**Full Stack Development**

**Capgemini**

**A training report**

Submitted in partial fulfillment of the requirements for the award of degree of

**Bachelor of Technology**

**Electronics and Communication Engineering**

**LOVELY PROFESSIONAL UNIVERSITY**

**PHAGWARA, PUNJAB**



**From 01/20/2020 to 05/08/2020**

**Name of student: Dande Sai Sumanth**

**Registration Number: 11613098**

**Signature of the student: D. Sai Sumanth**

**Student Declaration**

**To whom so ever it may concern**

I, **Dande Sai Sumanth**, **11613098** of the student, hereby declare that the work done by me on “**Full Stack Development**” from **January, 2020** to **May, 2020**, under the supervision of **T J Govindarajulu**, **Senior Java Architect and Trainer**, **Capgemini** and **Saval Tandon**, **Head of Department**, **Lovely Professional University**, Phagwara, Punjab is a record of original work for the partial fulfilment of the requirements for the award of degree, **B. Tech**

Name of the Student (Registration Number)

Dande Sai Sumanth (11613098)

Signature of the student

**Declaration by the supervisors**

**To whom so ever it may concern**

This is to certify that **Dande Sai Sumanth**, **11613098** from **Lovely Professional University**, Phagwara, Punjab, has worked as a trainee in **Capgemini** on “Full Stack Development” under my supervision from **Jan 2020 to May 2020**. It is further stated that the work carried out by the student is a record of original work to the best of my knowledge for the partial fulfilment of the requirements for the award of the degree, degree name.

Name of External Supervisor Name of Internal Supervisor

T J Govindarajulu Sawal Tandon

Designation of the External Supervisor Designation of the Internal Supervisor

Senior Java Architect and Trainer Head of Department

Signature of the external Supervisor Signature of the Internal Supervisor

**ACKNOWLEDGEMENT**

The satisfaction that accompanies that the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success.

We are grateful to our project guide Mr.T J Govinadrajulu and Mrs. Varsha Lonkar ma’am for the guidance, inspiration and constructive suggestions that helpful us in the preparation of this project.

We also thank our colleagues who have helped in successful completion of the project.

Dande Sai Sumanth

11613098

**1. INTRODUCTION OF THE COMPANY**

Capgemini SE is a French multinational corporation that provides consulting, technology, professional, and outsourcing services. It is headquartered in Paris, France. Capgemini has over 270,000 employees in over 50 countries, of whom nearly 120,000 are in India.

**Origin**:

Capgemini was founded by Serge Kampf in 1967 as an enterprise management and data processing company. The company was inaugurated as Sogeti.

Serge Kampf was a visionary and demanding rigorous leader, but he was also reserved, loyal, generous and considerate. Serge was an exceptional man. He was captivating and left no one indifferent. A leader who understood the evolution of our business from quite early on, his life was an extraordinary journey. Over nearly fifty years, he built the company from a two-room apartment in his hometown of Grenoble to being one of the global leaders in the ultra-competitive world of IT services.

Serge Kampf inspired the dreams of generations of IT professionals. He instilled the Group with the drive to win with his genius and entrepreneurial passion. Capgemini has completed over 40 acquisitions and welcomed over 50 countries and cultures because of the organizational model invented by Serge Kampf. As decentralized as possible, this model transcends differences and respects everyone involved in this shared adventure. Throughout the 49 years that Serge dedicated to Capgemini, he also demonstrated that it was possible to build one of the largest companies in France and expand it across the world, with only a few key values serving as his compass. Serge understood that a Group is only as strong as the resolute commitment of its employees. Therefore, he emphasized these same values in the men and women who worked at his side. Serge was also the perfect representation of what friendship could be. He was a friend to his colleagues and a friend to the world of rugby. Ultimately, he showed that you could create a global leader in business based on the principle of friendship.

In 1974 Sogeti acquired Gemini Computers Systems, a US company based in New York. In 1975, having made two major acquisitions of CAP and Gemini Computer Systems, and following resolution of a dispute with the similarly-named CAP UK over the international use of the name 'CAP', Sogeti renamed itself as CAP Gemini Sogeti.

Cap Gemini Sogeti launched US operations in 1981, following the acquisition of Milwaukee-based DASD Corporation, specializing in data conversion and employing 500 people in 20 branches throughout the US. Following this acquisition, The U.S. Operation was known as Cap Gemini DASD.

In 1996, the name was simplified to Cap Gemini with a new group logo. All operating companies worldwide were re-branded to operate as Cap Gemini.

Ernst & Young Consulting was acquired by Cap Gemini in 2000. It simultaneously integrated Gemini Consulting to form Cap Gemini Ernst & Young.

In 2017, Cap Gemini S.A. became Capgemini SE, and its Euronext ticker name similarly changed from CAP GEMINI to CAPGEMINI.

In 2019, Capgemini acquires Altran bringing the total employee count to over 250,000. This is the largest acquisition in the company’s history.

The Capgemini Group Executive Committee consists of 26 members. On 20 May 2020, Aiman Ezzat appointed as the new CEO. He is associated with Capgemini for more than 20 years. From 2005 to 2007, Aiman was Capgemini’s Deputy Director of Strategy. In November 2007, Ezzat was appointed COO of the Financial Services Global Business Unit, and became its Global Head in December 2008 till 2012. From January 2018 to May 2020, he served as Chief Operating Officer and prior to this as Chief Financial Officer, from December 2012 to 2018.

From 2012 to 2020 Paul Hermelin served as the Group Chairman and CEO. He joined Capgemini in 1993 and was appointed as its CEO in 2002. In May 2012, Hermelin became chairman and CEO of the Capgemini Group. He succeeded Serge Kampf, who served as the Vice Chairman of the Board until his death on 15 March 2016.

**Services Provided by the Company:**

Services provided by Capgemini are as follows:

* Outsourcing: Outsourcing is an agreement in which one company hires another company to be responsible for a planned or existing activity that is or could be done internally, and sometimes involves transferring employees and assets from one firm to another.
* Consulting: A consultant is a professional who provides expert advice in a particular area such as business, education, law, regulatory compliance, human resources, marketing, finance, health care, engineering, science, security, or any of many other specialized fields.
* Managed services: Managed services is the practice of outsourcing the responsibility for maintaining, and anticipating need for, a range of processes and functions in order to improve operations and cut expenses. It is an alternative to the break/fix or on-demand outsourcing model where the service provider performs on-demand services and bills the customer only for the work done.

**Revenue, Income, Assets of the Company:**

The company’s revenue is €14.12 billion (2019) with the operating income of €1.43 billion (2019) and the Net income of €856 million (2019).

The Total assets are €18.13 billion (2019) and the Total equity is €8.41 billion (2019) and the number of employees is 270,000 (2020).

**Capgemini Facilities Available:**

Irrespective of being fresher/experienced you will get following

* 5-day Free Bus Pass
* 1-day lunch coupon
* Salary bank account (depending on bank selection you get different offers, HDFC is the most preferred one)
* Access to library
* Access to Capgemini University portal (Skill development portal)
* If goodies excite you then depending on your project/client you might get some
* Courier service (Paid)

**Company’s Vision, Mission and Values:**

**Capgemini Vision: the business value of technology comes from and through people**

Capgemini understands that business value cannot be achieved through technology alone. It starts with people: experts working together to get to the heart of your individual business objectives and develop the most adapted solutions to fit these requirements. We believe this human-centered approach to technology is what makes the difference for your business.

**Capgemini Mission: with you, we create and deliver business and technology solutions that fit your needs and drive the results you want**

Capgemini enables you to transform your organization and improve performance. We aim to empower you to respond more quickly and intuitively to changing market dynamics. By bolstering your ability to harness the right technology, we help you become more agile and competitive.

Collaboration is central to the way we do business. Our experts join forces with your people to form a cohesive team. More than just a promise, our capacity to collaborate has become a key client expectation. We call this approach the Collaborative Business Experience. It shows in our every interaction and is our way of forging closer, more effective relationships.

People matter, results count.

**Capgemini Values:**

Capgemini’s Values and ethics are part of what makes working at Capgemini different. They empower people to master their business and technology domains, build meaningful relationships with their colleagues and clients, and design positive futures.

Since the creation of the Group in 1967, culture and business practices have been inspired by seven core Values. These are the guiding principles that company collectively and individually stand for, and they are at the heart of our approach as an ethical and responsible business.

Capgemini’s Values are not only rules of behaviour, but also guiding principles. Embedded in DNA, they shape the ethical culture, producing a shared mind-set that keeps ethics at the heart of company’s decisions and actions.

Company believe that sound ethics and integrity are the foundation of a profitable, sustainable business. The Ethics & Compliance (E&C) program, with the Values at its core, is crucial for the success of a diverse, decentralized Group, with multicultural teams operating in more than 40 countries. It has helped become one of the most trusted and respected companies in the world, recognized as “One of the World’s Most Ethical Companies” by the Ethisphere Institute seven years in a row since 2013.

