- 3*	27
6.4	*Application Screenshot-4*	27
6.5	*Application Screenshot- 5*	28

LIST OF TABLES

Table No.	Description of the Table	Page No.
2	*A brief overview of Literature Survey on Cloud.*	11

NOMENCLATURE USED

AWS	Amazon Web Services
GSD	Global Software Development
AGSD	Agile based Global Software Development
GUI	Graphical User Interface
QoS	Quality of Service
SSDLC	Secured Software Development LifeCycle
GQM	Goal Oriented Method
MVP	Minimum Viable Product
aaS	as-a-Serivce
js	JavaScript

Chapter 1

1. Introduction

Shifts in tech norms tend to change people and their ability to use devices and technology. This leads a change in consumer-market brought by Industries leading that market. Trends and standards are thus set by the newly formed tech community. After a while, Industries launch updates and achieves maximum consumer productivity thereby creating a revolution. Imagine the number of shifts made just by introducing a new norm! One such is the space of cloud computing which has brought in major changes after its inception.

Gone are the times to rely on software vendors and distributors. Instead one should be leveraging the Cloud Providers, now that trusted parties like Amazon, Microsoft, Google, IBM are in the play, and start adapting to the change by being a part of it. The end users include consumers as well as developers and the IT team, if one thinks about it, since they are the ones who directly use the provider's service to create an end product or another form of service for market utilization. According to *Forbes*, as of 2016, a whopping 5.4 million developers are building cloud apps! And why not? The rapid increase in Cloud based ERPs (Enterprise Resource planning) and CRMs (Customer Relation Management) available even for SMEs (small-to-mid-sized enterprises, the cloud is no longer just a means to reduce IT costs, but a means to gain competitive advantage, interact directly with customers in real time, and transform. It is eminent for the end users to pull brakes to conventional and traditional methods of development.

Now, As-a-service, has bloomed quite exquisitely in today's time. The days where cloud horizons were limited to IaaS (Infrastructure-as-a-Service), PaaS (Platform-as-a-Service) and SaaS (Software-as-a-Service) are no more. The XaaS is here where 'X' represents 'Anything'. Analysts already forecasted that *the global XaaS market will grow at a CAGR of 38.22% between 2018 and 2020*. Today, with presence of vivid kinds of as-a-Service like Windows-as-a-Service, Database-as-a-Service, Storage-as-a-Service, Gaming-as-a-Service and even Malware-as-a-Service, the 'Cloud-driven as-a-Service' era is undoubtedly upon us. The True essence of Cloud driven as-a-Service is to opt for solution over utility. To simplify things further, users have a choice to pick between as-a-Service and DIY (Do It Yourself) in which case, they need to earn the machinery, the manpower and even the technical learning curve only then can they perform the task. Instead why not opt for as-a-Service and save money and time while doing the job?

For instance, if a user needs to quickly render model into a 3D film, why not just find a solution to use Blender as a service? The alternative would be to find an IaaS provider, launch instances, install Blender, and then figure out how to use it. This only makes sense if the user is an absolute control freak or likes to make life more difficult than it needs to be. For the rest of us, there is Software-as-a-Service! Rather than of wasting time "building", where once could be hitting up rendering jobs from user's handy mobile device and leverage the power of high-performance cloud infrastructure without having to care much about it. All user needs to know is that the video rendered in a fraction of the time it would take on powerful, tech-savy, expensive machine. And the user did the job from a lounge chair on the beach!

Re-invoking the true purpose of Cloud and as-a-Service Era into the minds of people has become essential now that XaaS has been growing over IaaS and PaaS. Remember the pay-per-Kilowatt and Pay-per-core-per-hour costing strategy used for IaaS? Although it seems absurd now, but people used to pay that when using IaaS over their proprietary software vendors since they found a solution instead of doing it all by themselves. SaaS and XaaS dominates IaaS in this idea even more so in an ideal scenario. By abstracting infrastructure and orchestration from end users, SaaS and XaaS focuses strictly on solving problems. The cloud computing remains the top catalyst for these services. Therefore, cloud computing proves to be a key component in growth of IT Sectors. Hence more companies across the spectrum have gained trust in cloud infrastructure services. "Cloud adoption and migration" for leveraging solutions using SaaS is a smart and recommended choice.

The above statements are true in all aspects of usage and trends of cloud, however, with increasing updates in such short time span, the technology is proving to dismay people from moving to cloud.

Cloud adoption and migration seems an easy term to comprehend with Cloud existing over so many years, conversely, the recent times have said that the years of Cloud existence is the exact reason why it has become a nightmare for people. The constant updating layers in the cloud causes shifts in virtualized infrastructure and architecture in a sense that it cannot be traced back. Problems arise when the developers have to deal with the issues related to structural complexity and troubleshooting or debugging in such tedious and inordinate environments.

The Ideal Cloud should be able to reduce cost, risk and complexity but as of now, people really tend to overspend on maintenance of the cloud and seemingly burn away huge cash causing monetary and market risks which ends up complicating the situation even more. *Global News Wire* announced the results of *NetEnrich* 2019 Cloud Adoption survey, completed by 100 IT decision-makers, on public cloud adoption in the enterprise. It clarified that growth of cloud services are likely to continue post 2019. A key takeaway from the survey was that IT pros remain concerned about a range of cloud issues, like interoperability, scalability and keeping up with changes. *Dimensional Research* says when things go wrong, more than half of IT professionals struggle to resolve those issues due to the varying complexity of their Cloud environment. This is popular amongst Businesses who have adopted Hybrid Cloud approach.

1.1 Overview

From applications to business processes anything can be delivered as a cloud service wherever and whenever user needs them. Cloud app development promises exponential benefits for small, medium and large organizations. While some analysts say it is still struggling to bloom, enterprises are under great pressure to take advantage of cloud services although most of them are fully unaware of the challenges they bring.

Migrating to the cloud offers tremendous benefits; but at the same time Migration is a time killer. Does one wish to spend their quality-time on migrating a local application to cloud?

With the current active user base of the application, the downtime to migrate data and components, unsurely efficiency and performance output and unknown encounters with debugging sessions, it is safe to conclude that cloud migration is a bit painful as it may lead to an eminent loss of customers when the shift is in process. A loss of customers is equivalent to losing the competitive advantage. Cloud development isn't useful all the time or automatically cost-efficient approach for all development requirements [1].

One can only think what immense pressure a newly started business would face, when choosing how to adopt cloud. "the promise of cloud computing is to deliver the functionality of existing information technology (IT) and enable new features, yet at the same time it should dramatically reduce the costs of IT" [2]. Some examples do point to cases where people have been overspending way too much cash on Cloud, other examples point to privacy concerns. Couple all this with the novice situation where one is totally unknown to 'the HOWs, WHYs and WHATs' about the cloud world. Certainly, at some time, like normal humans, they too call the internet and follow up online community for help. Although majority of them directly or indirectly point to the architecture and cloud environment that has been setup, none of them direct to a solution of building "cloud-ready" applications and environment. Rather the sight of vague questions and undirected articles which include unknown issues and terms like interoperability, scalability and refactoring and migrating to cloud, would blast anyone's right mind. All cloud environments are not intended for wider audiences [3].

Cloud Adoption is the key for exponential growth of Business. This is certain. But amidst all the chaos and unnecessary skim is there a standard solution for adopting cloud and building a cloud application? A definitive yet generic answer or guide to this is surely needed.

1.2 Problem Definition

This section discusses about a problem which we call “The Cloud Adoption Dilemma.”

