Department of Computer Science and Engineering

GLA University
Mathura- 281406, INDIA
2020

Project Report
SoPRO

Submitted To:
Manoj Varshney

Submission Date:
25th Nov, 2020

Group Profile

Project Name: SoPRO

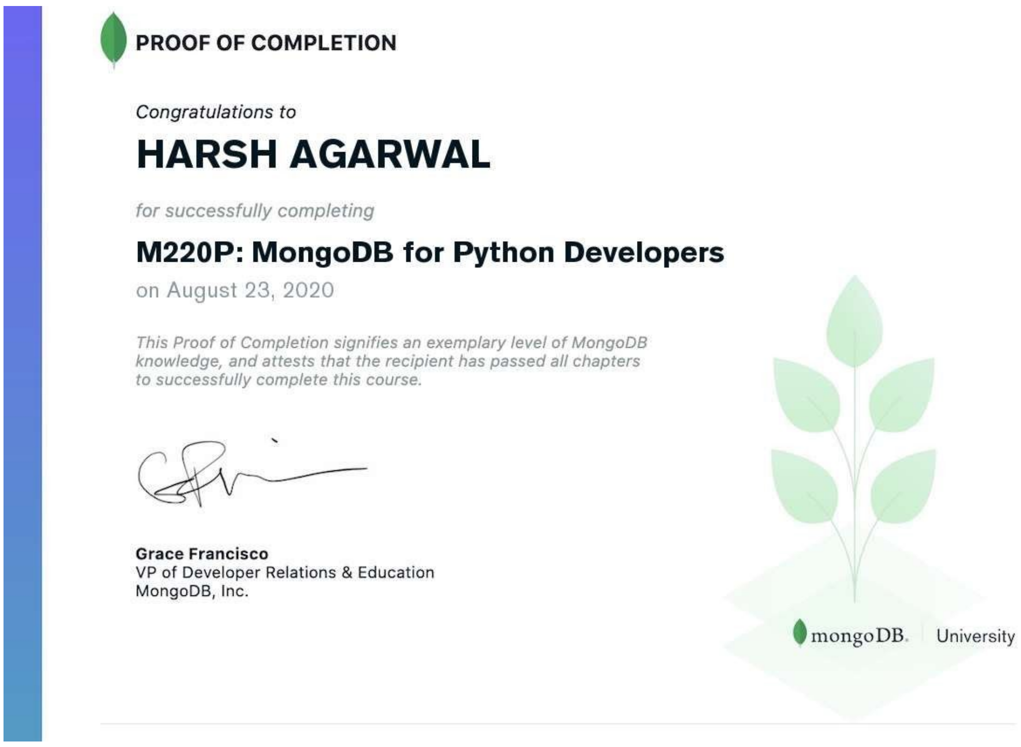
Course Title: Bachelor of Technology, CSE

S.No	Name	University RollNo.
1	Sameer Gautam	181500612
2	Anirudh Kushwah	181500093
3	Pradhum Bansal	181500460
4	Harsh Agarwal	181500246
5	Rishi Singh	181500568

Abstract

The project report has been prepared based on available data, forecasts provided by experts. The real life situation can be little different depending on the circumstances.The project is considered as not for profit.The member working in the team would gain experience and would be able to help the upcoming programmers to solve their problems faster. Full effort has been given to complete each and every pros and cons, so that they are taken into account. However, the report isn’t full proof. There is always room for improvement.

Certifications



PROOF OF COMPLETION

Congratulations to

HARSH AGARWAL

for successfully completing

M220P: MongoDB for Python Developers

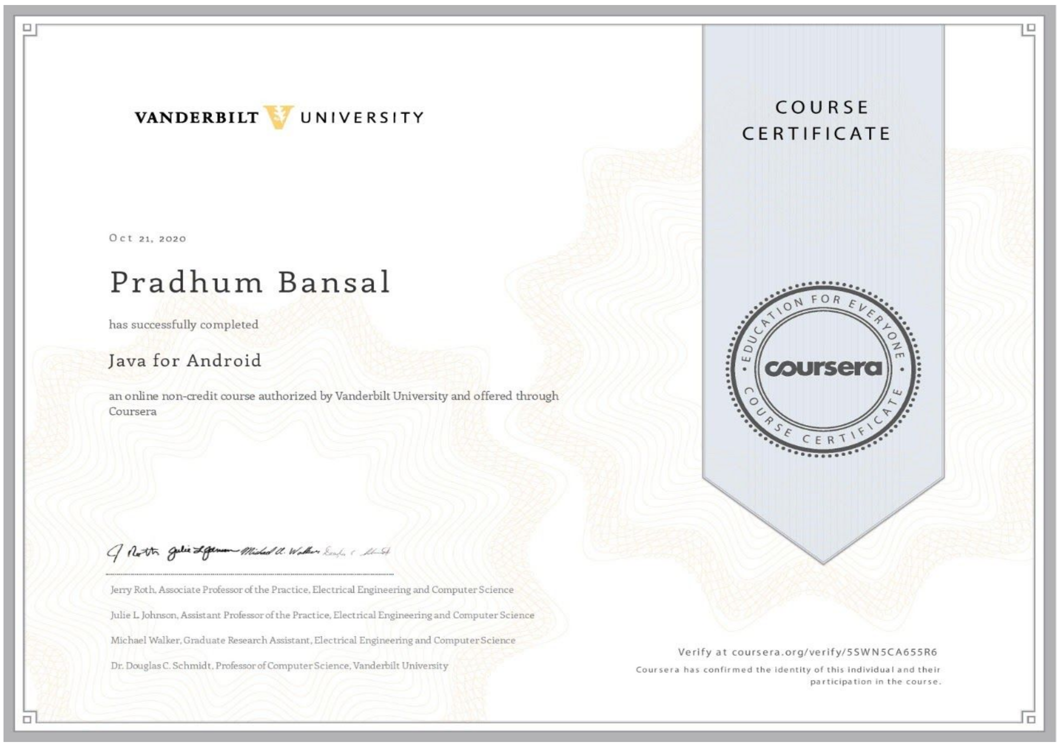
on August 23, 2020

This Proof of Completion signifies an exemplary level of MongoDB knowledge, and attests that the recipient has passed all chapters to successfully complete this course.

[illegible]

Grace Francisco
VP of Developer Relations & Education
MongoDB, Inc.

mongoDB. University



VANDERBILT UNIVERSITY

COURSE CERTIFICATE

Oct 21, 2020

Pradhum Bansal

has successfully completed

Java for Android

an online non-credit course authorized by Vanderbilt University and offered through Coursera

[illegible]

Jerry Roth, Associate Professor of the Practice, Electrical Engineering and Computer Science

Julie L Johnson, Assistant Professor of the Practice, Electrical Engineering and Computer Science

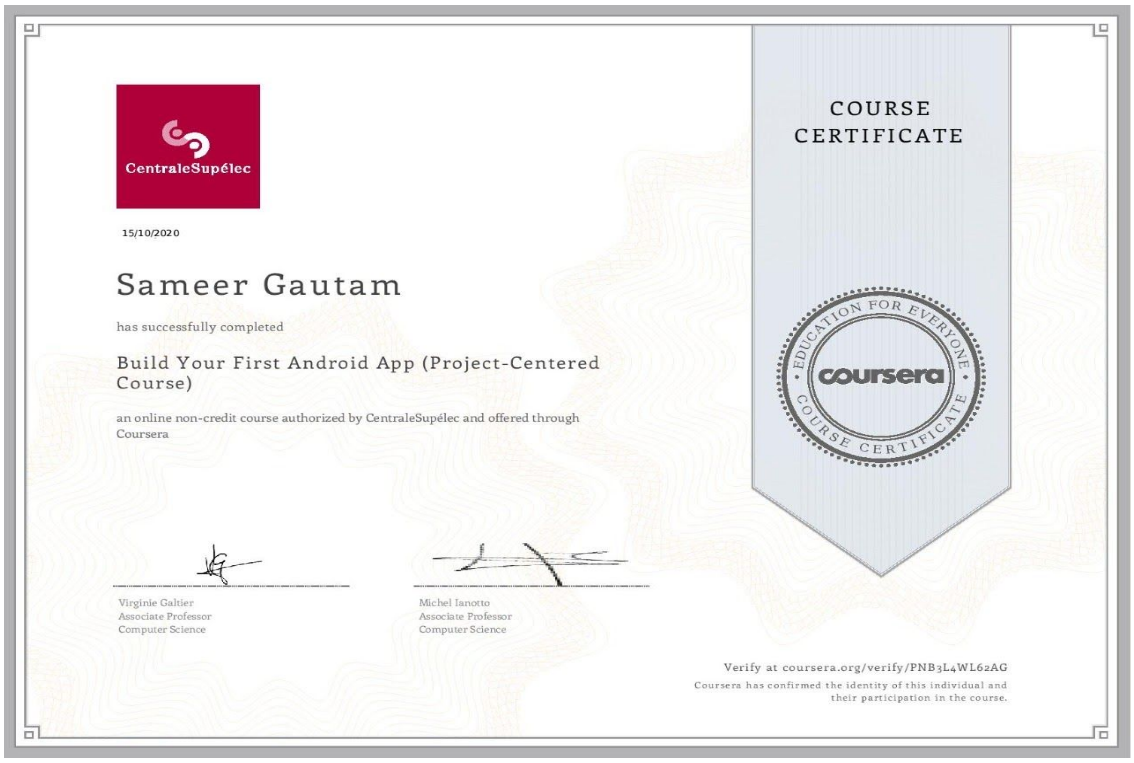
Michael Walker, Graduate Research Assistant, Electrical Engineering and Computer Science

Dr. Douglas C. Schmidt, Professor of Computer Science, Vanderbilt University

EDUCATION FOR EVERYONE coursera COURSE CERTIFICATE

Verify at coursera.org/verify/5SWN5CA655R6

Coursera has confirmed the identity of this individual and their participation in the course.