Company’s ethics program aims to create awareness among employees about ethical principles and policies, which work as a guiding force, enabling team members to make better decisions, based on honesty and integrity. Our compliance program is focused on ensuring that appropriate levels of control are in place, so that external and internal regulations are adhered to.

**Seven Values:**

Seven shared values have been at the heart of Capgemini since our formation. These values influence the way the company meet client needs while respecting the regulatory requirements of each country in which the company operate, and the way it promotes ethically sound practices within Capgemini and in their partnerships.

The seven values are:

HONESTY, loyalty, integrity, uprightness, a complete refusal to use any underhanded method to help win business or gain any kind of advantage. Neither growth, nor profit nor independence have any real worth unless they are won through complete honesty and probity. And everyone in the Group knows that any lack of openness and integrity in our business dealings will be penalized at once.



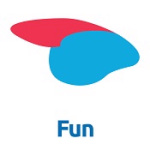
BOLDNESS, which implies a flair for entrepreneurship, and a desire to take considered risks and show commitment (naturally linked to a firm determination to uphold one’s commitments). This is the very soul of competitiveness: firmness in making decisions or in forcing their implementation, an acceptance periodically to challenge one’s orientations and the status quo. Boldness also needs to be combined with a certain level of prudence and a particular clear sightedness, without which a bold manager is, in reality, merely dangerously reckless.



TRUST, meaning the willingness to empower both individuals and teams; to have decisions made as close as possible to the point where they will be put into practice. Trust also means giving priority, within the company, to real openness toward other people and the widest possible sharing of ideas and information.



FREEDOM, which means independence in thought, judgment and deeds, and entrepreneurial spirit, creativity. It also means tolerance, respect for others, for different cultures and customs: an essential quality in a Group of over 200,000 people of around 120 different nationalities.



FUN, means feeling good about being part of the company or one’s team, feeling proud of what one does, feeling a sense of accomplishment in the search for better quality and greater efficiency, feeling part of a challenging project.

MODESTY, that is simplicity, the very opposite of affectation, pretension, pomposity, arrogance and boastfulness. Simplicity does not imply naivety (simple does not mean simpleton!); it is more about being discreet, showing natural modesty, common sense, being attentive to others and taking the trouble to be understood by them. It is about being frank in work relationships, loosening up, having a sense of humor.



TEAM SPIRIT, meaning solidarity, friendship, fidelity, generosity, fairness in sharing the benefits of collective work; accepting responsibilities and an instinctive willingness to support common efforts when the storm is raging.

**Capgemini Brand:**

Three fundamental and differentiating points inspired company’s new brand identity that reflects what makes the company unique; strengths as a business:

Dynamism**:** because the company are adaptive, agile, always in motion, constantly learning and adapting. The Company are open to and constantly adapt and master the latest innovations, the future in technology and business.

Precision**:** The company focus means that accuracy is vital for us and for clients – smart, sharp, crisp and clean in a complex world – simply said, trusted and reliable. Precise in grasping client challenges and precise in delivering them the right solution based on our commitments. It is this promise of precision that influences the company’s visual thinking.

People**:** The Company succeed because they work collaboratively, because they take time to understand what clients need. Some do it for them, some do it to them, The Company take implying that not just our technological expertise but also sense of humanity and integrity are palpable and now reflected in how the Company look.

**Organisation Tree:**

Like most other companies of such a stature, Capgemini also have a hierarchical inner structure. From the level of the lowest employee to the highest level a pyramid like structure exists, and therefore in order to reach at a specific level, you need to work for sometimes in a lower section. The hierarchical structure of the company is explained below.



Fig1.1

At the highest level of a career in Capgemini hierarchy, there is the Vice president, followed by the principal, the managing consultant, and others, with the analyst at the lowest level of the hierarchy.

* **Vice President-** At the Capgemini Consultancy Company, the Vice President is the topmost leader. There can be more than one vice president at a time, providing leadership to the company. It is their duty to guide the company towards the right channels and avenues. In order to become a Vice president in the Capgemini company, only skill and talent is not sufficient, but years of experience and a good track record is also needed.
* **Principal-** The immediate second rank to the vice president in the company belongs to the Principal. He or she is also an important leader of business. It is the duty of the principle to produce projects of high impact and high levels and thereby forming food business relations with important companies. It is also the responsibility of the principal to create teams which are properly qualified to deal with specific projects.
* **Managing consultant-** Next in the hierarchical structure of the company is the managing consultant. In the company, this position is respected by all as it is a very important one. The managing consultant has a number of duties such as, proving leadership to a team of consultants, increasing the volume of business with the help of add on sales, making sure that the projects are delivered at the right time, and managing other important aspects of the business.
* **Senior Consultant-** The next person in the company hierarchy is the senior consultant. In order to become a senior consultant one needs years of experience along with specific skills. A senior consultant can work on his own or with a team, depending on the demands of the project. At times they head small teams and sees to it that deadlines are met. He or she also deals with the client.
* **Consultant-** Under the guidance of a senior consultant works one or more consultants. They work in a team using their skills and knowledge and often have to understand the requirement of the client and work accordingly.
* **Analyst-** An analyst stands at the lowest tier of the company hierarchy. In some places this position does not exists, in most others they are supervised by consultants and senior consultants.

**2. INTRODUCTION OF THE PROJECT UNDERTAKEN**

**Project Name: Online Payment Wallet**

**Objective:** Online Payment Wallet is a website to provide the customers the facility to create a wallet account and add amount to that wallet and transfer fund from one wallet account to another wallet account and the user can see the balance whenever he wants and the user can get the list of transactions he made to particular wallet account or to get transactions from particular date to particular date.

**Functionalities of Project:**

1. Creating a wallet account
2. Login using wallet account details
3. Show Balance of the user
4. Adding amount to the wallet account
5. Fund transfer
6. Download Transactions
7. View Transactions
   1. View Transactions in particular date range
   2. View Transactions sent to a particular wallet user
   3. View Transactions received from particular wallet user
   4. View Transactions sent to a particular wallet user in date range
   5. View Transactions received from particular wallet user in date range
   6. View last six months transactions.

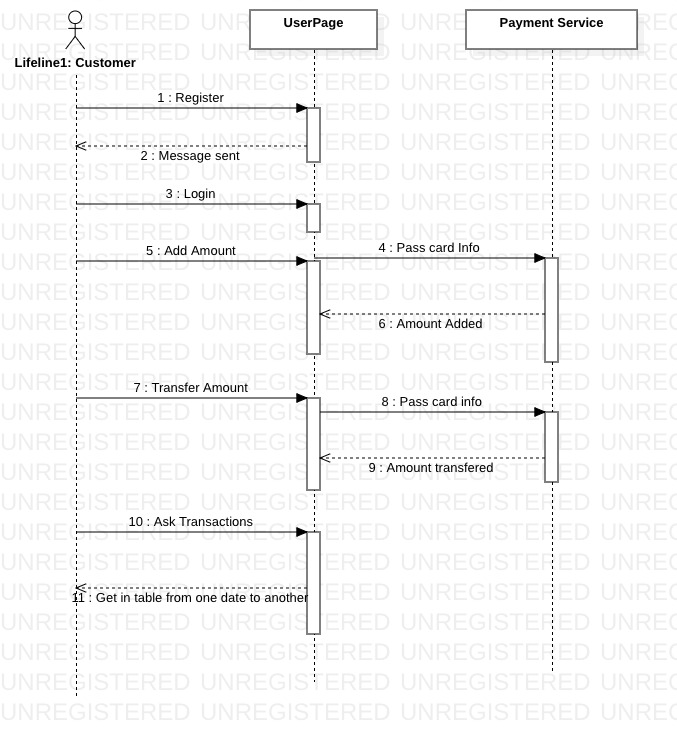
**Work Plan:**

During our project we have divided our time into three sprints and each sprint is of 6 days. In first sprint we have coded the backend and connection to database using spring and populating test data in database. In second sprint we have designed the frontend using HTML, CSS and Bootstrap and Angular 7. In sprint3 we have coded the extra like we have implemented login page, AUTH guard and download option for transactions. Our team consist of four members and divided the project among ourselves. Each one of us took 2 functionalities. And we took 2 extra days for integrating the whole projects as project was coded by the whole team and we have uploaded the project on GitHub.