There is no particular solution for this problem. None that provides standard, step by step implementation ensuring ease at developers’ or end users’ side. In fact, with the indistinct and obscure information available on the internet, which in itself is disorganized in nature, one falls more into the state of confusion as one tries to approach the answer. Hence this problem is called as “The Cloud Adoption Dilemma”. It includes those very repetitive set of questions the articles conclude to indirectly, while providing no answer and leaving the people with a sense of dismay. These questions range from Which Cloud Service Platform to pick? What tools to use for my development on Cloud? Which Cloud service would favor the development? Which phase does the product gets delivered? By what amount of time can a working prototype can be developed and delivered to the client? How to keep a client interested during the pre and post development phases? How to implement consumer demands in an ongoing development stage?

This demands a more friendly and simplified answer which guides the new comers and on-premise developers on how to leverage Cloud for meeting their application needs Cloud natively.

1.3 Objective(s)

Secure Web-Development in Cloud Paradigm aims to deliver a guide which answers to the demand of a general yet centralized idea or a suggestion which could address and possibly solve make unknown problems visible and known to developers, making the overall work flow easy for the entire management and functionality teams. So this project although addresses issues which could not only help developers but the entire organization.

This guide is crafted carefully to bring out the best of the 2 widely adopted standard development techniques which are Agile development and MVP development and combines both of them to smoothen the end users experience in making the shift to the cloud and make it less confusing.

This would present a generic and theorized remedy which on implementation in any cloud adoption scenario highlights and identifies development related problems while solving them in a step-by-step procedural manner.

Why is the guide used as one of the two end deliverables of the project? Because end users stand in need of best practices engineered for them, not infinite variations and options to (hopefully) eventually arrive at the same best practice experts would.

1.4 Methodology

This guide proposes a step-by-step guide with a model case following which provides a solution for issues related to cloud computing and cloud adoption. Here is an overview of problems faced and how its model provides solution to those problems.

- First and foremost, there is no standard guidelines set for generic cloud development situation. This guide simplifies it by breaking down the major steps and combining issue prone steps with it. The model is a step by step process which eases the pain of self-searching unknown troubleshooting issues. These issues typically arise due to improper deployment or management of cloud, which in discussed case won't be occurring because of the procedural nature of the model.
- There is always a growing confusion on how to select cloud platform and what necessary elements to consider. The guide's presented model, has included a specific cloud evaluation process as one of its initial steps which provides cases on types of service to use in what given condition. For instance, consider user needs to create a certain type-A app then so and so service would be useful where Type-B requires X cloud platform with listed so and so services can be used. Considering that there are multiple roundabouts to a given problem this guide tries to deliver its ideals with a "can" instead of should.
- This work sides to the preference of working over one single cloud platform instead of multiple cloud platforms i.e. Hybrid cloud approach. This is due to 2 main reasons. First being that it promotes cloud adoption in order to induce a sense that people view it as a solution and not a utility, thereby not allowing the risk of overspending cost on maintenance. This is because having taken the considerations of "cost" and "audience-type" per say startups or meagre group of developers, in picture. However, if the user has enough cash-inflow one should be considering Hybrid cloud approach for scalability. Secondly, since this work emphasize on ease of approach, it tends perform on a single cloud platform. Because through this, maximization of loose coupling is achieved. In a customer-centric world where needs keep changing it is essential for the cloud native applications to not be tightly coupled to the underlying service logic. Loose coupling ensure independency of cloud native applications, by not only offering required flexibility but also more reusability when components are added, replaced or modified.

Developers using different vendor services have to understand the different protocols used and take responsibility for orchestrated application behavior and tendencies.

- Cloud native applications are more prone to risks as per the norm. So, to keep everything in check, the user is recommended to make sure that the data is encrypted. Alternatively, one can also add more security-based cloud services which specifically stronghold the firewall. A reverse proxy can be used to address security concerns, or make use of SSL. The project tries and recommend this exact security standard to its users to meet the enterprise security concerns.
- Being on cloud one should fully take advantage of the global availability by making sure that global development is utilized. To prepare this guide, various models like waterfall, spiral, V model etc. were studied and some flaws were highlighted that have been traditionally passed down by those models. Next, this guide makes sure Agile principles are implemented and established to enable GSD since it is solely based on cloud. This enabled the members working on the project to track and manage issues and sprints in real time and become more aware of Agile Global Software Development (AGSD) during the phase of 3 months lockdown due to the recent COVID-19 widespread. Even though none of them could physically meet, Agile Scrum principles allowed them to stay in sync and track progress with each element that were distributed amongst the development team members. So with everything being stated this guide and the project supports agile principles combined with MVP deployment model and is completely in sync with it.

1.5 Hardware and Software used

Since the project in itself is whole and sole based on use of cloud computing platforms, it is an acknowledgement as well as a sense of stating the obvious, that the hardware requirements are bare minimum.

Have a look at the Hardware requirements for the project:

- The Hardware is required to test and run things locally before committing changes to the main team branch on cloud.
- All though not compulsion but each developer needs to own a Laptop/PC where he/she can work on.
- A bare minimum configuration of 2 GB RAM with 10+ GB of free disk space would be more than sufficient.
- Operating Systems are user preference. Whichever OS the user is comfortable with can be used.
- The Laptop/PC configurations aren't much stressed. Any configuration machine whether old or now heavy or lite doesn't matter. One needs it run code and open websites that's all.
- A good internet connection is required. A low latency connection on development machine is always welcomed and appreciated.

However, since the project deals with cloud development and web applications, the software requirements seem to be more.

Have a look at the Software requirements for the project.

- As mentioned previously, no specific OS is demanded compulsorily but one needs to know the complete use of terminal. Hence command-line terminal or console should be installed and one should know how to use it.
- GitHub Software and local Git repository should be present in the machines.
- Knowledge on application development.
- In depth knowledge on AWS/Azure/GCP services isn't required but would definitely be a plus.
- Knowledge on developing applications.
- Balsamiq Wire Framing

- Text/Code Editor: Atom or Visual Studio Code
- Web Browser with scripting enabled.
- Jira Cloud installed.

That being said, since the end product will also be an application, one should have enough knowledge about developing it. Here's the learning curve:

- Front-End Design languages
- Back-end Server Coding language
- Interaction between server-client
- Database query handling
- Command-Line
- Version Control
- Jira Scrum (Issue control and Sprints)
- CI/CD
- Code Pipeline
- Lambda and Server less
- Deployment testing

Chapter 2

2. Literature Survey

For developing the guide, a thorough and in-depth research and analysis of work has been done. This work covers about 8 years of development and updates that has been seen in the cloud domain. Right from its inception. To brief about each year's development here are one liner on the topics that are covered in each year.

Year	**Cloud concepts which were researched and published in that year.**
2011	• Global Software Development (GSD) came into existence.
2012	• Awareness about Software as a Service spreading as opposed to vendors. Highlights new challenges and needs that are to be tackled.
2013	• Concentration on Quality of Service (QoS) and various aspects that contribute to QoS. • Energy efficient cloud provisions were discussed. • General security implementation and standard establishment. • Goal Oriented Method (GQM) as a process measurement tool. • Various approaches to cloud evaluation.
2014	• First time established research about SSDLC Secured Software Development Life Cycle.
2015	• Green computation and energy efficient cloud adoption for low carbon footprint. • Need for more cloud monitoring systems.
2016	• Problems in GSD. How Agile based GSD (AGSD) can solve the problem. A proof of concept on how agile can be worked with cloud migration and cloud adoption.
2017	• Micro services and Serverless were introduced. Containers for cloud came into wide existence.
2018	• Security testing and tools used for secured testing. • Growth of XaaS (Anything as a Service) as a business venture in cloud domain.

Table 2:

2.1 Related Work

A revised 2011 Research highlights the different problems defined to implementation of Global Software Development. [4] These problems are mainly Geographic (Distance, Time, Knowledge transfer tools), Cultural (Unequal distribution of work, lack of trust, Fear, temporal and linguistic). They highlighted that by using the cloud to support GSD products, problems like Geographic issues and ownership issues can be resolved by 'Service Ownership'[4] which allows service users to focus on their core activities.