CentraleSupélec

COURSE CERTIFICATE

15/10/2020

Sameer Gautam

has successfully completed

Build Your First Android App (Project-Centered Course)

an online non-credit course authorized by CentraleSupélec and offered through Coursera

EDUCATION FOR EVERYONE · coursera · COURSE CERTIFICATE

[illegible]

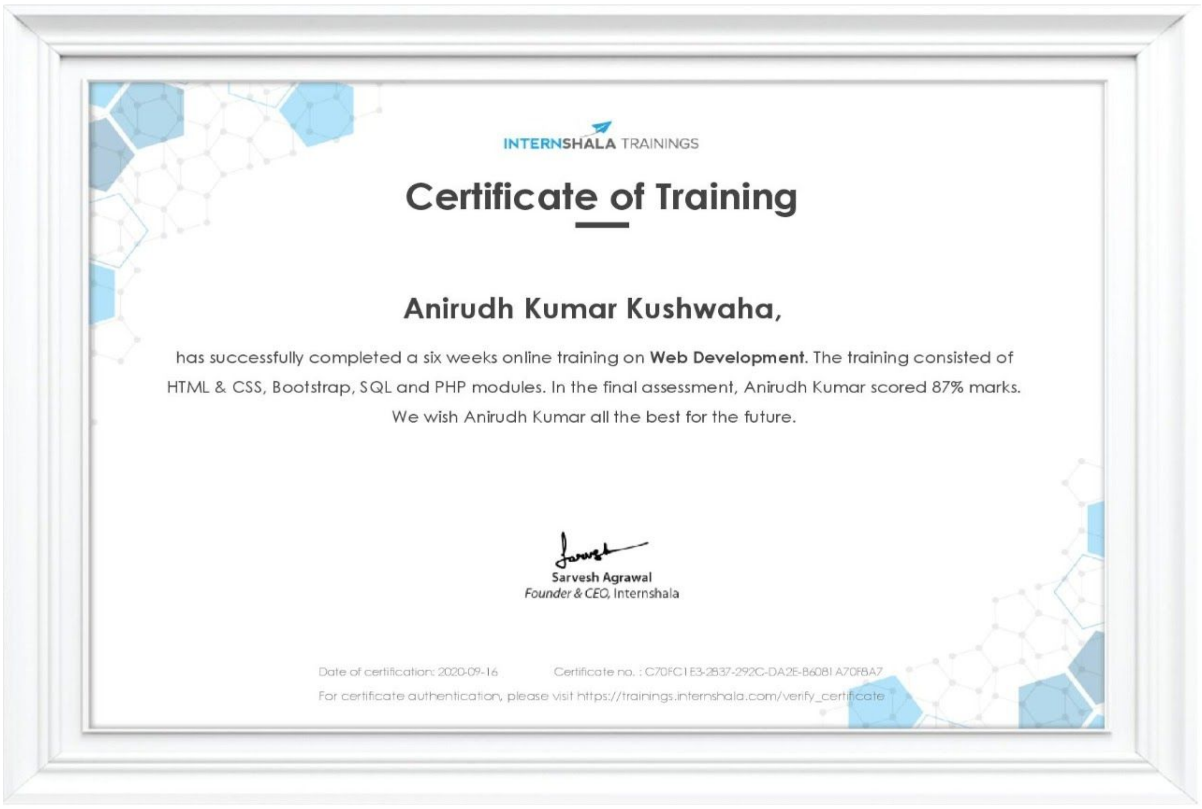
Virginie Galtier
Associate Professor
Computer Science

[illegible]

Michel Ianotto
Associate Professor
Computer Science

Verify at coursera.org/verify/PNB3L4WL62AG

Coursera has confirmed the identity of this individual and their participation in the course.



INTERNSHALA TRAININGS

Certificate of Training

Anirudh Kumar Kushwaha,

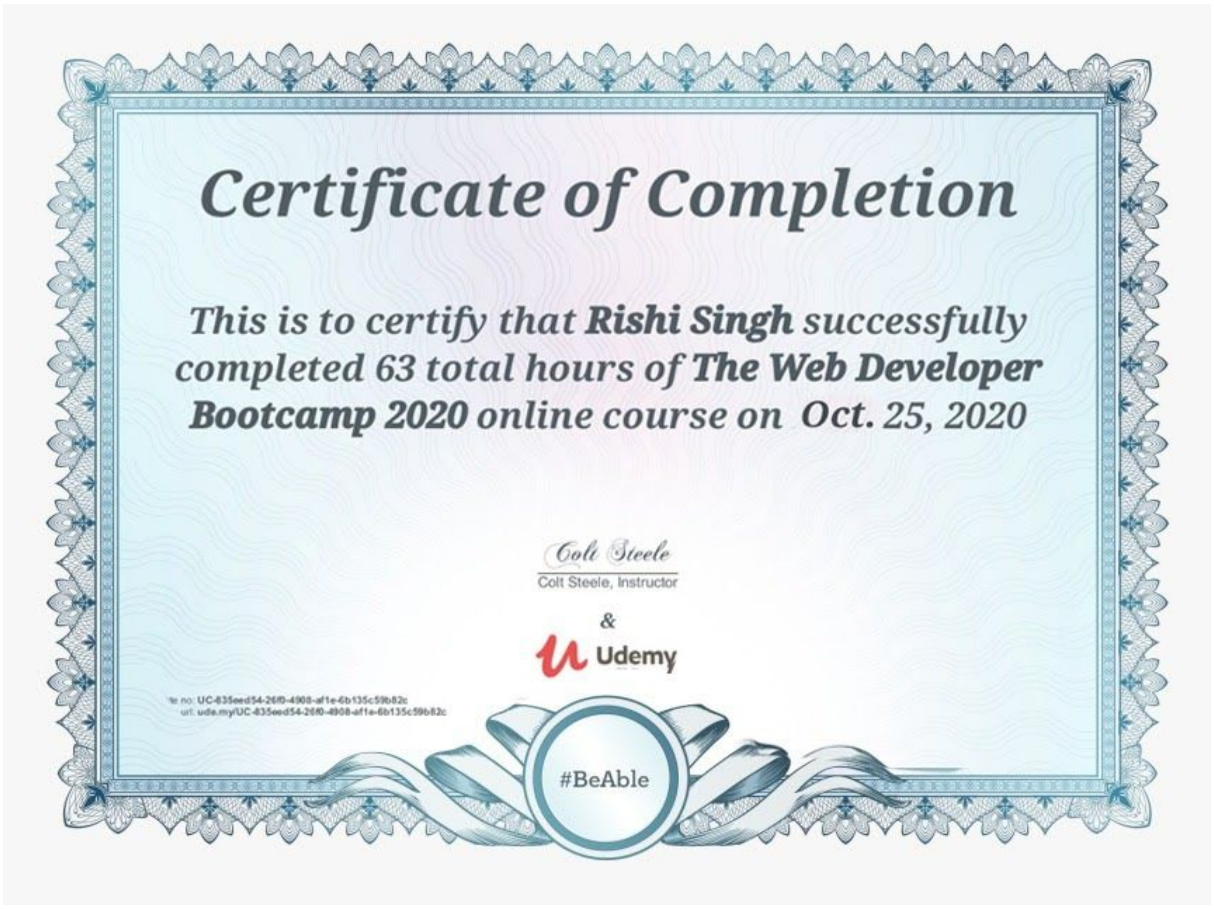
has successfully completed a six weeks online training on **Web Development**. The training consisted of HTML & CSS, Bootstrap, SQL and PHP modules. In the final assessment, Anirudh Kumar scored 87% marks. We wish Anirudh Kumar all the best for the future.

[illegible]

Sarvesh Agrawal
Founder & CEO, Internshala

Date of certification: 2020-09-16   Certificate no. : C70FC1E3-2837-292C-DA2E-B6081A70F8A7

For certificate authentication, please visit https://trainings.internshala.com/verify_certificate



Certificate of Completion

This is to certify that ***Rishi Singh*** *successfully completed 63 total hours of* ***The Web Developer Bootcamp 2020*** *online course on* ***Oct.*** *25, 2020*

[illegible]
Colt Steele, Instructor

&

Udemy

te no: UC-835eed54-26f0-4908-af1e-6b135c59b82c
url: ude.my/UC-835eed54-26f0-4908-af1e-6b135c59b82c

#BeAble

Table Of Content

1.Introduction

1.1 Overview

- All the functional/non-functional requirements, corresponding DFD’s, UML and Use Case Diagrams have been organized in this report. Along with these designs, this report also contains the essential data of this project.
- The complete description of the application followed by the functionalities has been listed initially. Later on, application has been described diagrammatically with the help of different designing tools like Data Flow Diagram, Use Case Diagram, Interaction Diagram and E-R Diagram.