**Role and profile:**

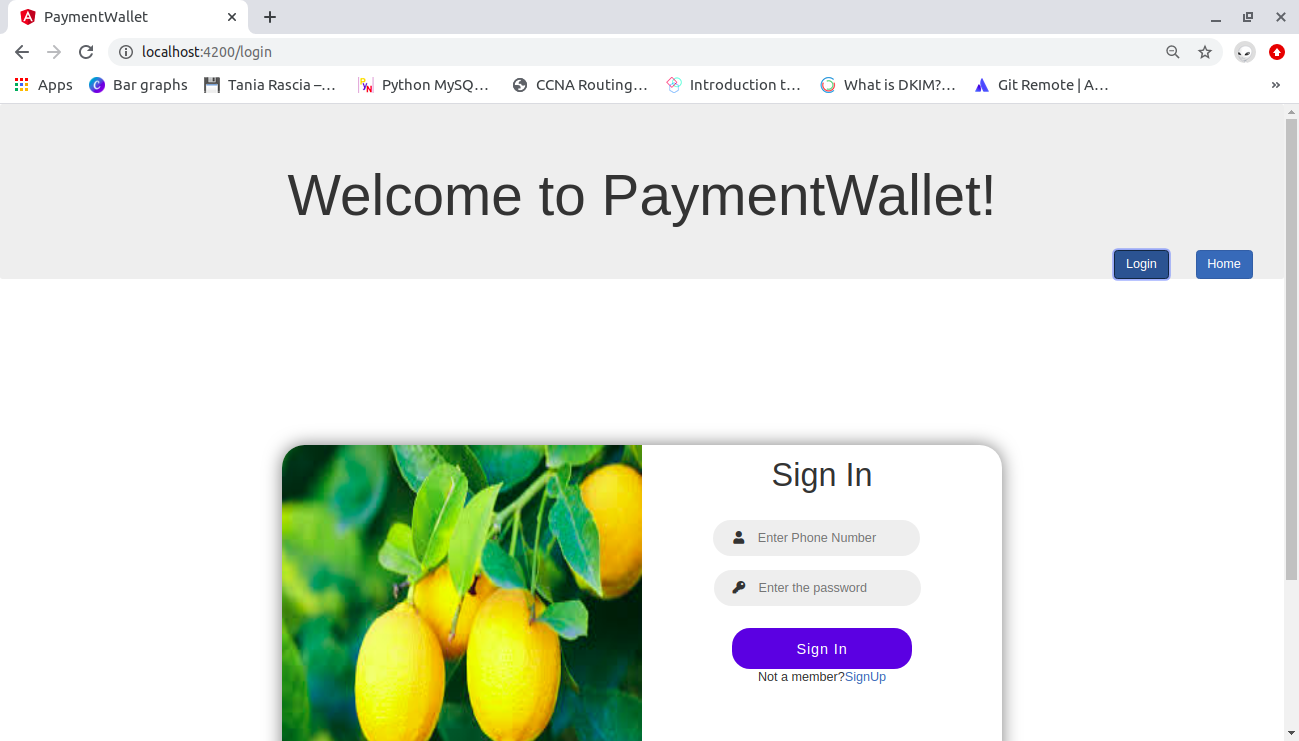
In this project my role is scrum master that is to conduct the scrum to know what the team members have done yesterday and what he is going to do today. I conducted the scrum meeting every day to make sure that there is no backlog in work and no team member is facing any problem.

**Project Flow:**



**Implementation of Project:**

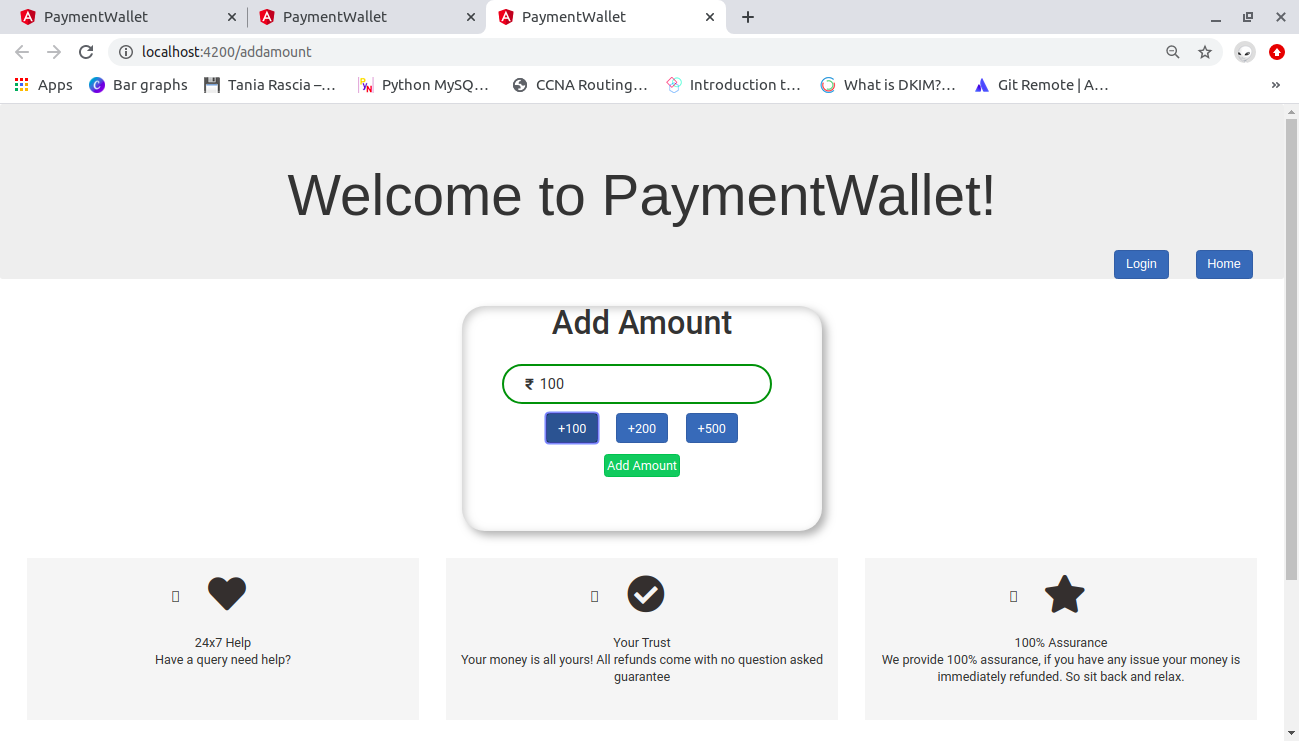
First the user will be displayed home page



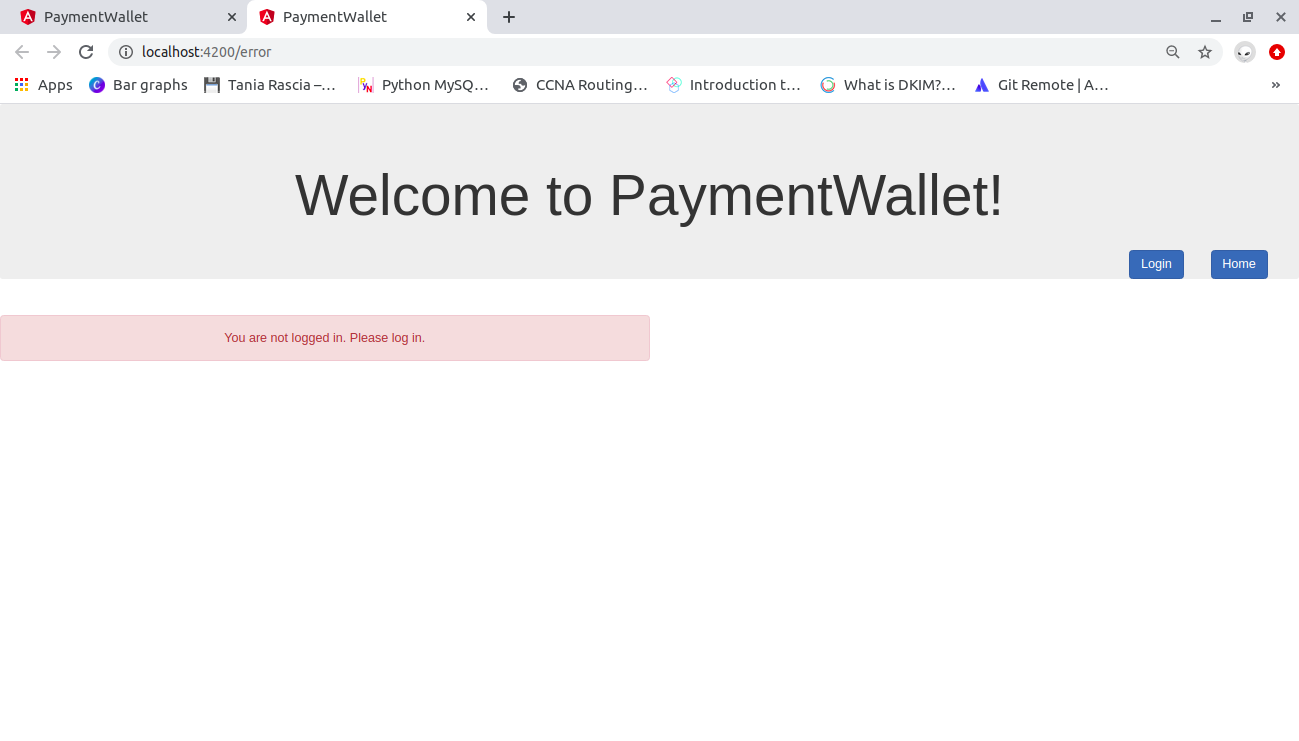
If the user enter correct login details the page will be redirected to account summary page and from home page the user can select any functionality