The very next research on SaaS helped us gain the purpose of how SaaS becomes a better deliverable over other business in the domain. [5] It presents some great insights on SaaS. Answers on why SaaS is a better deal as compared to traditional software package (cost, model, security, and other benefits) can be found in there. Overall, it has a great deal of information providing surface knowledge on business benefits of SaaS.

On the same year of 2012 another groundbreaking published research surfaced. The reason why it was ground breaking was that this publication [6] directly presents the challenges and needs in the architecture and business model of Cloud in general. This highlighted the area which needed modernization and also tells about problems adopting SaaS [6] leaving various legacy system problems which may tend to cause hassle and are hard to solve.

2.2 Existing Work

There was a relatable existing work on one of the features which the model directly or indirectly provides to or rather highlight it to the viewers. Quality of Service (referred to as QoS henceforth), a proprietary researched topic on 2013 emphasized its concentration on QoS and various aspects which contribute to QoS.

Majorly commendable work rises on 2017. One of the most prolific frameworks which is theorized came into existence named Unicorn Application Framework (henceforth referred to as UAF), although this was based for the purpose of IoT, it induced lots of free-floating ideas that can be implied to Cloud suggesting some room for further enhancement. The major principles of UAF were as follows: 1) support for cloud deployment, 2) mobile-first - the focus is on supporting mobile application, and 3) IoT-readiness – the architecture enables access for traditional human users as well as IoT devices.

The next pivot point is the Li Brien et al [7] evaluating Cloud's commercially available services based on Performance, Security and Economics. GQM (Goal Oriented Measurement) paradigm , a process improvement method in which measurements are laid down to address completeness, consistency and suitability of a process, conducted by Rakesh et al [8] clearly states that Li Brien's catalogue seems to match and meet the goals of GQM i.e. CPU, communication, memory (for processing), scaling latency, benchmark cost etc.

Apart from this a research also provided Comprehensive detailing about SSDLC for Web Application Development. [9] It assisted in how different team members (referred as Actors) can contribute to different phases (feasibility, planning and development) of the same cycle of SDLC with security in place. A new SDLC model was hinted and coined as SSDLC or 'Secured Software Development Life Cycle'. [9] SSDLC has been broken into modules and a robust monitoring system is suggested to be kept in place to ensure that a security output of a module in SSDLC is efficient enough to integrate and work well with the next module in the cycle and also monitor system-level security. A suggestion is provided on how to address issues related to security on the existing traditional SDLC model. They address all the key phases of SDLC to ensure security measures by initiating any kind of development activities.

2.3 Problems in Existing system

Here are the list of problems, out of which one or more cases highlight the issues in some of the previously mentioned existing and related work:

- The set of research, stands and supports only 1 specific situation or goal thereby is not addressing the generic solution but the one which falls specifically to a particular issue.
- On that basis, some of the work need to mention more vivid scenarios for application.
- The way the testing environment should be developed is still remains unmentioned and is thereby an area of further research.
- The environmental settings are very cost-bulky and complicates the environment settings.
- Excessive modifications are needed to assure proper follow through of the framework model. Reusing standard framework services across the project saves programming effort and improves reliability of the application.
- Moving from Global Software Development to Cloud Native development brings in lots of challenges while solving some of them requires a complex learning curve.
- Based on what factor would one decide when to move on next phase? It really does seem blurred out in the specifics.
- While guaranteeing the outcome, it does not talk about QTC Quality, Time and Control.
- One needs a fixed architecture to test the vulnerability flaws, whereas ideal life Cloud seems to be tight coupled with Services and Logic Implementation.

2.4 Proposed System

This answers to the demand of a general yet centralized idea or a suggestion which could address and possibly solve multiple issues that have been identified over the time.

Secure Web-Development in Cloud Paradigm presents a generic and theorized remedy in form of a *guide*, which on implementation in any cloud adoption scenario highlights and identifies development related problems while solving them in a step-by-step procedural manner.

As stated, it provides guidance in following aspects,

- This model states and covers the recommended actions and steps to be implemented while following a standard procedure.
- It is a representation of standard development process using Agile Development, with a major addition of doing everything “Cloud Natively”.
- It provides an immersive view of the whole process of Cloud Native app development.
- Follows developer friendly model combination of Agile and MVP (Minimum Viable Product) Development techniques to bring the best out of the two.
- Highlights security aspects and implementation phases as and when required.
- Massive help on initial stages during Cloud provider and platform evaluation.