1.2 Problem Statement

Programmers who are learning a new skill or working on a project may face difficulty during their work and may not have someone in their knowledge who's familiar with that technology. So we will make a good learning platform where programmers around the globe would be able to collaborate and share their own codes.

To create a social platform for learners that'll help resolve queries or get new ideas. Existing platforms coding forums have a certain issues such as :-

- **Letting new users face the entire community at once-**
A new user who just started and needs help with his coding might get lost in the vast platform. Search results may be irrelevant and unhelpful at times.
SoPRO will tackle this by sorting the posts language-wise and letting the user see the content that is meant for him.

- **Many people don’t get helped-**

  It is likely that an error faced by a programmer has already been seen and solved by another programmer on such platforms but the various parameters of a program may be different, their causes may be different as each person has their own way of coding.
  SoPRO tackles this by helping programmers find issues in his OWN personal code rather than somebody else’s similar code.

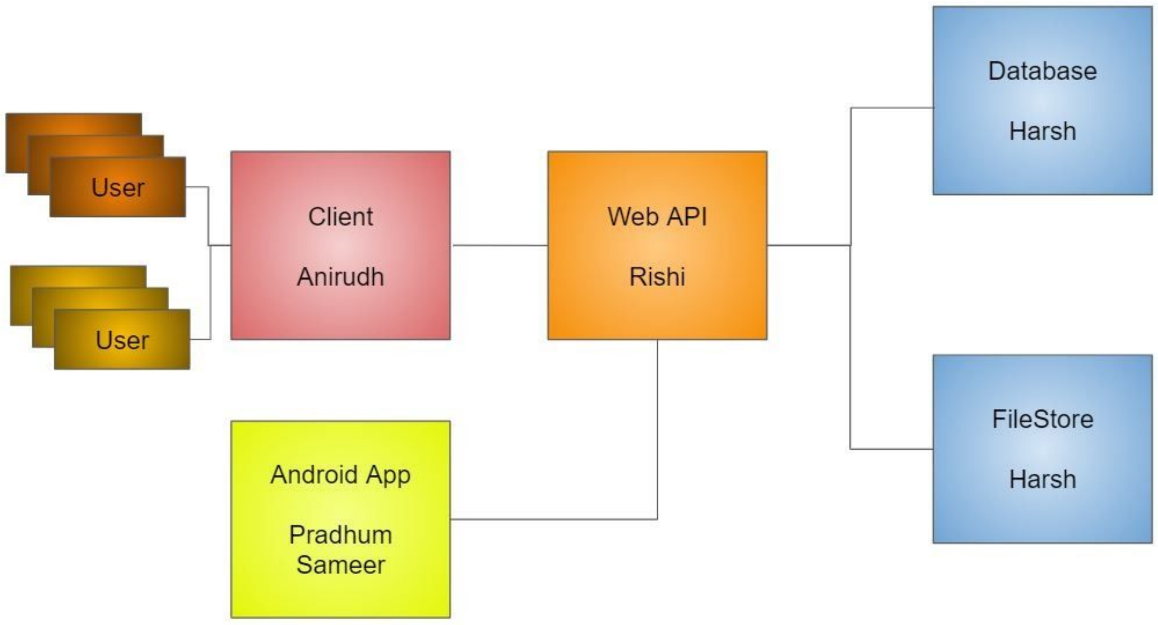
- **Beginners may get disappointed-**

  Often it is seen that when a new contributor joins such a community he faces various issues such as being downvoted for no reason.

1.3 Objective

SoPRO is an error solving platform for novice and enthusiastic programmers which they face during the beginning phase of programming and development. It also provides a platform for programmers to debug and resolve their code error. In addition to this it also gives personalized solutions to errors and allows the users to share new ideas and thoughts on coding. SoPRO tackles this with a transparent medium where there’s no such concept of a downvote as ‘EACH ANSWER MATTERS’.

2. Roles


User
User
Client
Anirudh
Web API
Rishi
Database
Harsh
FileStore
Harsh
Android App
Pradhum
Sameer

3. Software Requirement Analysis

3.1 System Analysis

System analysis is a process of collecting and interpreting facts, identifying the problems, and decomposition of the system into its components.

It is a process of studying a system in order to define its goals or purposes and to discover operations and procedures for accomplishing them most efficiently.

Here, the problem is novice programmers face a common problem of exposure. Most of the time, they remain unaware that the queries they post on other online platforms are not answered by other more experienced or professional programmers. Thus, they abandon that part of programming due to lack of knowledge.

3.2 Software Interfaces

The application runs in the latest version of Chrome or Firefox browser on Windows, Linux and Mac. Along with a separate Android application for AndroidOS.

3.3 Hardware Interfaces

The application is intended to be a stand-alone, single-user system. The application will run on an android device along with a website on desktop/laptop. No further hardware devices or interfaces will be required.

3.4 User Interfaces

The application GUI provides menus, toolbars, buttons, panes, containers, grids allowing for easy control by a keyboard and a mouse. It also provides a navigation menu in Android applications for ease of use.

3.5 System Analysis

- Startup Time: The application should display the opened content within 10s after it is started.
- Edit Response Time: The application displays updated values within 1s after the user triggers the edit operation.
- Smooth Scrolling: While a user scrolls the requirements table, the application does not display scrolling jerks longer than 200ms.

3.6 Communications Interfaces

The application shall communicate with the database and software services via API function calls. Because the application will be written in Java for android and Javascript for WebApp.

HTTP

The Hypertext Transfer Protocol (HTTP) is an application layer protocol for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web, where hypertext documents include hyperlinks to other resources that the user can easily access, for example by a mouse click or by tapping the screen in a web browser.
In response to HTTP requests, servers often issue response codes, indicating the request is being processed, that there was an error in the request or that the request is being redirected. Common response codes include:

- 200 OK. This means that the request, such as GET or POST, worked and is being acted upon.
- 300 Moved Permanently. This response code means that the URL of the requested resource has been changed permanently.

- 401 Unauthorized. The client -- the user making the request of the server -- has not been authenticated.
- 404 Not Found. This is the most frequent and most recognized error code. It means that the URL is nor recognized or the resource at the location does not exist.
- 500 Internal Server Error. The server has encountered a situation it doesn't know how to handle.

3.7 Memory constraints

The application shall allow users to upload data up to total size up to 10MB.

3.8 Operations:

- The application allows users to expand and collapse all changes in the pane.
- The application displays all changes/updates of the selected requirement.
- Each displayed post contains the author, title and description of the change.

4. Methodology

We are using Microservice Architecture to provide the Independency of using a tech stack to each member, which would ensure a great environment where everyone would be choosing their preferred language and tools. What are microservices?

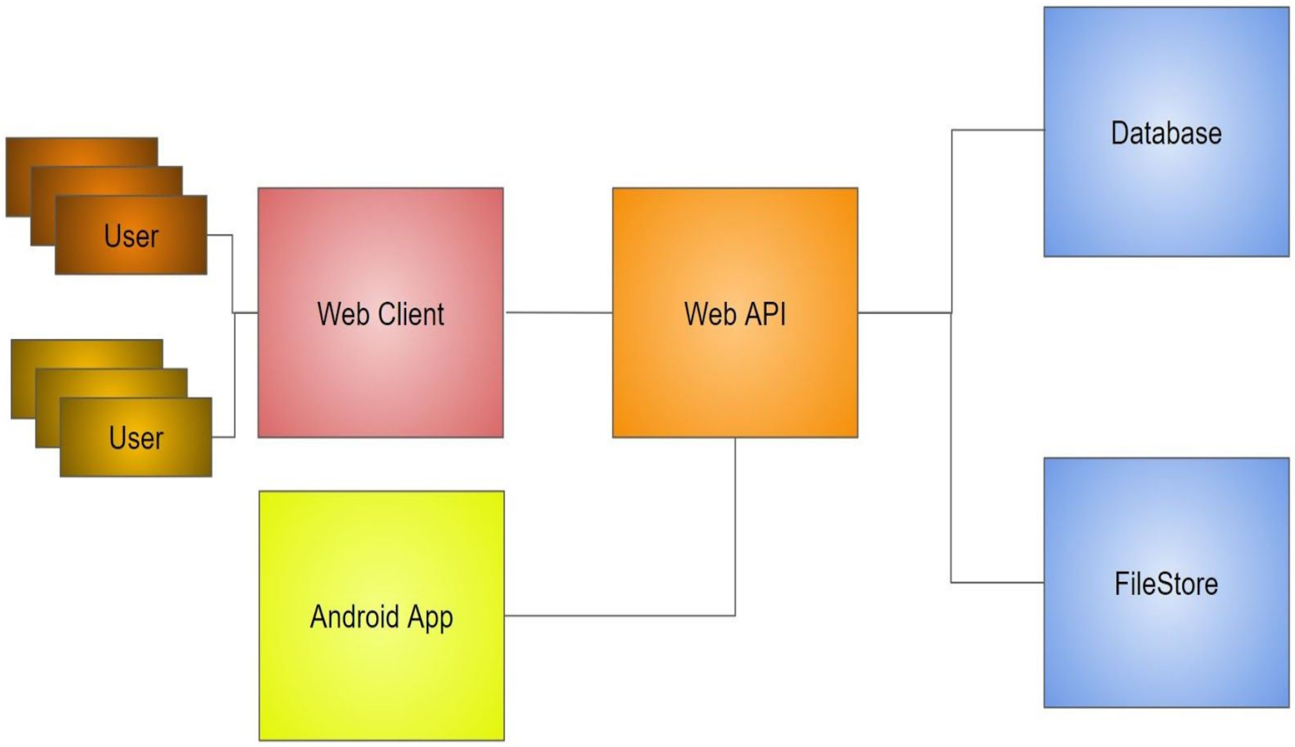
What are microservices?