In this project I have done add amount to wallet account and show balance functionality

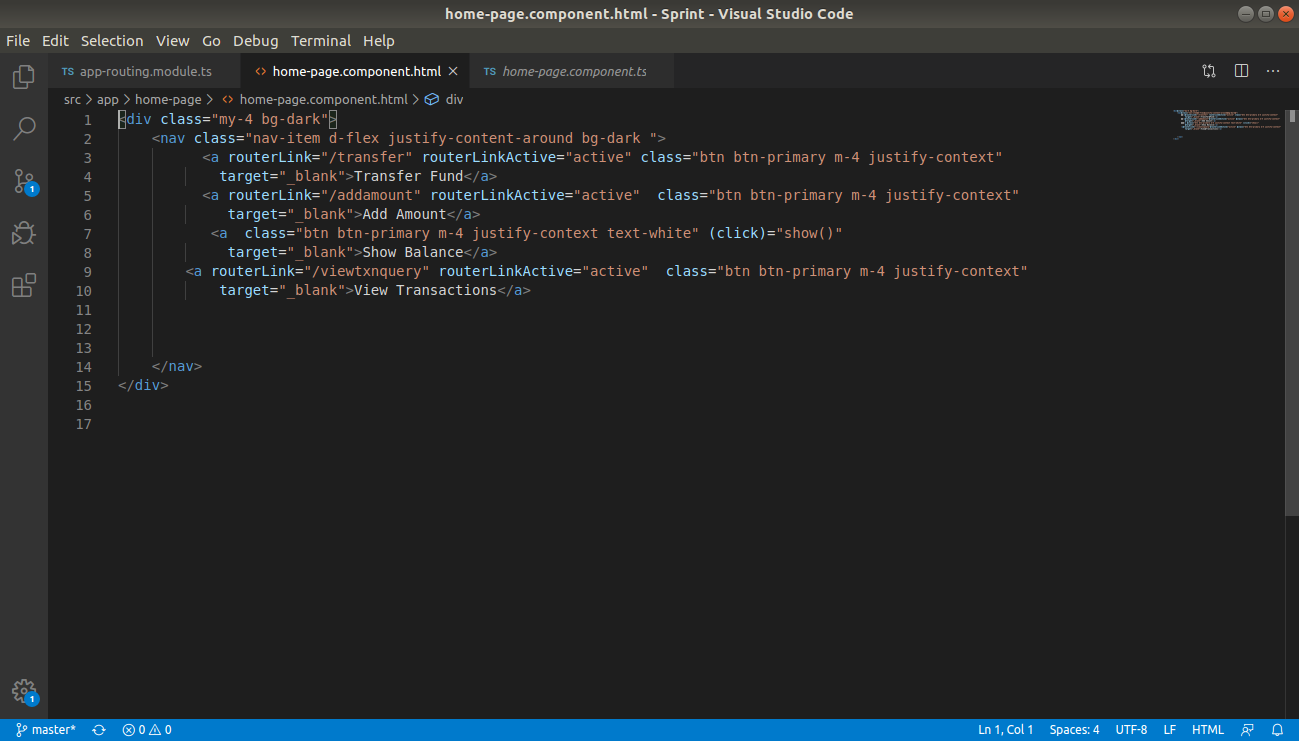
If the user clicks on add amount button the page will be redirected to add amount page.



And if the user clicks any functionality button without sign in an error page will be displayed. And this functionality is called AUTH guard.

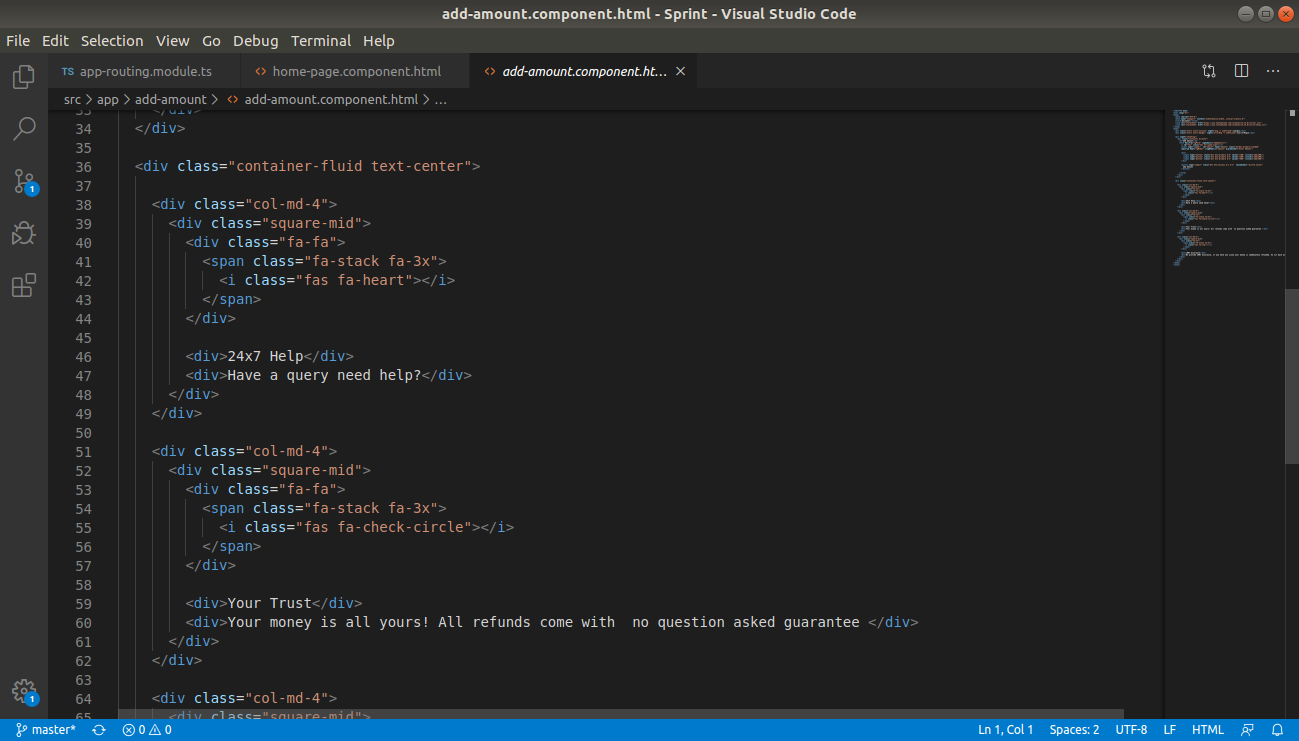
That is the frontend part of the project.

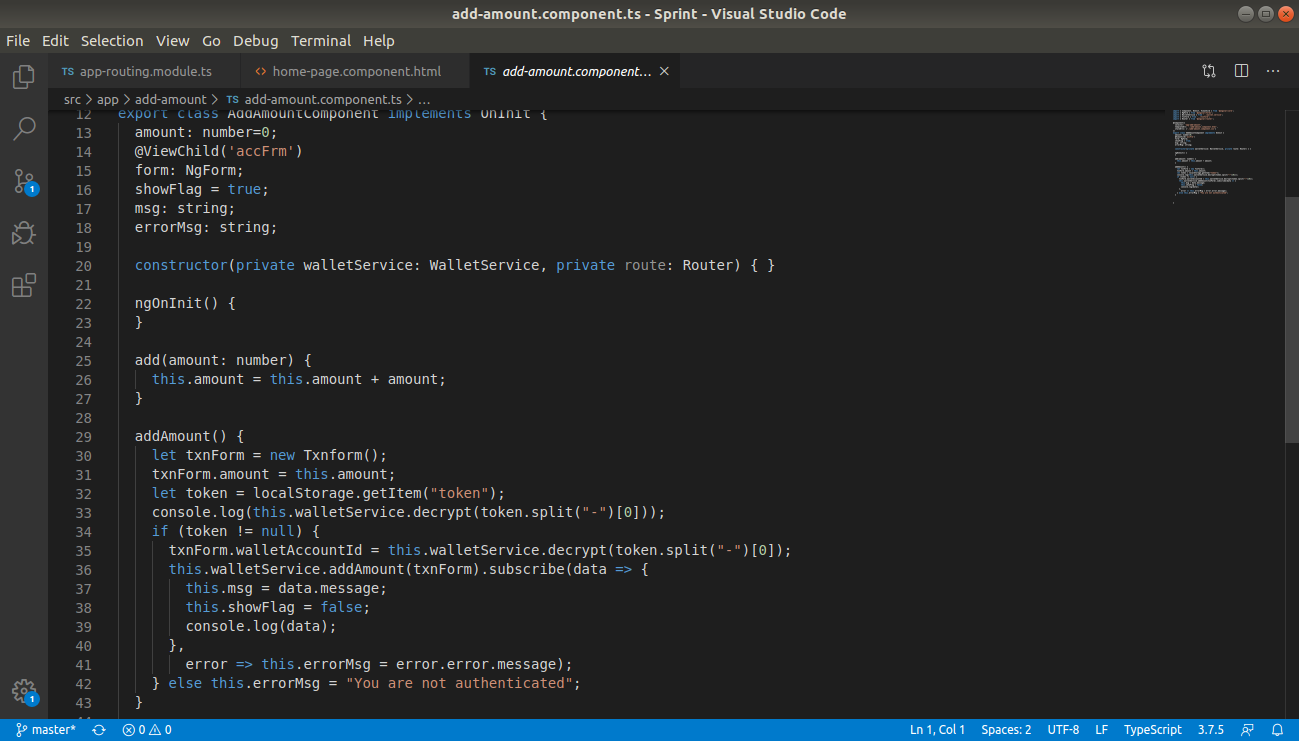
And the frontend code starts from homepage