Chapter 3

3. System Design

3.1 Architecture

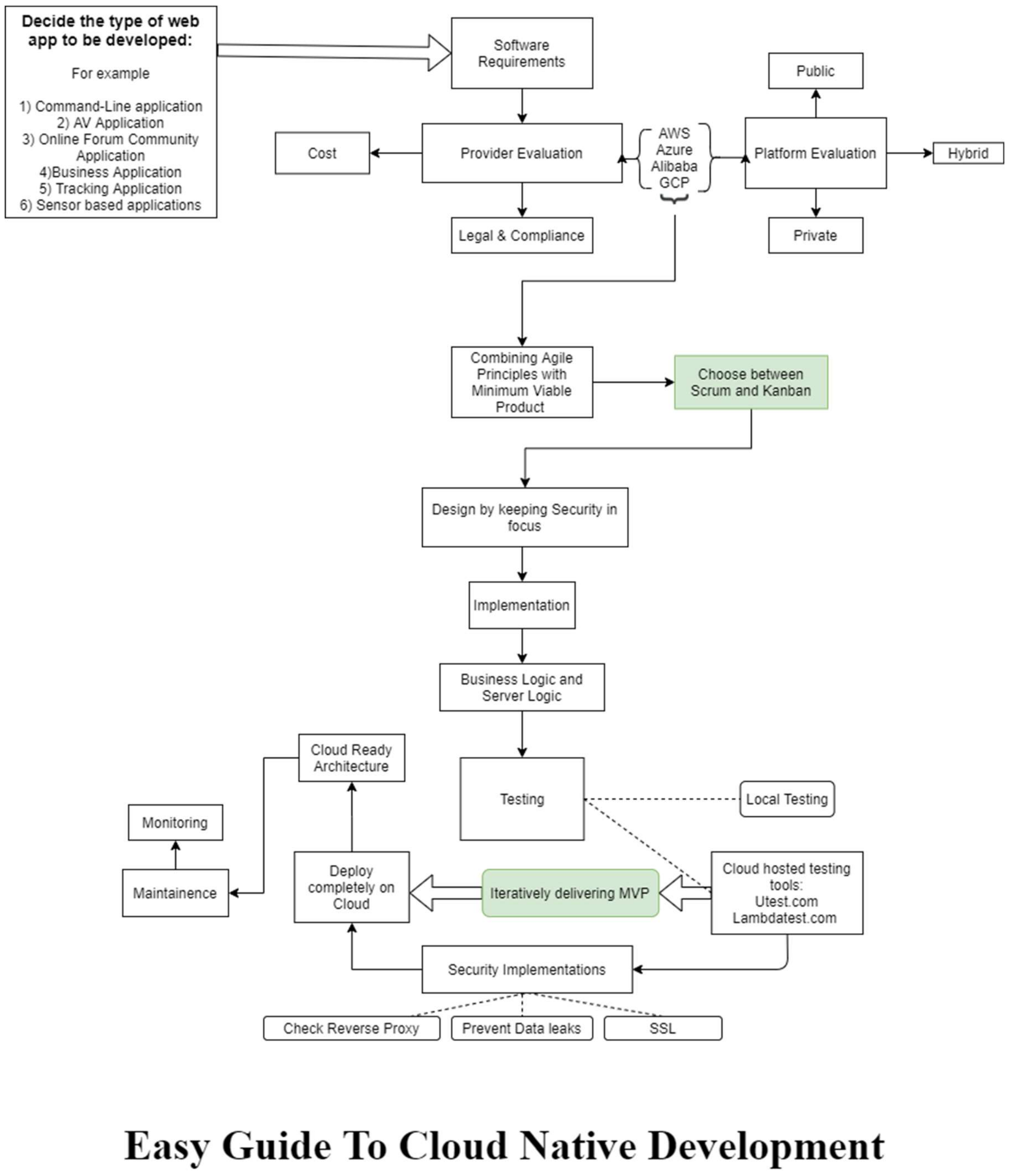

Decide the type of web app to be developed:
For example
1) Command-Line application
2) AV Application
3) Online Forum Community Application
4)Business Application
5) Tracking Application
6) Sensor based applications
Software Requirements
Provider Evaluation
Cost
Legal & Compliance
AWS
Azure
Alibaba
GCP
Platform Evaluation
Public
Hybrid
Private
Combining Agile Principles with Minimum Viable Product
Choose between Scrum and Kanban
Design by keeping Security in focus
Implementation
Business Logic and Server Logic
Testing
Local Testing
Cloud hosted testing tools:
Utest.com
Lambdatest.com
Iteratively delivering MVP
Deploy completely on Cloud
Cloud Ready Architecture
Maintainence
Monitoring
Security Implementations
Check Reverse Proxy
Prevent Data leaks
SSL
Easy Guide To Cloud Native Development

Figure 3.1 Architecture and flow proposed by the guide

As per the figure 3.1 the guide recommends the following actions to be performed which leads to hassle free and easy Cloud Native Development.

1. It all starts with the decision on “what type of application needs to be developed based on business needs”. This is essential and acts as a base for deciding what do Agile users expect as an outcome and what kind of technology needs to be included to achieve it. It also helps in further evaluation of cloud services that can be used for achieving business logic.
2. Requirements for software are laid, which encompasses all types of requirements such as mission, goal, scalability, audience, feature set, budget, etc This helps in gathering information about customers and market and how they react with each other creating demand for the application. This is a part of MVP development scheme that is used in combination with agile in this guide.
3. Then a thorough analysis is done on what kind of cloud provider will be best suited and the decision for a provider is mainly based on cost, legal, and compliance.
4. Once the provider is decided, its platform is evaluated, and its services which can help us to achieve the solution. The main agenda is to look at what kind of web application is being developed and which kind of target audience would be its market. This can bring clarity to the services and costs which needs to be put into action to achieve the development.
5. The guide recommends to undergo AGSD methodology combining the power of cloud and its services. Agile development methodology is much more beneficial as compared to other traditional software development methodologies
6. As a part of Agile development, the team gets to choose between Scrum framework and Kanban framework. The guide doesn’t specify the user to pick a specific one as depending on the type of application and adaptation of the developers, any of the two can be picked and followed to manage modules.
7. MVP or Minimum Viable Product is the correct balance of Minimum and Viability factor which ensures proper interests of end users and also takes in the minimum amount of efforts that needs to be put in the development. Efforts here can be a replacement of cost, scalability, mobility and sometimes even features. A product of MVP development method is a subset of Minimum and Viable.

8. The mixture of Agile values and MVP development gives a much better, clear and clean solution as it:
    - Ensures delivery of working software in weeks rather than months.
    - Instead of customer satisfaction, customer interest is prioritized.
    - Sustainable development with constantly evolving pace depending on evaluating customer wants and delivering a MVP in process.
    - Simplicity in workflow ensures maximizing the amount of work undone.
    - Welcoming changing requirements
    - Regularly the team reflects how to cash in customers' interest and become more effective.
9. Information Security is given the high regards, which is why it is included in the designing phase right from the start, making the developers aware that they need to implement and build a secured application right from the scratch rather than adding security to the application after it is built. This sorts out unwanted troubleshooting and structural problems.
10. In the implementation phase, implementation is laid where business logic, backend, front end is implemented using suitable technologies.
11. Every version produced thorough the development proceedings is tested both locally and on the cloud in a pseudo production environment keeping all the edge cases into consideration.
12. Once basic functionality is achieved and tested, security is implemented right after such as user validation, database security, through checking for owasp top 10, along with segregation of public resources in public availability zones and mission-critical resources and serves in private VPN.
13. In the final stage, it deployed in the cloud, since it is a cloud-ready architecture there are no major lifts and shifts during deployment which leads to final stage of maintenance.
14. In the final stage of maintenance, logs are thoroughly monitored, there is a watchdog for error and continuous improvement takes places based on the changing business needs. This guide truly promotes team growth over process and tools and always is adaptable to change, the model can endure change and keep up with it. Additional layers of updates cannot complicate the architectural structure of the model. It shall secure and harden the end point.

3.2 Sequence Diagram

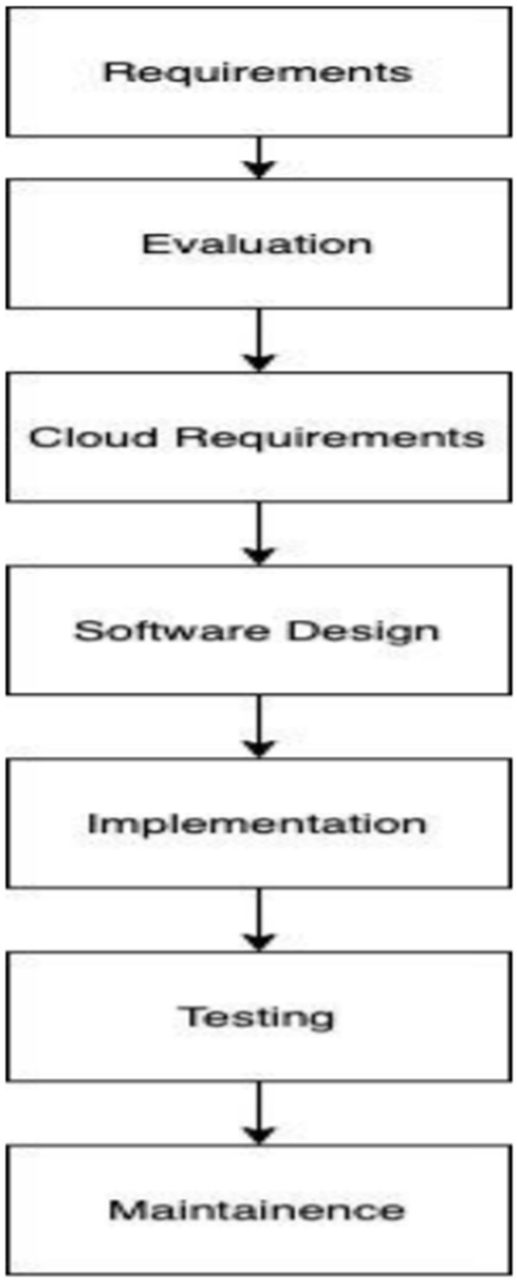


Figure 3.2: a general sequence flow used in the architecture

Chapter 4

4. Tool Description

4.1 Hardware Requirements: Description

There is no stress on hardware requirements at all. It will be a decade now since the introduction of cloud computing and as it has modernized now, this guide is comfortable to tell that cloud users need not to worry about their infrastructure as far as Hardware is concerned. This is because cloud host platforms provide user with all the required hardware. A GUI access is given to the user to run and command the assigned hardware through user's web client (browser). This makes it hassle free and for sure saves the cost of owning and cost of maintenance. All-in-all this is why the guide does not emphasize about the hardware requirements to the users being a step to adopt cloud native development and routine.

Note: Everything above being said, user laptop/computer/machine should be capable of supporting heavy browser activity and local code deployment. So a minimum bar of 2 GB System RAM and about 10+ GB of disk space (apart from the local and Operating System storage space) is set for the purpose of code/data storage and caching. This configuration is available in every machine at this point (even machines ranging five years of obscurity from current date match the given configuration).