Microservices -- also known as the microservice architecture - is an architectural style that structures an application as a collection of services that are:

- Highly maintainable and testable
- Loosely coupled
- Independently deployable
- Organized around business capabilities
- Owned by a small team

The microservice architecture enables the rapid, frequent, and reliable delivery of large, complex applications. It also enables an organization to evolve its technology stack.

4.1 Summary



We divided the whole into the following services:

- Web Client
- Web API
- Database
- FileStore
- Android App

4.1.1 Web Client

WebClient would consist of :

- Backend will be the logical side of the WebClient which will help to provide the dynamic nature of WebClient and would make requests to the API according to the Users action.

- FrontEnd will be the visual side of the WebClient of the microservice project, it will provide an interface for users to login/register and use various functionality.

4.1.2 Web API (RESTful)

What is REST

REST is an acronym for **RE**presentational **S**tate **T**ransfer. It is an architectural style for **distributed hypermedia systems** and was first presented by Roy Fielding in 2000 in his famous dissertation.

Like any other architectural style, REST also does have it’s own 6 guiding constraints which must be satisfied if an interface needs to be referred as **RESTful**. These principles are listed below.

Guiding Principles of REST

1. **Client–server** – By separating the user interface concerns from the data storage concerns, we improve the portability of the user interface across multiple platforms and improve scalability by simplifying the server components.
2. **Stateless** – Each request from client to server must contain all of the information necessary to understand the request, and cannot take advantage of any stored context on the server. Session state is therefore kept entirely on the client.
3. **Cacheable** – Cache constraints require that the data within a response to a request be implicitly or explicitly labeled as cacheable or non-cacheable. If a response is cacheable, then a client cache is given the right to reuse that response data for later, equivalent requests.

4. **Uniform interface** – By applying the software engineering principle of generality to the component interface, the overall system architecture is simplified and the visibility of interactions is improved. In order to obtain a uniform interface, multiple architectural constraints are needed to guide the behavior of components. REST is defined by four interface constraints: identification of resources; manipulation of resources through representations; self-descriptive messages; and, hypermedia as the engine of application state.
5. **Layered system** – The layered system style allows an architecture to be composed of hierarchical layers by constraining component behavior such that each component cannot "see" beyond the immediate layer with which they are interacting.
6. **Code on demand (optional)** – REST allows client functionality to be extended by downloading and executing code in the form of applets or scripts. This simplifies clients by reducing the number of features required to be pre-implemented.

Application Programming Interface(API) will receive requests from Client server, process them and return a response depending on the nature of the request made by the client.

API would be using bcrypt as a password-hashing function followed by Salting the hashes to save in user login credentials inside databases so not even the developer could access a user account with login credential inside the database.

The API will be implemented with an Authorization method to check whether the User trying to make a request is authorized with a valid JWT(JSON Web Token) token or not and return a response accordingly.

4.1.3 Database

Most databases can be categorized as either:

- Relational
- Non-relational

Relational database

A relational database typically stores information in tables containing specific pieces and types of data. For example, a shop could store details of their customers' names and addresses in one table and details of their orders in another. This form of data storage is often called structured data.

Relational databases use Structured Query Language (SQL). In relational database design, the database usually contains tables consisting of columns and rows. When new data is added, new records are inserted into existing tables or new tables are added. Relationships can then be made between two or more tables.

Relational databases work best when the data they contain doesn't change very often, and when accuracy is crucial. Relational databases are, for instance, often found in financial applications.

Non-relational database

Non-relational databases (often called NoSQL databases) are different from traditional relational databases in that they store their data in a non-tabular form. Instead, non-relational databases might be based on data structures like documents. A document can be highly detailed while containing a range of different types of information in different formats. This ability to digest and organize various types of information side-by-side makes non-relational databases much more flexible than relational databases.

Non-relational databases are often used when large quantities of complex and diverse data need to be organized. For example, a large store might have a database in which each customer has their own document containing all of their information, from name and address to order history and credit card information. Despite their differing formats, each of these pieces of information can be stored in the same document.

Non-relational databases often perform faster because a query doesn’t have to view several tables in order to deliver an answer, as relational datasets often do. Non-relational databases are therefore ideal for storing data that may be changed frequently or for applications that handle many different kinds of data. They can support rapidly developing applications requiring a dynamic database able to change quickly and to accommodate large amounts of complex, unstructured data.

The benefits of a non-relational database

Today's applications collect and store increasingly vast quantities of ever-more complex customer and user data. The benefits of this data to businesses, of course, lies in its potential for analysis. Using a non-relational database can unlock patterns and value even within masses of variegated data.

There are several advantages to using non-relational databases, including:

- Massive dataset organization
  In the age of Big Data, non-relational databases can not only store massive quantities of information, but they can also query these datasets with ease. Scale and speed are crucial advantages of non-relational databases.
- Flexible database expansion
  Data is not static. As more information is collected, a non-relational database can absorb these new data points, enriching the existing database with new levels of granular value even if they don't fit the data types of previously existing information.
- Multiple data structures
  The data now collected from users takes on a myriad of forms, from numbers and strings, to photo and video content, to message histories. A database needs the ability to store these various information formats, understand relationships between them, and perform detailed queries. No matter what format your information is in, non-relational databases can collate different information types together in the same document.

- Built for the cloud

  A non-relational database can be massive. And as they can in some cases grow exponentially, they need a hosting environment that can grow and expand with. The cloud’s inherent scalability makes it an ideal home for non-relational databases.

4.1.4 FileStore(AWS S3)

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. This means customers of all sizes and industries can use it to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides easy-to-use management features so you can organize your data and configure finely-tuned access controls to meet your specific business, organizational, and compliance requirements. Amazon S3 is designed for 99.999999999% (11 9's) of durability, and stores data for millions of applications for companies all around the world.

4.1.5 Android App

Our team will also provide Android applications for cross-platform functioning, resulting in a larger user base for creating communities. Thus, increasing the number of problem solvers and decreasing waiting time.

5. Environments

In today's modern business environment, it is imperative for product development teams to maintain an optimal workflow if they hope to remain relevant in the competitive market. Having a well-tuned workflow not only keeps a team productive, but it also helps them deliver software that is reliable and in a timely manner. Implementing DevOps has become an important requirement for any team maintaining a large project or working on multiple projects. DevOps provides the process and the tools that keep teams working effectively and efficiently.

No matter what your DevOps process looks like and the various tools you use, a recommended practice for any significant software development project is the use of multiple environments. Using multiple environments ensures that your software is rigorously tested before it is deployed and made available to users.

An example setup could have development, staging and production environments:

5.1 Development

The development environment would be the first line of defense against bugs. Here, developers deploy their code and test any newly implemented features. Any bugs found are dealt with before re-deploying for further testing. The process is iterated until the code is ready for the next stage of testing.

5.2 Staging

Once developers are satisfied with their code and consider it fairly stable, it is then deployed to the staging environment for further testing. This is where Quality Assurance (QA) is performed. Testers access the staging servers and ensure that the application works as it should. They run test cases to detect bugs and run performance tests to find areas that could be improved. Any bugs or enhancements are reported back to the developers and the process is repeated until the code passes the staging phase.

5.3 Production

Once the code has been thoroughly tested, it is then pushed to production where it is made available to end-users.

The above environment setup is just an example and shows the three common environments for software projects. Your setup may vary according to your project and team's needs. You could have more or fewer environments set up, for instance, some people prefer to have a Pre-production environment to further test the code before the final deployment to Production, and others maintain separate Staging and QA environments where developers perform further tests (e.g. integration tests) in the Staging environment and the QA environment is exclusively used for quality assurance testing.