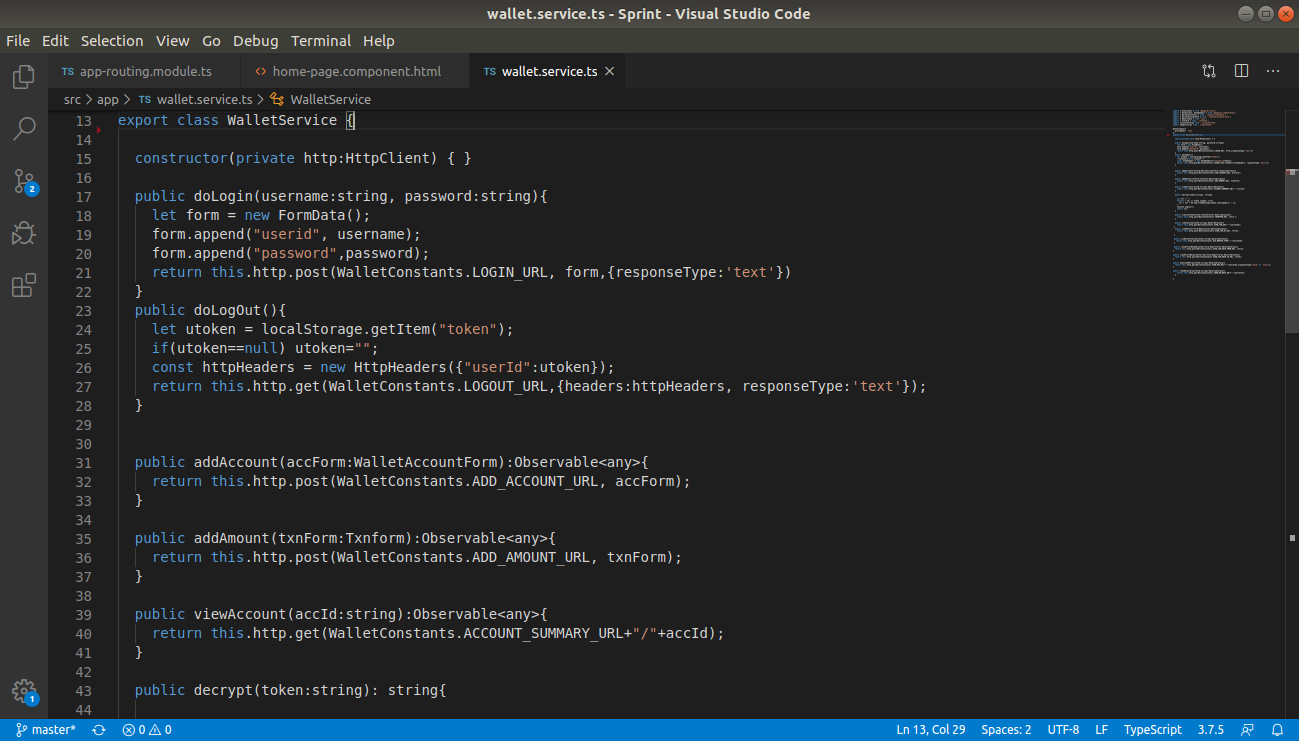


From home page it will be redireccted to any component.

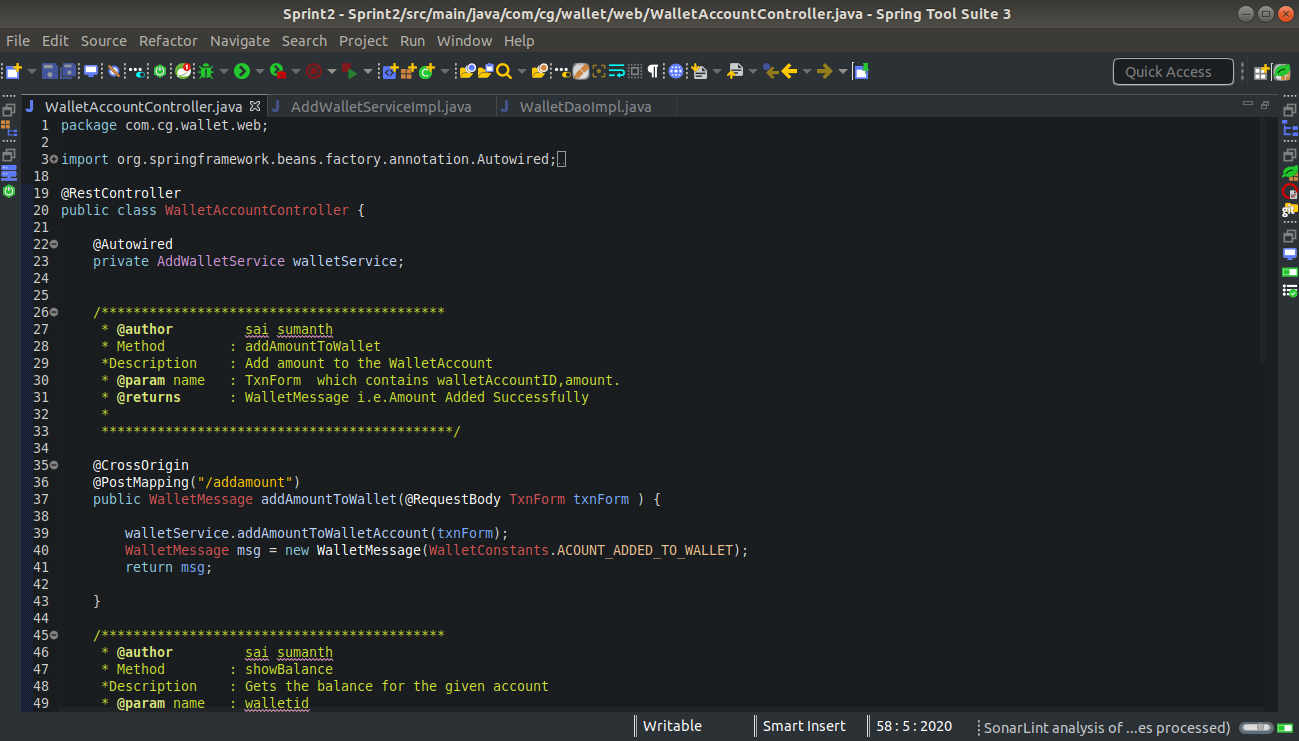
If I click add amount button it will be redirected to add amount page and the code of add amount page is:

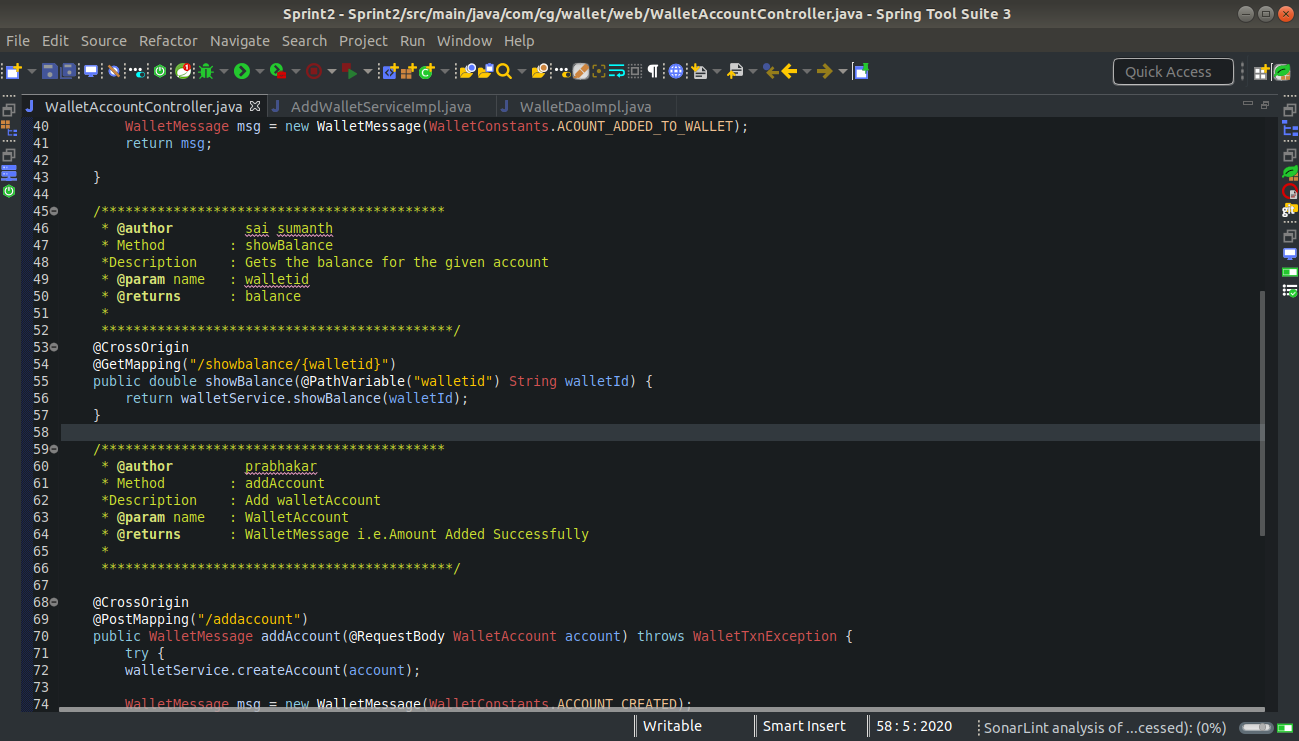


From add amount component it will go to service

 And from the service http request will be made to backend and required operations take place in backend.

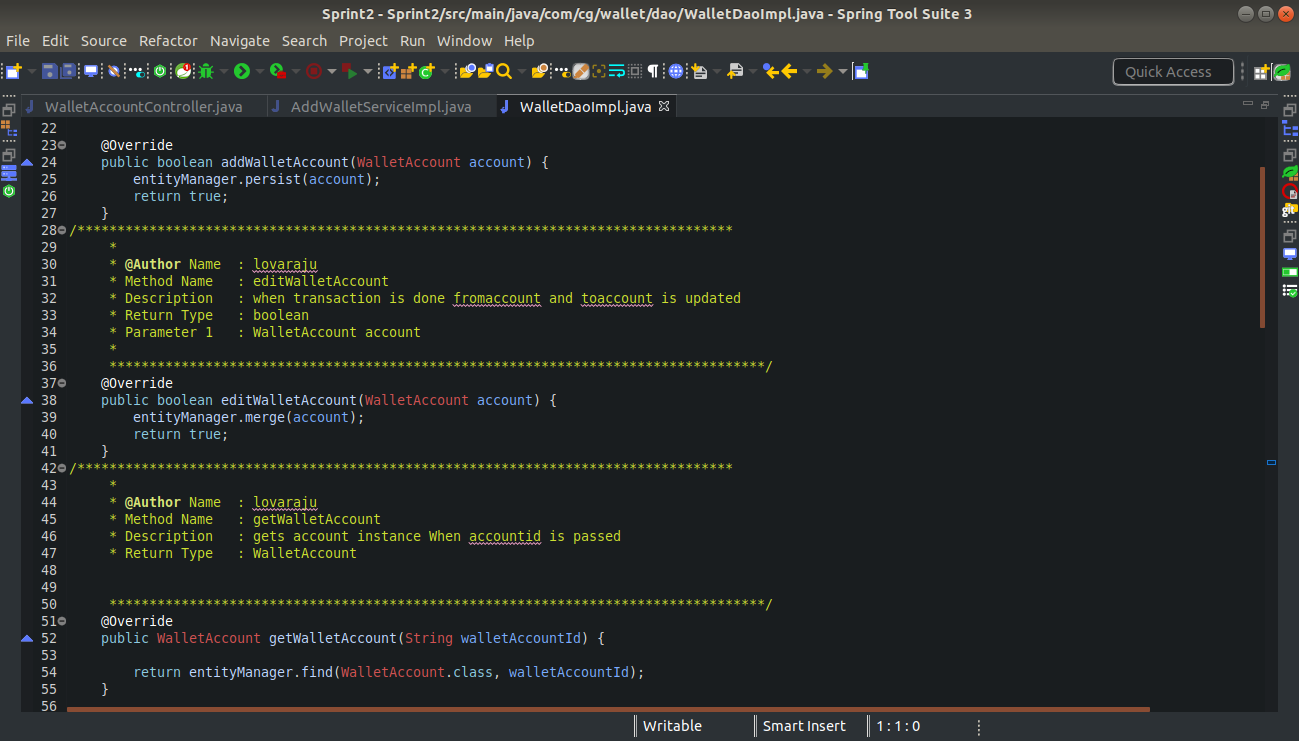
From frontend the request will be received by the restcontroller in backend.