4.2 Software Requirements: description

This guide do not recommend any specific OS on user's system, hence users are free to use any Operating System which they are comfortable with.

What the guide ask users, is a good amount of knowledge on local application development and the process involved in it. This includes a thorough usage of terminal or command-line as well. Considering majority of the target audience shall be developers or cloud enthusiasts, this applies to the norm as a default.

Starting with actual description of required software the first and foremost is knowing about Git & installing GitHub. Hands down, version control guarantees flexibility to a team of developers, allowing them to store a repository to master branch which serves as the main head branch of code line, while giving a feature to test and grow user's code in a pseudo repository which is stored locally in user machine. This typically allows a team to test the updates and debug errors in local code. If every module of the code works out correctly without any errors, the local repository can commit to the main branch repository stored on GitHub.com. This is helpful in all development cases especially ones which have a team of 2+ developers with task to develop allotted modules and combine the distributed modules into a single package for the whole application to work.

Secondly, choose and install libraries required for the language(s) one prefers to develop the application with.

Since being a generalized model, the project is sticking with the standard HTML, CSS and JavaScript format for its choice of language usage. Hence required libraries need to be setup in the environment. Compiling every language and working with it together will be tough, hence developers are recommended to install a text/code editor, can be as lite as Atom or can be as heavy as Visual Studio Code. But having an IDE or Text/Code editor is absolutely essential. Alternatively, one can also use an Online IDE hosted on the cloud, Cloud9 IDE for instance, is an online IDE with all the required libraries readily installed for user's online usage. Another such example is codeanywhere.com which essentially is a cross platform Cloud IDE, giving an additional power to collaborate and run user code using any device.

Every interactive web application needs a backend server to host it. That's where Node.js comes into picture. It is JavaScript runtime environment which can execute JavaScript code outside of a web client/browser. It is generally used to create scalable web applications. Node allows developers to deploy JavaScript code as a machine process instead of running in browser. This is why, Node can be used to create server-side applications.

Agile is a methodology which ensures enhanced and procedural way for development. To comply with agile principles inclusion of Jira Cloud by *Atlassian* is done. This allows to imbibe agile principles by implementing Agile Scrum framework and/or Agile Kanban framework for developer team. Opting and implementing one of the two frameworks depends on the kind of process user wants to take to build the application. The main goal of Agile is to break the project into small iterative parts the Kanban framework focuses on process improvements whereas Scrum framework is concerned with amount of work delivered in shorter time frames. This is achieved by creating an issue pool and adding tasks to the issue. This issue is then worked on sprint basis. Sprints are short time intervals after which one needs to showcase his or her ongoing task deliverables. That's how one issue gets resolved. This functionality along with multiple big timeline, project environment task, and multiple small module task can be managed and viewed using Jira. Jira makes the project Agile-ready and thereby enables us to create, manage and monitor tasks and distribute the task and issues among developers by implementing sprints and accepting deliverables recursively.

Balsamiq Wire Frame is used to wire framing or in simpler terms designing an application before actually designing it. One creates a blueprint of how and what the application is going to look like and after creating a wire frame one can start working on the actual project of building the application.

Apart from these, one needs to have a knowledge on AWS and Azure Services. This will allow a user to gain a diversified option all the while have a comparative analysis on both of them. An account for developer access to AWS and Azure is thereby needed.

Chapter 5

5. Implementation

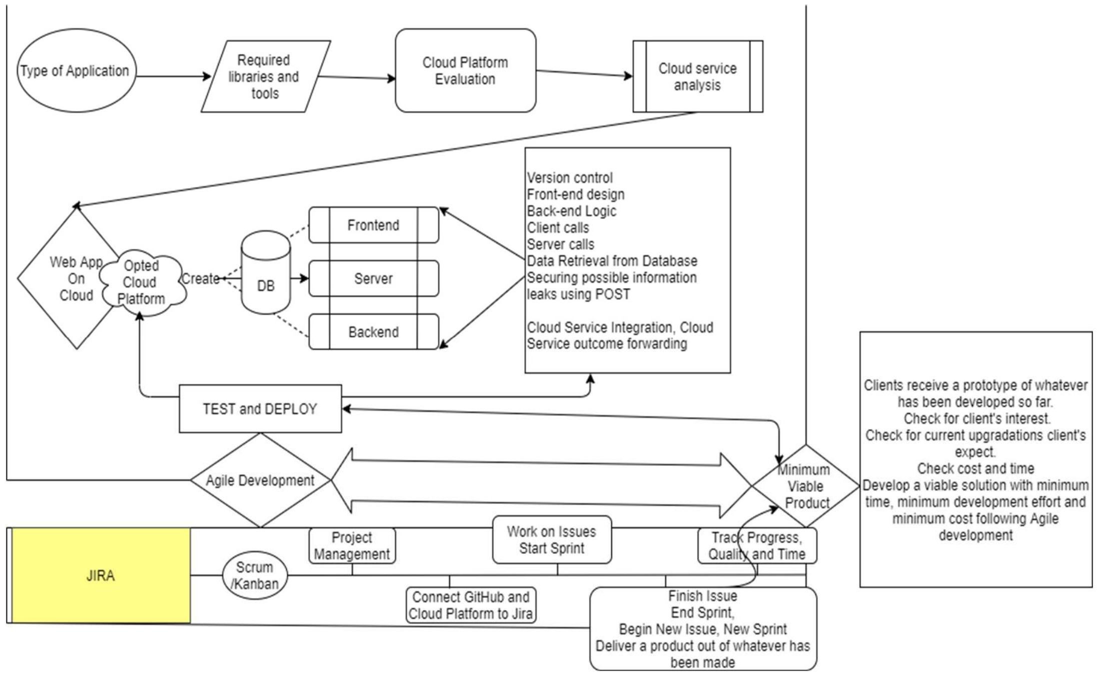

Type of Application
Required libraries and tools
Cloud Platform Evaluation
Cloud service analysis
Web App On Cloud
Opted Cloud Platform
Create
DB
Frontend
Server
Backend
Version control
Front-end design
Back-end Logic
Client calls
Server calls
Data Retrieval from Database
Securing possible information leaks using POST
Cloud Service Integration, Cloud Service outcome forwarding
TEST and DEPLOY
Agile Development
Minimum Viable Product
Clients receive a prototype of whatever has been developed so far.
Check for client's interest.
Check for current upgradations client's expect.
Check cost and time
Develop a viable solution with minimum time, minimum development effort and minimum cost following Agile development
JIRA
Scrum /Kanban
Project Management
Work on Issues Start Sprint
Track Progress, Quality and Time
Connect GitHub and Cloud Platform to Jira
Finish Issue
End Sprint,
Begin New Issue, New Sprint
Deliver a product out of whatever has been made

Figure 5.1: an abstracted view of implementation for the app development routine.

Figure 5.1 depicts the app creation process from the first analysis of the problem. Note that use of word '*abstracted'* as this diagram aims to just shed light on the proceedings of the application developed using this guide. In this paper one can see the validation of this model by developing an industry standard web application following the guide's modular steps where AWS is used as the cloud platform due to the application type.

The diagram highlights those contexts of the discussed case and gives an overview of what a development process on cloud native looks like, which is not so different from local development after all. Hence making cloud more familiar and friendlier to approach for all the software developers out there.