6. System Study

6.1 Authentication

6.1.1 Description and Priority

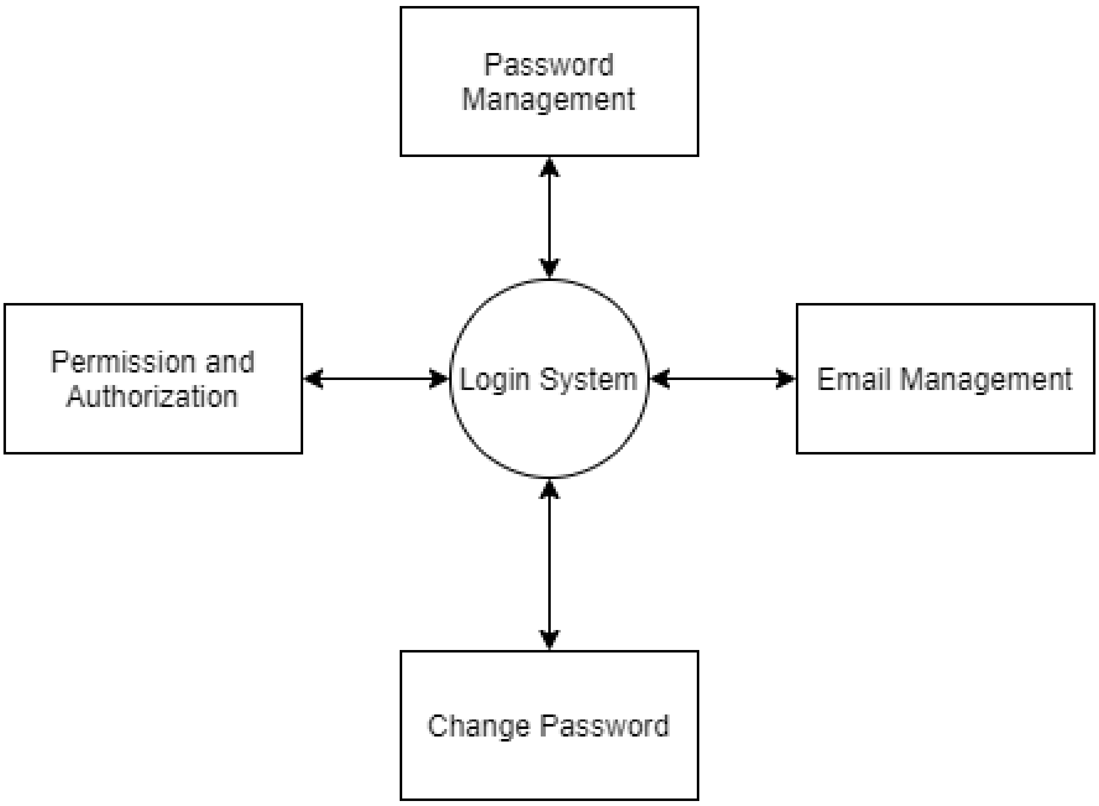
This feature will give the user a secure and simple login screen. The login is enabled for the administrator use only. It has only a limited and handful of input capacity of a limited number of administrators. Anybody else cannot enter the routine settings and hamper the system.

6.1.2 Stimulus/Response Sequences

It will consist of three basic fields - Username, Email and Password. A message is displayed at the screen defining the login is for the user only. There is a button Login for submitting the entered data. On successful entry the user will be prompted from login screen to home screen of the application. If data submission fails, the user will be redirected towards the login screen with the error message.

6.1.3 Functional Requirements

The Most important function of the login page is to provide access only to the registered users only.



6.2 Calling of Web Service

In this module, the user will need to select details such as the title of the post, description of post and the image for which the post is created. After doing so, the user needs to call the web service by clicking a button provided on the screen. The web service thus invoked would return the confirmation message that the post has been uploaded.

6.3 Display post

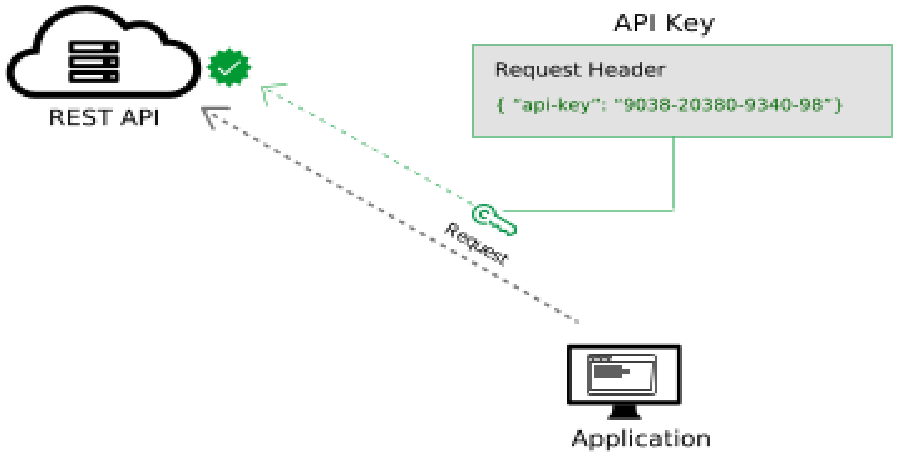
Once the post has been created successfully, the user can anytime view the post on the main screen and the section ‘My Post’. The information thus displayed would include the title of the post, user image who posted it and the image.

6.4 Authorization

Authorization is the process of controlling permissions. This can include permissions to perform actions, access systems, create, view, delete, change information, etc…
There are several methods for authorization. The following are various types of API authorization you might encounter:

- API Keys:

  Most APIs require you to sign up for an API key in order to use the API. The API key is a long string that you usually include either in the request URL or request header. The API key mainly functions as a way to identify the person making the API call (authenticating you to use the API). The API key might also be associated with a specific app that you register.



- Basic Auth:

  Another type of authorization is called Basic Auth. With this method, the sender places a username:password into the request header. The username and password are encoded with Base64, which is an encoding technique that converts the username and password into a set of 64 characters to ensure safe transmission. Example:

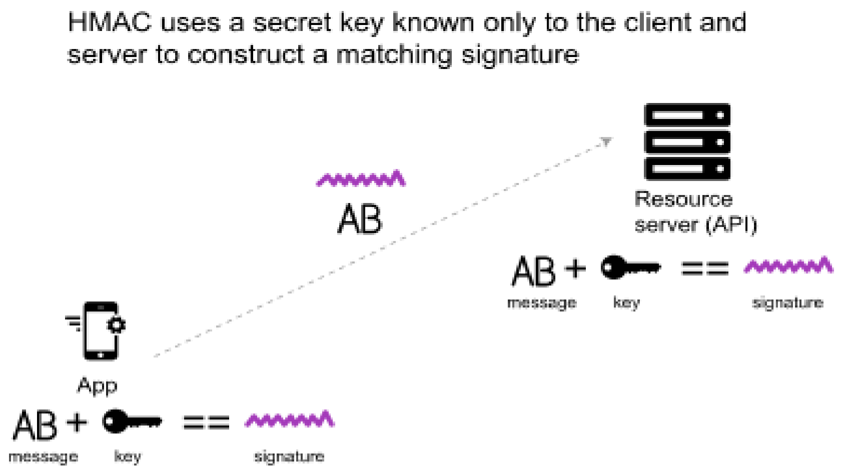
  Authorization: Basic bG9sOnNlY3VyZQ==

- HMAC (Hash-based message authorization code)

HMAC stands for Hash-based message authorization code and is a stronger type of authentication, more common in financial APIs. With HMAC, both the sender and receiver know a secret key that no one else does. The sender creates a message based on some system properties (for example, the request timestamp plus account ID).

The message is then encoded by the secret key and passed through a secure hashing algorithm (SHA). (A hash is a scramble of a string based on an algorithm.) The resulting value, referred to as a signature, is placed in the request header.

When the receiver (the API server) receives the request, it takes the same system properties (the request timestamp plus account ID) and uses the secret key (which only the requester and API server know) and SHA to generate the same string. If the string matches the signature in the request header, it accepts the request. If the strings don't match, then the request is rejected.