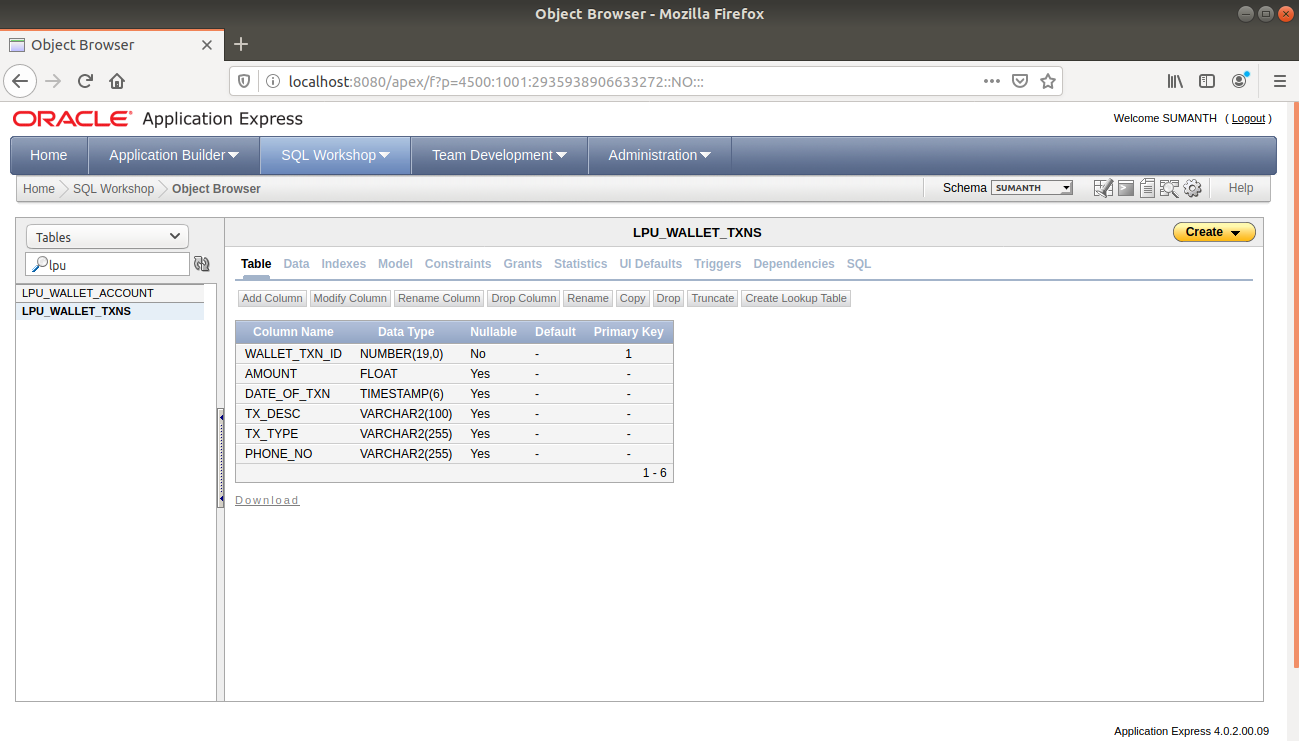


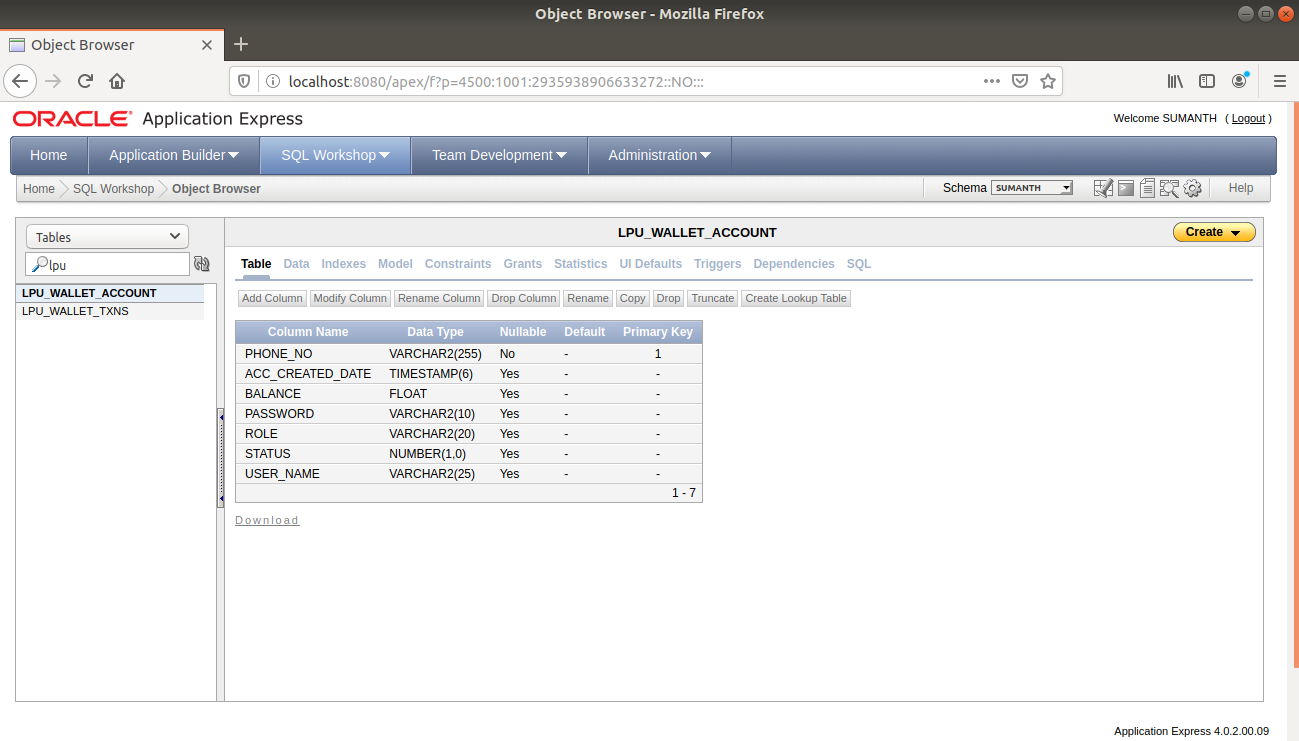
From rest controller the request will be sent to service where business logic code is written

And the service will make a call to Dao which connects it to the database.



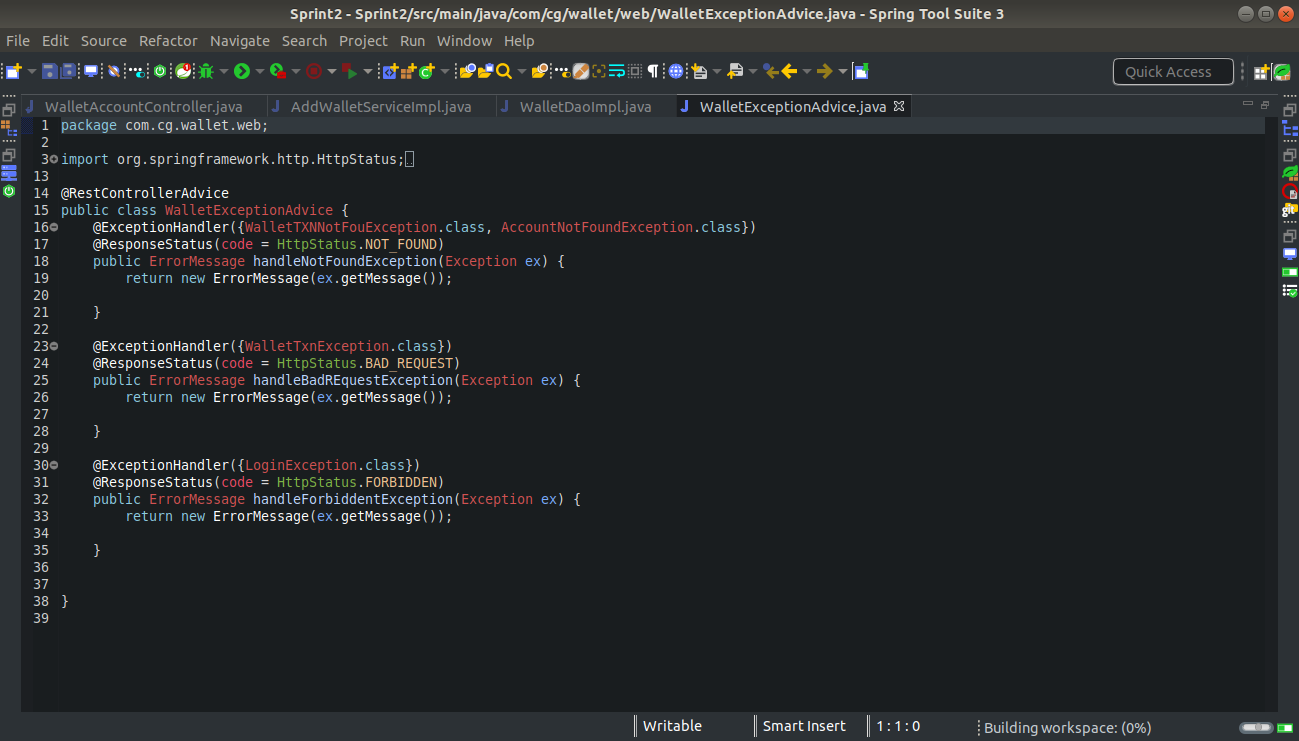
In Dao layer operations will be done in database and data will be stored.