The initial stage tests and gather information regarding customer's interest and the type of deliverable that the customer is expecting. A minimum benchmark has to be set in regards of this. For any real time interactive application, a front end and a backend logic must exist with appropriate api and server calls and function handling. This project uses a specific set of learning curve but any front end and backend language can be applied so as to say. Testing each part of the code and deploying it on AWS Platform to see how it performs on cloud vs how it performs locally. Modularity is achieved by having version control ability from Git and GitHub. While doing so one needs to make sure about the time of collaboration and the time of development. Overall, keeping track of time is an essential part of having a large project, since there will be different modules or parts of apps which shall be created separately and then collaborated together as a whole package, and these tasks are moreover distributed to a team of developers. Complications like unwanted errors, loss of updating to branch repository, mishandled commits and loss of data, delay to produce models are just a few among the long list. To monitor the situation and avoid these problems one can chose to walk the agile path, this is achieved by Atlassian's Jira cloud tool. It is an issue manager and task distribution tool which also provisions agile frameworks i.e. Scrum and Kanban for the user's project depending upon the development strategy. Creating an issue in the issue pool helps to proceed with a module. It connects with user's cloud platform and user's GitHub repository to ease code management. It also features a monitoring ability which tells the time taken to end an issue, overdraft time, incomplete issues/error and even completed and solved issues are shown. User can provision a whole timeline of issues i.e. enter all the modules needed for the application and distribute it to his team and manage accordingly. This can only be done when the team is 100 percent sure about what they want from application and what the need to achieve that feat.

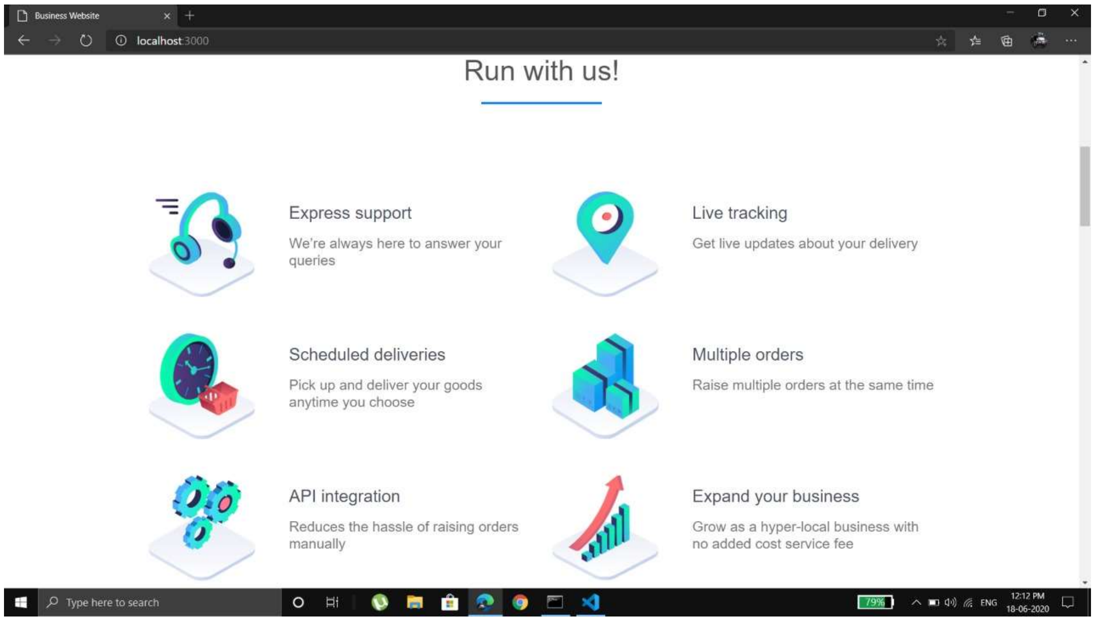
So, module by module issues are created and resolved, the code is deployed on cloud platform and tested there. After successful test a final commit is made to the Main Branch repository on GitHub and a different issue is created to develop another module and integrate it later again as such. Meanwhile a constant check and evaluation of interest and changes in customer's needs is noted. Changes according to that are implicated on the currently developed web application and presented to the customer for a green signal for full development. This is how the whole application is created and tested at the same time while also testing and ensuring customer-interest, market reaction and other cloud factors like deployment delay and latency etc. That can only be achieved on loud adoption using agile and MVP in combination.

Chapter 6

6. Results and Discussion

As an end product deliverable, an industry standard application is created following the guide’s procedures and thereby validating it. This application is rather complex to build. This project builds a (proprietary and commercial) business application for food ordering and delivery service with order tracking capabilities where end users can not only order food delivery but can also track their food while it is on its way to the customer. The app’s target audience comprises of all the possible actors there can be in food ordering and delivery space, the end user, the restaurants and also the delivery people who provide the delivery service. The technology stack used in preparation of this:

- Html
- CSS
- Bootstrap
- DOM
- JavaScript
- Node.js
- Express
- jQuery
- EJS
- MangoDB and Mongoose
- Google geo Location tracker Api
- Geo Distance matrix Api
- Payments Gateway Api
- Web Sockets and room creation
- Server Broadcasting
- Data retrieval
- Authentication for users



Run with us!

Express support
We're always here to answer your queries

Live tracking
Get live updates about your delivery

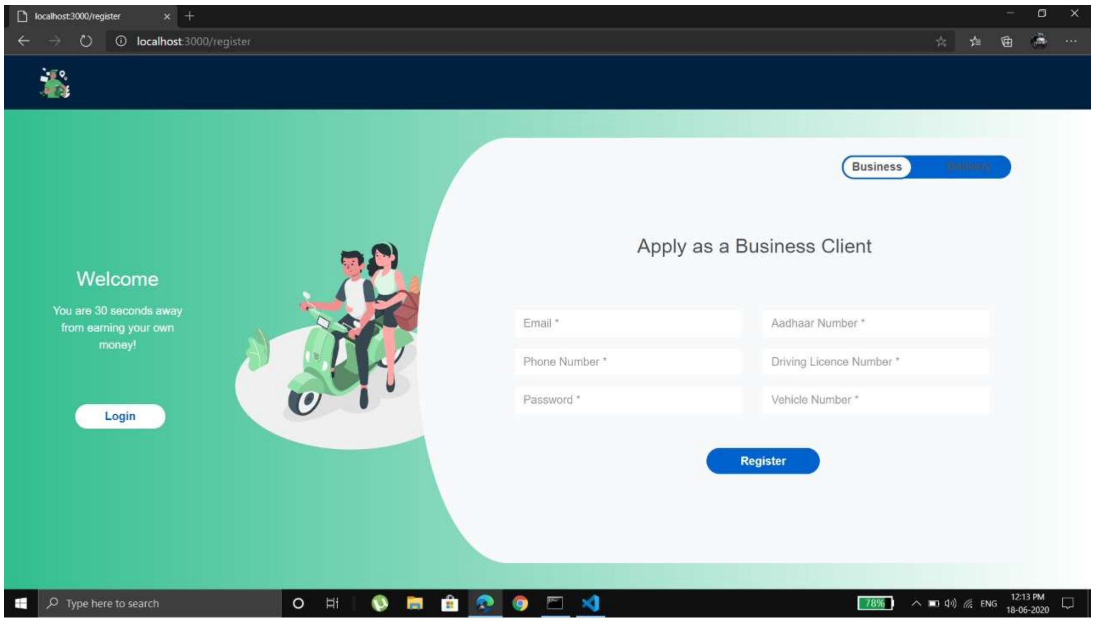
Scheduled deliveries
Pick up and deliver your goods anytime you choose

Multiple orders
Raise multiple orders at the same time

API integration
Reduces the hassle of raising orders manually

Expand your business
Grow as a hyper-local business with no added cost service fee

Figure 6.1 : A Screenshot displaying features of the developed web-application



Welcome

You are 30 seconds away from earning your own money!