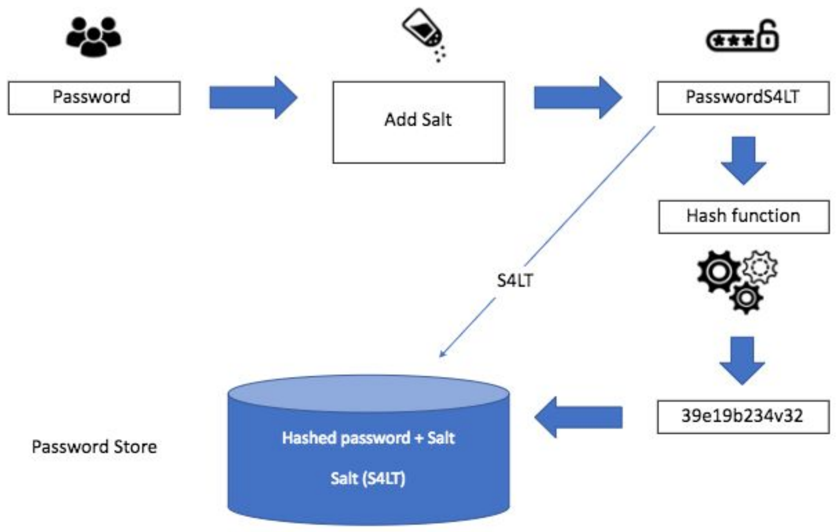

HMAC uses a secret key known only to the client and server to construct a matching signature
AB
Resource server (API)
AB + ==
message
key
signature
App
AB + ==
message
key
signature

- Password Salting

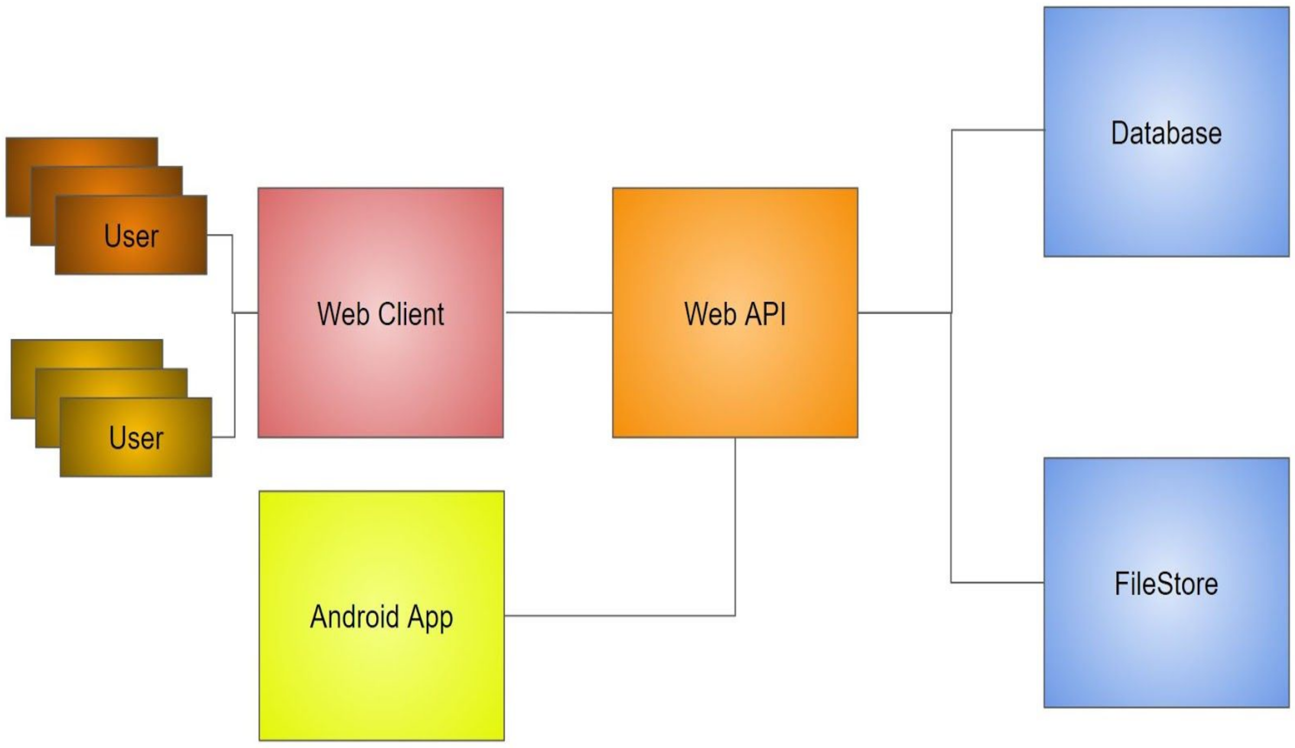
Password hashing is a key step to protecting your users on the backend, but it's not infallible because it hashes in a consistent way. This means it is predictable and can be beaten by dictionary attacks or rainbow table attacks.

"Hello", for example, will always equal the same combination of letters and numbers, and therefore can be guessed through brute force. One way of protecting against this is by adding salt or using salted passwords.

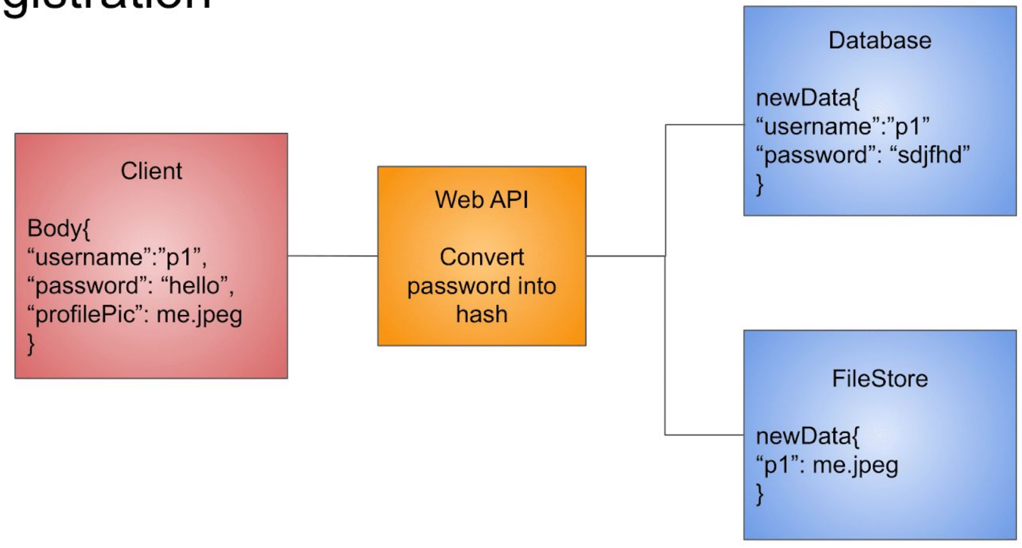
Salting is the act of adding a series of random characters to a password before going through the hashing function. How does it work? Let's take a look:



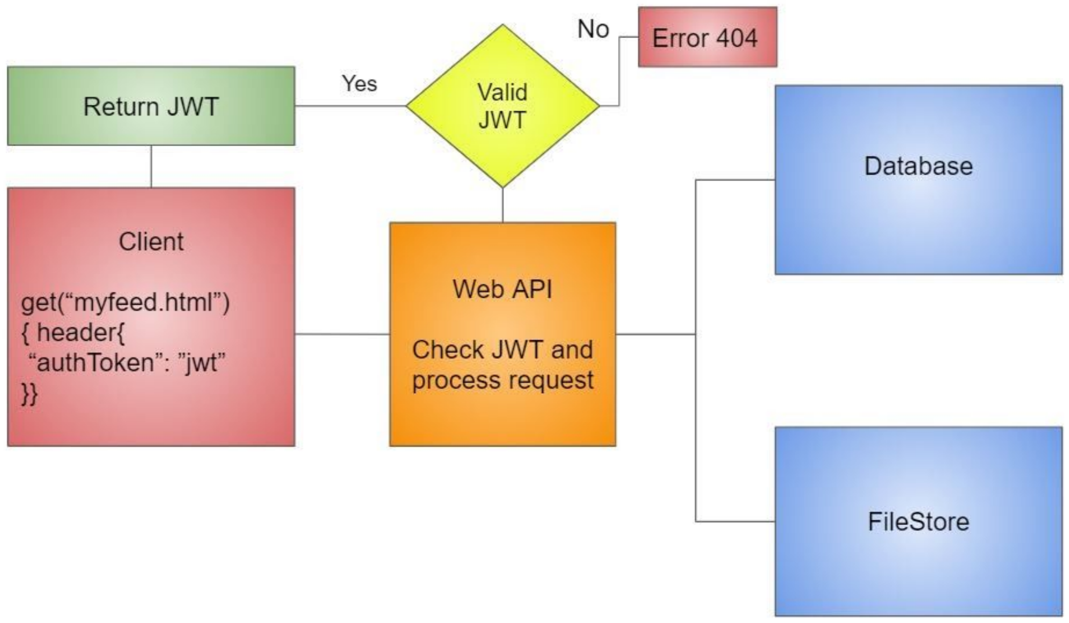
7. Visualizing the System



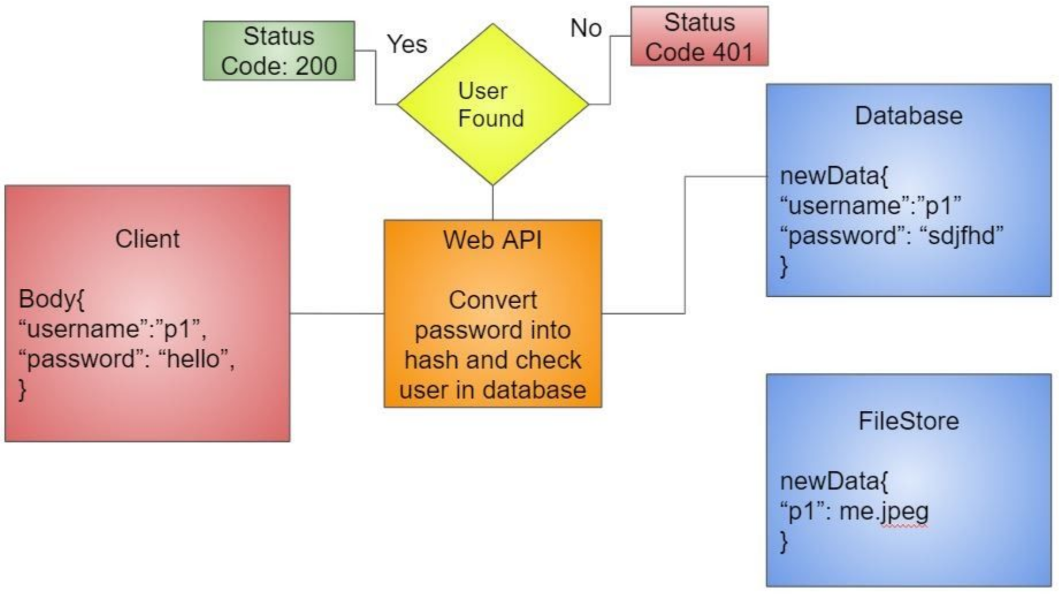
Registration



Authentication



Authorization



8. Testing

8.1 Frontend Testing

Front End Testing is a testing technique in which Graphical User Interface (GUI), functionality and usability of web applications or a software are tested. The goal of Front end testing is testing overall functionalities to ensure the presentation layer of web applications or a software is defect free with successive updates.

For Example: If you enter your name into the frontend of application, numbers should not be accepted. Another example would be checking the alignment of GUI elements.

Apart from this Frontend testing is conducted for:

- CSS Regression Testing: Minor CSS changes that break the frontend layout
- Changes to JS files that make the frontend non-functional
- Performance Checks

8.1.1 <u>How To Create A Frontend Website Testing Plan?</u>

Creating a Frontend testing plan is a simple 4 step process:

Step 1) Find out tools for Managing Your Test Plan

Step 2) Decide the budget for Front End Testing

Step 3) Set the timeline for the entire process

Step 4) Decide the entire scope of the project. The scope includes the following items

- OS and browsers used by users ISP plans of your audience
- Popular devices used by audience
- Proficiency of your audience
- Internet correction speed of the audience

8.1.2 Why Create a Frontend Testing Plan?

A Frontend Testing plan helps you determine

1. Browsers
2. Operating Systems

Your project needs to cover. There are innumerable combinations of Browsers and OS that you could test your front end on. Having a plan will help you reduce the testing effort and money.
By creating frontend testing, plan you will get following advantages-

1. It helps you to get the complete clarity about the scope of the project
2. Performing frontend testing also gives confidence in deploying the project

8.2 Front-End Performance Optimization

Front-end performance testing checks "How fast does page loads."
Optimizing the front-end performance for a single user is a good practice before testing an application with high user loads.

8.2.1 Why Is Front-End Performance Optimization Important?

Earlier performance optimization meant optimizing server-side. That is because most of the websites were mostly static and most of the processing was done on the server side.

However, with the beginning of Web 2.0 technologies, web applications become more dynamic. As a result, client-side code has become a performance hog.

8.2.2 What is the Benefit of Front-End Performance Optimization?

- In website testing, apart from server bottlenecks, finding the client side performance issues are equally important as they easily impact the user's experience.
- Improving back-end performance by 50% will increase the overall performance of the application by 10%.
- However, improving front-end performance by 50% will increase the overall performance of the application by 40%.
- Moreover, front-end performance optimization is easy and cost-effective as compared to the back-end.

8.2.3 Front-end Performance Testing Tools

1. Page Speed

Page speed is an open source performance testing add-on launched by Google. The tool evaluates the web page and provides suggestions to minimize loading time. It makes web page retrieval quicker when users access web pages using Google search engine.

2. YSlow

YSlow is a frontend web performance testing tool. It analyzes web page performance by examining all the components on the page, including components created by using JavaScript. It also measures the page's performance and offers suggestions to the users

8.3 Backend Testing

Software Applications are complex; there is more than what meets the eye.

Most system testing efforts go through GUI. This is because testing validates if the software is 'fit for use' by the end-user or not. End-users use GUI and so do we; that is why it is really important that software fares well in this area.

But, software has a lot of other elements too that aren't directly visible or available to the user for direct interaction. It does not make these elements any less important and they must too undergo thorough testing.

The combination of all these well-functioning elements makes a fully formed software application. We can combine everything we do not directly see as 'Back-end'.

Some of the Backend Testing elements are:

- Database
- APIs
- Servers

Depending on the nature of the application a back-end can include various network configurations, communication protocols, etc. But most often, there are three elements

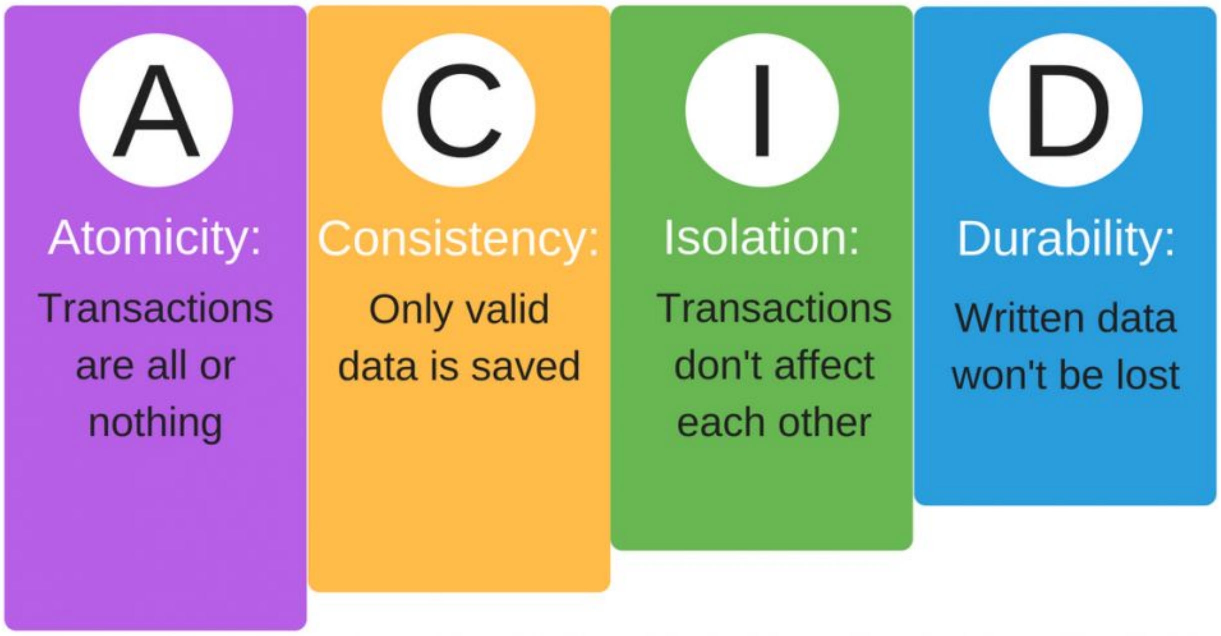
8.3.1 Database Testing

Most commonly when the term ‘Back End Testing’ is used, it implies Database testing.

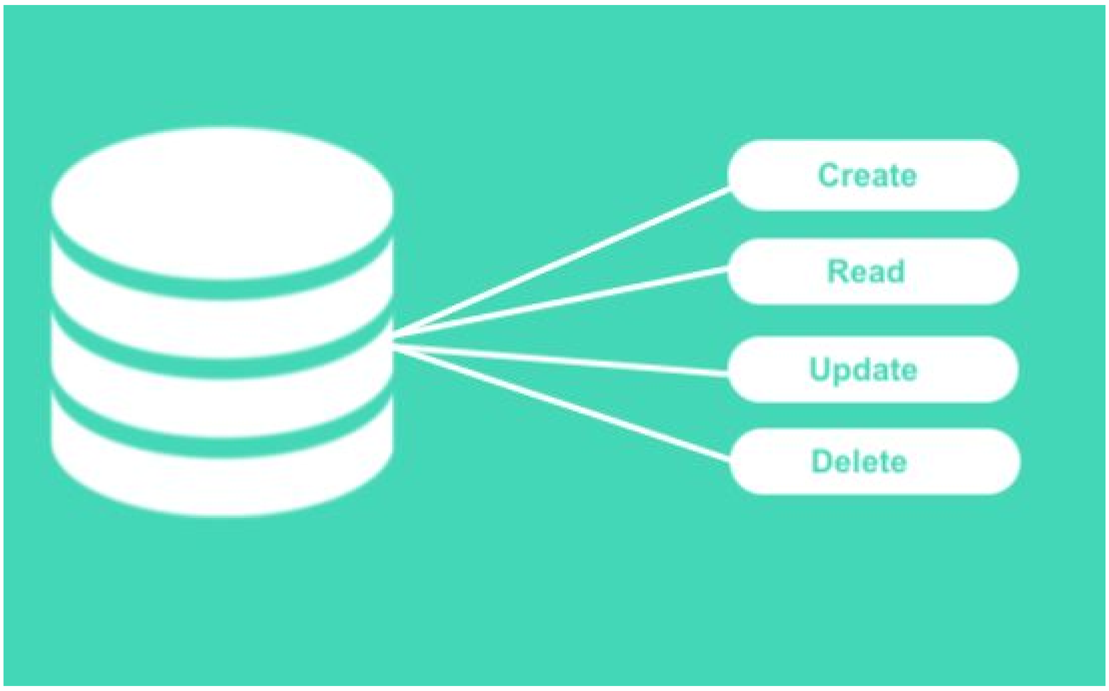
The database is an important element of any application. When the GUI and the DB interact with one another seamlessly your application works well. If there are problems, you experience inconsistent results, security threats and performance bottlenecks.