Apply as a Business Client

Email *
Aadhaar Number *
Phone Number *
Driving Licence Number *
Password *
Vehicle Number *

Figure 6.2: A screenshot of registration page for Business clients and hotels

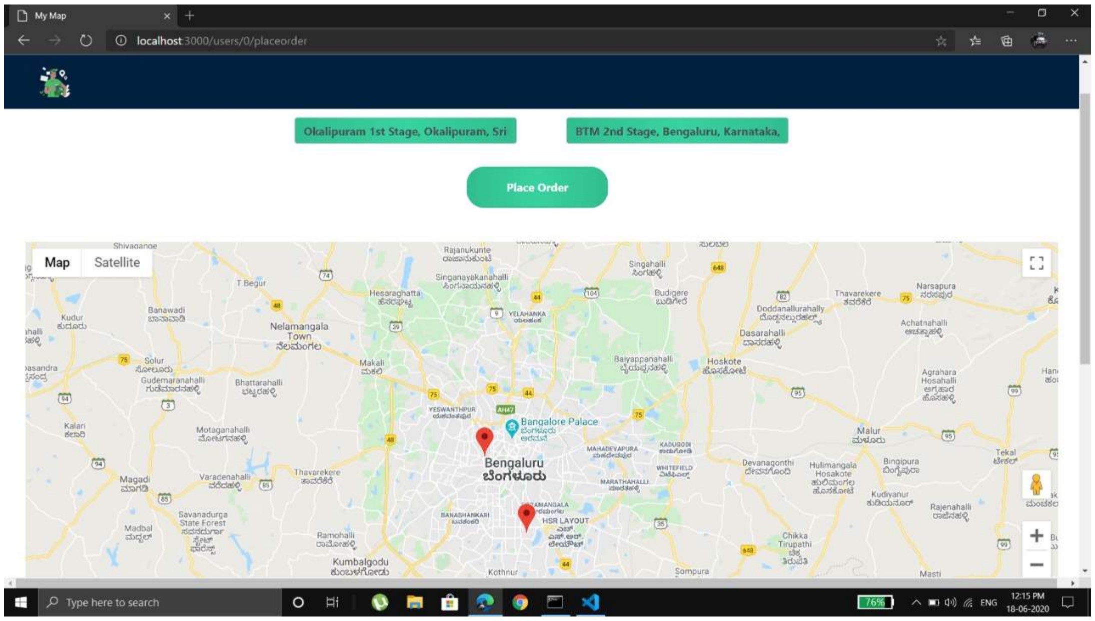
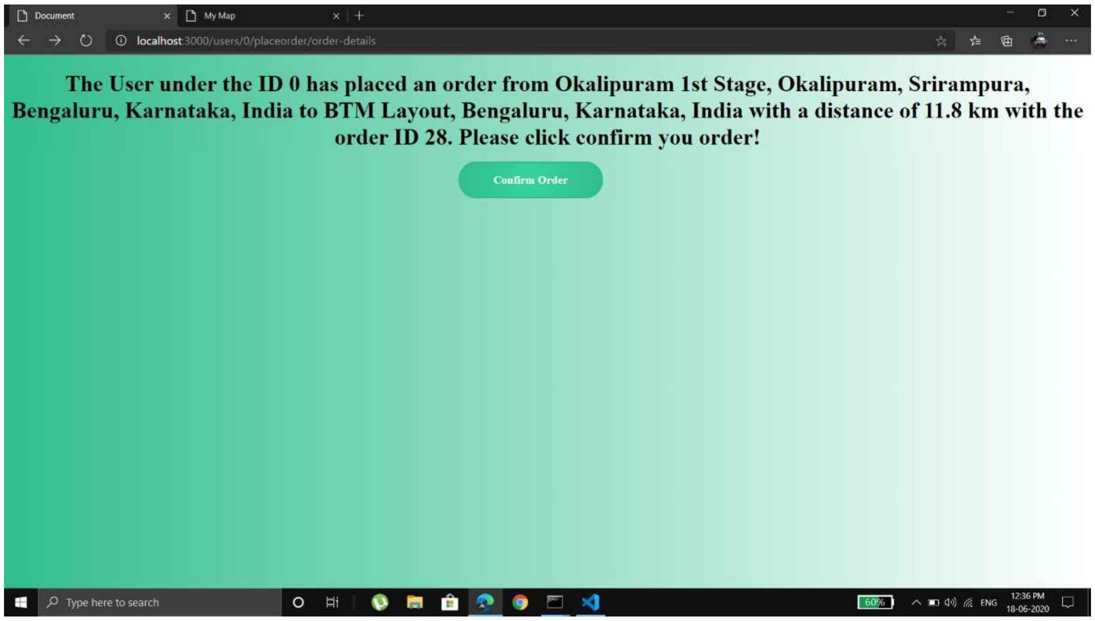
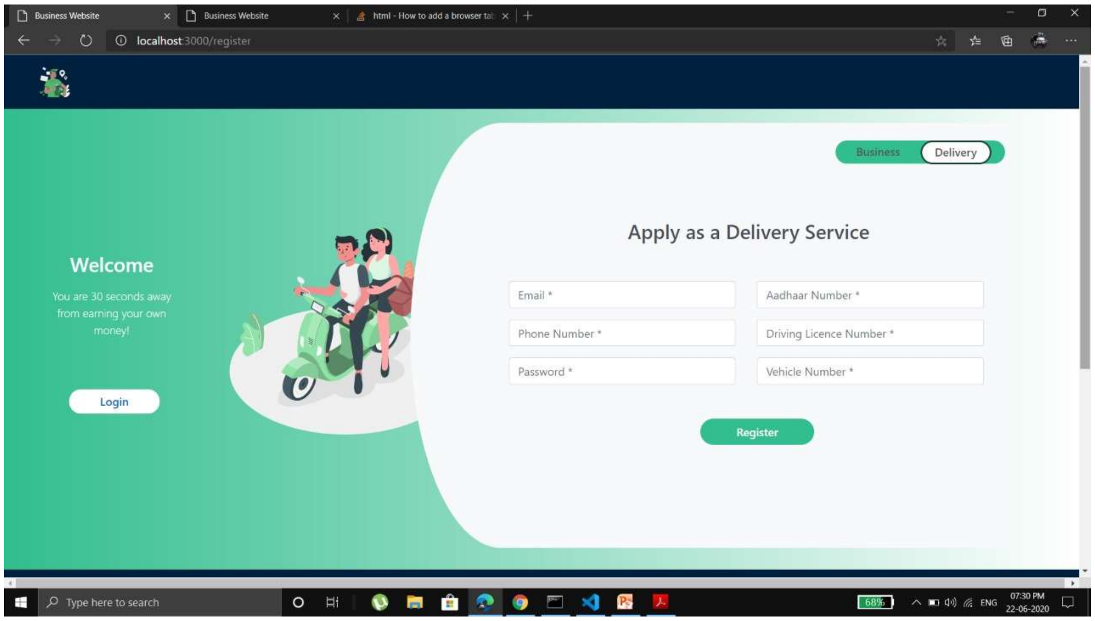

My Map
localhost:3000/users/0/placeorder
Okalipuram 1st Stage, Okalipuram, Sri
BTM 2nd Stage, Bengaluru, Karnataka,
Place Order
Map
Satellite
Bengaluru

Figure 6.3: A screenshot of live order tracking feature on web-app.



The User under the ID 0 has placed an order from Okalipuram 1st Stage, Okalipuram, Srirampura, Bengaluru, Karnataka, India to BTM Layout, Bengaluru, Karnataka, India with a distance of 11.8 km with the order ID 28. Please click confirm you order!

Figure 6.4: A screenshot for client order confirmation message.



Apply as a Delivery Service

Email *	Aadhaar Number *
Phone Number *	Driving Licence Number *
Password *	Vehicle Number *

Figure 6.5: A Screenshot of riders and delivery person registration page.

***Note**: The testing of the proposed Cloud SDLC is done by developing an industry-standard web-app following the given steps. Discussed app is in the constant development phase as the discussion proceeds.*

6.1 ***Discussion***:

Differences noticed while adopting Cloud-native approach as observed by on premise developers.