Databases are usually validated for:

1. ACID properties


A
Atomicity:
Transactions are all or nothing
C
Consistency:
Only valid data is saved
I
Isolation:
Transactions don't affect each other
D
Durability:
Written data won't be lost

2. CRUD operations



3. Schema
4. Migration
5. Business rule conformance
6. Security
7. Performance

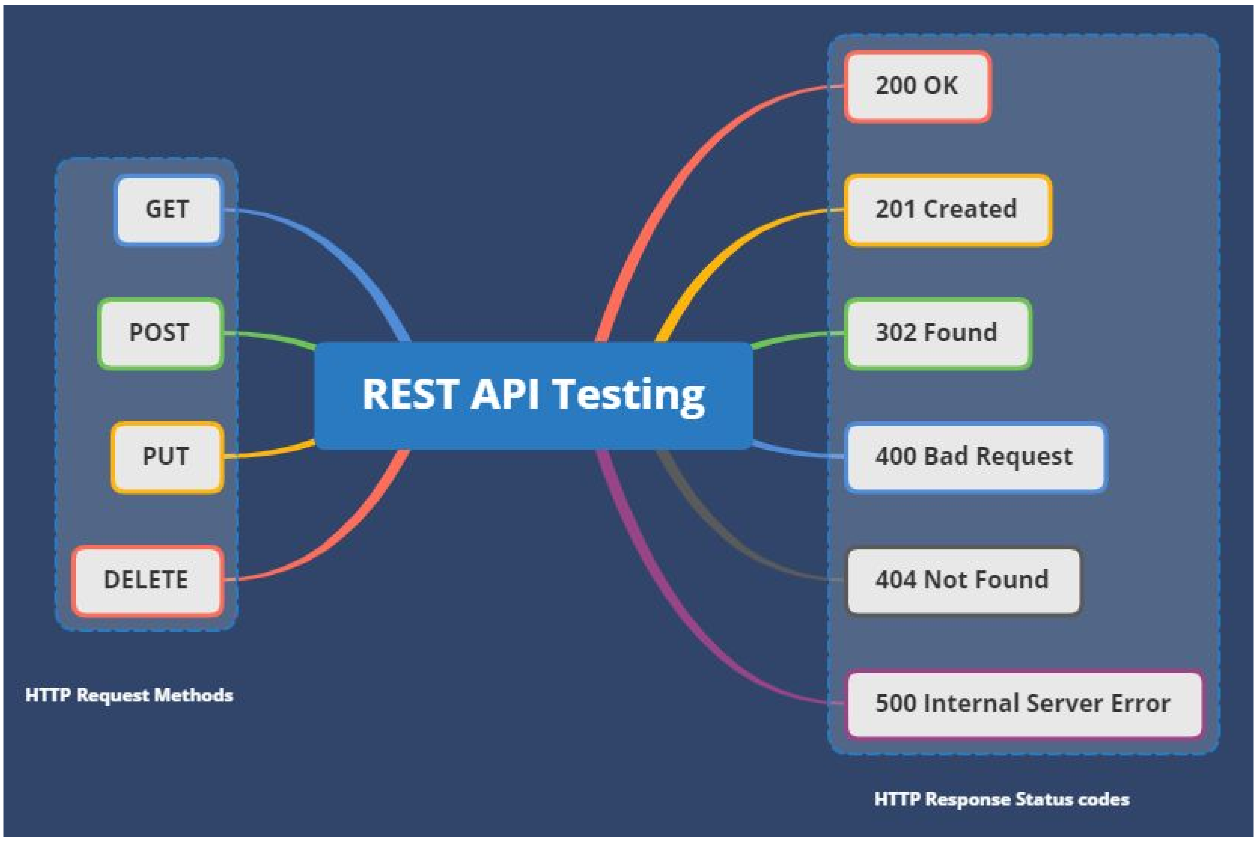
8.3.2 API testing

API is strictly speaking not the back-end but since we are loosely grouping everything that is not visible to the end-user as the back-end, let's talk about this briefly too.

API stands for Application Program Interface and this is basically where all the programming logic resides. It does not have a UI which is one of the biggest challenges when it comes to testing it. On the other hand, since APIs are generally created before the application's UI comes into existence, testing the API usually means early testing.

Messaging and send/receive calls are used instead of direct send and receiving of input and output data.

The most popular tool used for API testing is *POSTMAN*



.

8.5 Android Testing

Android Studio is designed to make testing simple. With just a few clicks, you can set up a JUnit test that runs on the local JVM or an instrumented test that runs on a device.We can also extend our test capabilities by integrating test frameworks such as 'Mockito' to test Android API calls in your 'local unit tests', and 'Espresso' or 'UI Automator' to exercise user interaction in your 'instrumented tests'. You can generate Espresso tests automatically using 'Espresso Test Recorder'.

8.5.1 Local Test Units

Located at **module-name**/src/test/java/.

These are tests that run on your machine's local Java Virtual Machine (JVM). Use these tests to minimize execution time when your tests have no Android framework dependencies or when you can mock the Android framework dependencies.

At runtime, these tests are executed against a modified version of android.jar where all final modifiers have been stripped off. This lets you use popular mocking libraries, like Mockito.

8.5.2 Instrumented test

Located at **module-name**/src/androidTest/java/

These are tests that run on a hardware device or emulator. These tests have access to Instrumentation APIs, give you access to information such as the Context of the app you are testing, and let you control the app under test from your test code. Use these tests when writing integration and functional UI tests to automate user interaction, or when your tests have Android dependencies that mock objects cannot satisfy.

Because instrumented tests are built into an APK (separate from your app APK), they must have their own AndroidManifest.xml file. However, Gradle automatically generates this file during the build so it is not visible in your project source set. You can add your own manifest file if necessary, such as to specify a different value for `minSdkVersion` or register run listeners just for your tests. When building your app, Gradle merges multiple manifest files into one manifest.

9. Tools

9.1 Git and GitHub

All developers will use some kind of version control system (VCS), a tool to allow them to collaborate with other developers on a project without danger of them overwriting each other's work, and roll back to previous versions of the code base if a problem is discovered later on. The most popular VCS (at least among web developers) is Git, along with GitHub, a site that provides hosting for your repositories and several tools for working with them. This module aims to teach you what you need to know about both of them.

9.2 Postman

Postman is a popular API client that makes it easy for developers to create, share, test and document APIs. This is done by allowing users to create and save simple and complex HTTP/s requests, as well as read their responses. The result - more efficient and less tedious work.

9.3 Chrome DevTools

Chrome DevTools is a set of web developer tools built directly into the Google Chrome browser. DevTools can help you edit pages on-the-fly and diagnose problems quickly, which ultimately helps you build better websites, faster.

9.4 NPM

npm is the package manager for the Node JavaScript platform. It puts modules in place so that node can find them, and manages dependency conflicts intelligently. ... Most commonly, it is **used** to publish, discover, install, and develop node programs.

9.5 Android Studio

Android Studio is the official Integrated Development Environment (IDE) for android application development. Android Studio provides more features that enhance our productivity while building Android apps.

10. Conclusion

It feels frustrating and infuriating when a few lines of code makes you stop coding ? Isn't so. Well if this is the way for experienced coders think about the newbies who when trying new algorithms get stuck and are not able to get solutions. Many might argue go on Google for this but for beginners it is overwhelming to go through all the stuff.

For tackling this problem, our platform will provide you with active coders who prefer the same language as the user does. This will not only help in getting the solutions for as small as a doubt the user has. This way every doubt will be solved.

11. Future Scope

Scope for extension into a major project:

- Providing a separate portal for hosting coding competitions which will also be accessible by GLA University for the same purpose.
- Implementing “File Search” feature for quick searching of particular files in large and complex structures.
- A PC version of soPro.
- Online code editor so the bug fix process just look like “*Upload problem->Someone fix the bug -> Download bug fixed file and replace it with original* ”

12. References

12.1 Android

1. https://firebase.google.com/docs/
2. https://firebase.google.com/docs/android/
3. https://material.io/
4. https://www.postman.com/
5. https://developer.android.com/studio
6. https://github.com/

12.2 Web

1. https://restfulapi.net/
2. https://www.npmjs.com/
3. https://nodejs.org/en/
4. https://mongoosejs.com/
5. https://www.mongodb.com/
6. https://firebase.google.com/
7. https://developer.mozilla.org/en-US/
8. https://microservices.io/
9. https://github.com/
   https://developers.google.com/web/tools/chrome-devtools/console