The development of the web-application was done both locally and on-cloud (cloud-natively) for the purpose of validation. Here are some points that has been listed to draw a distinguishing factor between traditional and cloud-native approach

Disclaimer: *These points do not talk about the obvious advantages of cloud like cost, reliability, performance, quality, accessibility etc. Instead, these are based on real time experience of on-premise developers making the shift to cloud-native development in order to validate the guide's model.*

- If one is totally new to the cloud-native development, setting the required configurations for user's environment may present as a big obstacle. Traditional methodology seems to have a rather easy or less stressful journey on this take. But, nothing ventured nothing gained. To make the shift it is essential for one to take the first step by willing to learn new technology and the knowledge that comes with it.
- To ease the pain, modern tools can greatly reduce the time to get started with development with a working development pipe-line. Google Cloud Build, Amazon Code Build are a few examples for it.
- As mentioned earlier there are almost no architectural standards and best practices for a cloud-native development. Therefore, implementing open-source solution will act as a risk to the business. Whereas, local development has the presence of communities that provide major support and best practices.
- The complexity just adds on. If the ongoing operations tend to becoming unmanageable, the developer has failed to ensure which technologies work well together and how. Understanding the key difference is important and essential.
- The developers need to take special care of the 'functionality' aspect of the cloud-native development. While wanting to do more with the cloud platform, one needs to always replicate user data pipeline which leads to reconstruction of pipeline in those data store. Before one knows about it there are three to four different structures of same replicated data which increases the cost heavily and suddenly.

- The code development now needs no infrastructure or library updating, since it is on the internet the cloud does the job.
- Cloud-native developer has direct access to auto scaling and load balancing features of the cloud platform that can be activated and deployed after a few small clicks and tweaks. Which, on the other hand, is a tedious task to achieve for an on premise developer.
- A special care should be taken by the cloud developers to not lose the ability to port while achieving their development in cloud. It is not easy to port an application which is localized for a specific cloud platform to another cloud platform.
- There are tons of work overs available for cloud developers which include usage of micro services, cloud platforms, and also containers like Kubernetes and Docker for asynchronous delivery. All of this comes bundled with the cloud native application development.
- One can integrate agile components such as containers to deliver a feature that is not only reusable but also can be integrated in well-described ways even in multi-cloud scenarios. This adds as an iterative automation feature for the development team.
- The ease of connected resources cannot be expressed more happily by a cloud developer. Locally, it is very irritating to connect an app with its networking components and security permissions since most of the times they are hard coded and tend to crash when being moved. Cloud-native approach, can re-platform the environment to make their application run on cloud. This involves work to implement changes in storage, networking and even database altogether.
- On premise applications need manual handling, whereas cloud-developers are looking for repeated tasks to automate. Due to the sheer speed of delivery by the automation and orchestration tools.
- The guide here promotes statelessness. A feature which silently indicates the loose coupling nature of cloud apps since they are not tied to infrastructure. The application runs in a highly distributed manner with instances coming and going, yet the app doesn’t fail to track where in unit of work the application is. The on premise applications are ‘stateful’, so as to say their state is stored on the infrastructure where the code runs. This may be the reason that sometimes the application breaks when adding new resources.

- The waterfall model lacks the ability of updating as all the requirements cannot be gathered in initial stages of project. The incremental model fails due to rising issues when joining iterations. Spiral Model is just plain costly and inapplicable to smaller projects. The Rapid Application Development is applicable to larger projects but comes in with unnecessary modelling skills as a mandate. While Agile model is beautiful carved, it fails due to the constantly shifting nature of customer. Agile works only when the customer is cent per cent sure about the product.
- Where older methodologies like waterfall and even agile fail, cloud developers have shifted to newer technologies like MVP and rapid iteration. This model combines the agile with MVP to make things simpler, minimum and deliverable by ensuring consumer's needs as well as satisfaction.

Note: *The amount of difference may or may not be same for every developer as this depends very much on the type of application that is being built on cloud. Since app that was developed to test and validate the model falls under the delivery logistics business web application, services used might be different than those used to create an AV (audio-video) sharing application or an online forum community-based web application. Nonetheless, the differences between local development and cloud native development will be visible to those who chose to make the shift.*

Conclusion and Future Scope

Successful development of a proper guide which can easily provide sufficient amount of help and visibility for Cloud-native web application development thereby promoting cloud adoption to the readers and end users by making use of Agile model combined with MVP development technique. This unlocks the potential to explore more on the cloud while contributing more solutions to the community. The opportunities for development are endless when cloud is taken into consideration and as discussed in the introduction, technology is deemed to change and update. By provisioning a guide for developing applications cloud-natively, the final aim is to ensure more end users make the shift to cloud and start using the technology to create more solutions for the community.

The security feature can be amped up to be the crux of every essential phase of the model and this is where the future of this model lies. A brief overview of this implementation is demonstrated in a way where a security evaluation is conducted right after the platform evaluation phase. Another dominant role of security can be having a privacy enabled design and planning of architecture right from the start where the developers work on privacy issues side by side as the modules are being rolled out. Testing of security by threat modelling and risk assessment after the normal cloud-based testing is conducted. Improvises in security and privacy keeps the future of cloud-native development running.

References

[1] S. Bibi, D. Katsaros, and P. Bozanis, "Application Development: Fly to the clouds or stay in-house?" in Proceedings of the 2010 19th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises, ser. WETICE '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 60–65. .

[2] S. Marston, Z. Li, S. Bandyopadhyay et al., "Cloud computing - The business perspective," Decision Support Systems, vol. 51, no. 1, pp. 176 – 189, 2011.

[3] T. Dillon, C. Wu, and E. Chang, "Cloud Computing: Issues and challenges," in Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on, 2010, pp. 27–33.

[4] "Using the Cloud to Facilitate Global Software Development Challenges" by Sajid Ibrahim Hashmi, Viktor Clerc, Maryam Razavian, Christina Manteli, Damian Andrew Tamburri, Patricia Lago, Elisabetta Di Nitto, and Ita Richardson.

[5] "A new approach to Software as Service Cloud" by Gurudatt Kulkarni, Pragati Chavan, Hemant Bankar, Kundilk Koli and Vidya Waykule

[6] "From Software as a Good to Software as a Service: Preparing the Evolution of Software Products into the Cloud" 2012 IEEE 6th International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA) Closing Keynote by Leire Orue-Echevarria Arrieta

[7] Z. Li, L. O'Brien, H. Zhang, and R. Cai, "On a catalogue of metrics for evaluating commercial cloud services," in Grid Computing (GRID), 2012 ACM/IEEE 13th International Conference on, sept. 2012, pp. 164 –173.

[8] Rakesh Pandit "Comparison of different approaches to evaluate cloud computing services" Department of Computer Science University of Helsinki

[9] "Analysis for Security Implementation in SDLC" by Gaurav Raj, Dr. Dheerendra Singh, Dr. Abhay Bansal.

APPENDIX-I

PHOTOGRAPHS

STUDENT NAME	EMAIL ID	PERMANENT ADDRESS	PHONE NUMBER	PHOTOGRAPH
Jatin Mehrotra	jatinjmehrotra@gmail.com	Bhopal, Madhya Pradesh.	+91 9826159719	
Herish R Chovatiya	herish.rc@gmail.com	Bangalore, Karnataka.	+91 9066436692	
Mahir Gada	gadamahir1710@gmail.com	Mumbai, Maharashtra.	+91 8369771474	
Hrishikesh D Kakkad	hrishidkakkad@gmail.com	Bangalore, Karnataka	+91 9845744554	

APPENDIX - II

SOURCE CODE

GitHub: (link will be uploaded shortly, after implementing all the censorship and privacy measures.)

Note: *This web application is still in development phase and shall be proprietary and may be commercialized of the creators wish to do it.*