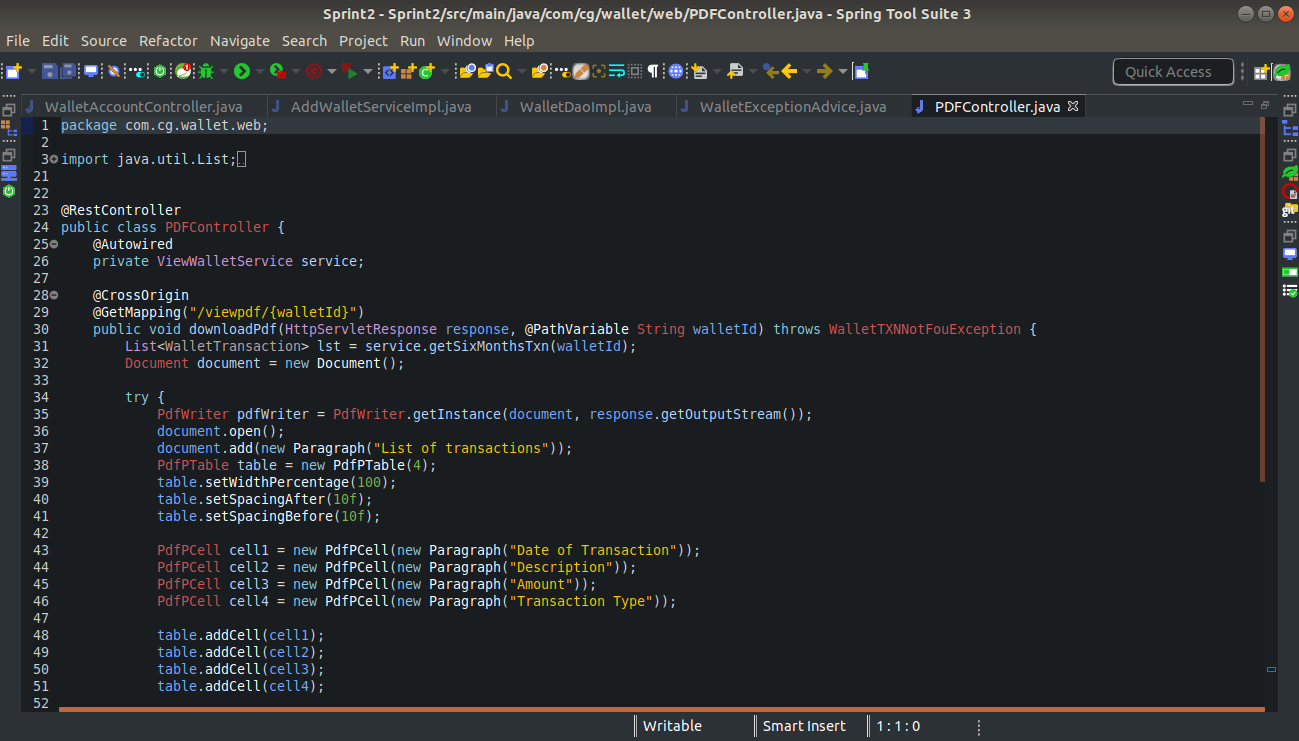


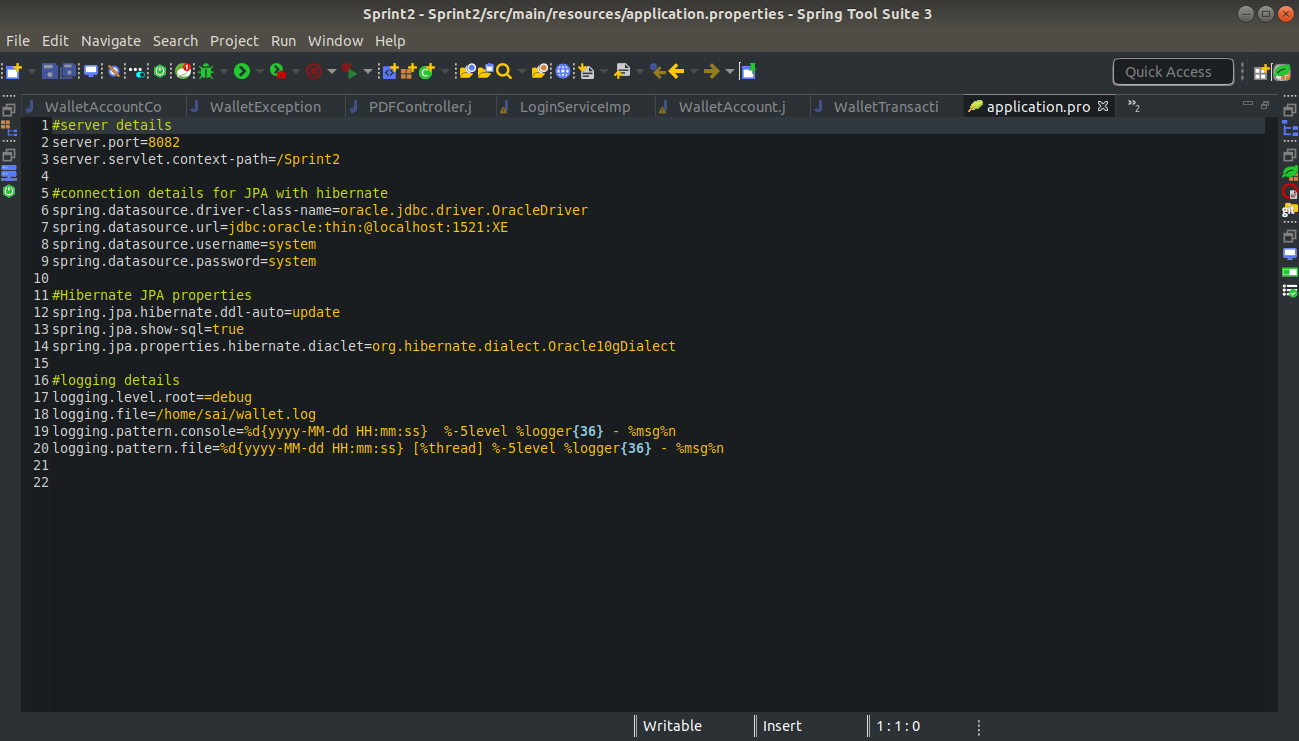
Data is stored in two tables one table for storing wallet account details and another table for storing all transactions.

And if there is any exception raised in any of the rest controllers it handled globally by ExceptionHandler



And for downloading pdf we have another rest controller





Properties used to connect to database and on which port the server should run.

**3. Brief Description of Work Done**

**Technologies learnt:**

1. Core Java
2. HTML
3. CSS
4. BootStrap
5. Angular
6. Spring

**Concepts:**

**Encapsulation:**

Every object is encapsulated in such a way, that its data and implementations of behaviours are not visible to another object. Encapsulation allows restriction of access of internal data.

Encapsulation is often referred to as information hiding. However, although the two terms are often used interchangeably, information hiding is really the result of encapsulation, not a synonym for it.

**Inheritance:**

It is one of the fundamental mechanisms for code reuse in object-oriented programming. Inheritance allows new classes to be derived from existing classes. In Java, inheritance specifies how different is the subclass from its parent class. Thus, we can add new variables, methods and also modify the inherited methods. To inherit a class, you simply extendthe super class into the subclass. Only single level inheritance is possible in Java.

* **Superclass and Subclass**: Any class preceding a specific class in the class hierarchy is said to be super class. On the other hand, any class followingspecific class in the hierarchy is called as subclass. All classes in Java are by default extensible. All Java classes that do not explicitly extend a parent class automatically extend the java.lang.Objectclass.

**Polymorphism:**

Poly meaning “many” and morph means “forms”

It’s capability of method to do different things based on object used for invoking method. Polymorphism also enables an object to determine which method implementation to invoke upon receiving a method call.

There are two ways in which polymorphism is implemented in Java.

1. Method Overloading

2. Method Overriding

Method Overloading is compile time polymorphism where the same method name has different meanings. Method overriding on other hand, is kind of runtime polymorphism where a subclass defines method with the same signature as defined by its superclass.

Method Overriding is run time polymorphism where the method in sub class has the same name and type signature as the method in super class, then the sub class method overrides the super class method.

**@Override:**

The Compiler checks that a method with this annotation really overrides a method from the Super class or not.

While it's not required to use this annotation while overriding a method, it helps to prevent errors. If a method marked with @Override fails to correctly override a method in one of its super classes, the compiler generates an error.

Most commonly, it is useful when a method in the base class is changed to have a different parameter list. A method in a subclass that used to override the superclass method no longer does so due to the changed method signature. This can sometimes cause strange and unexpected behavior, especially while dealing with complex inheritance structures. The @Overrideannotation safeguards against this.

@Override is useful in detecting changes in parent classes which has not been reported down the hierarchy. Without it, a method signature can be changed and altering its overrides can be forgotten. With @Override, the compiler catches it for you.

**Interface:**

An interface declaration can have two other components:

The public access specifier

The list of super interfaces

While a class can only extend one other class, an interface can extend any number of interfaces, and an interface cannot extend classes. An interface inherits all constants and methods from its super interface.

The interface body contains method declarations for the methods defined within the interface and constant declarations. All constant values defined in an interface are implicitly public, static and final. Similarly, all methods declared in an interface are implicitly public and abstract. One class can implement more than one interface at a time by separating those using commas.

**Maven Fundamental:**

•Maven is a tool for project management and build automation.

•Maven is not Ant++

•Maven serves a similar purpose to the Apache Ant tool, but it is based on different concepts and works in a profoundly different manner.

•Maven is hosted by the Apache Software Foundation, where it was formerly part of the Jakarta Project.

•It is pronounced as “May-ven” and Maven is a Yiddish (Jewish Lang) word which means “accumulator of knowledge “.

Maven provides features such as:

•Build tool Capabilities

•Run Reports

•Generate a website

•Dependency Management

•Repositories (Reusable Plug-ins)

•Continuous Integration build systems

•Portable

•Building configuration using maven are portable to another machine without any effort.

Maven allows to comprehend the complete state of a development effort in the shortest period of time. To attain this goal, Maven deals with:

**Making the build process easy:**

While using Maven doesn't eliminate the need to know about the underlying mechanisms, Maven does provide a lot of shielding from the details.

**Providing a uniform build system:**

Maven allows a project to build using its project object model (POM) and a set of plugins that are shared by all projects using Maven, providing a uniform build system. Once you familiarize yourself with how one Maven project builds you automatically know how all Maven projects build saving you immense amounts of time when trying to navigate many projects.

**Providing quality project information:**

Maven provides plenty of useful project information that is in part taken from your POM and in part generated from your project's sources. For example, Maven can provide:

•Change log document created directly from source control

•Cross referenced sources

•Mailing lists

•Dependency list

•Unit test reports including coverage

**Maven Dependency:**

When a dependency is declared within the context of your project, Maven tries to satisfy that dependency by looking in repositories. Maven uses a local repository to resolve its dependencies. If not found one or more remote repositories are consulted to find a dependency. If found, the dependency is downloaded to the local repository and used from the local repository.

**Collections :**

Collections Framework:

A Collection (sometimes called a container) is an object that groups multiple elements into a single unit. Collection is used to store, retrieve objects, and to transmit them from one method to another.

The Collections API (also called the Collections framework) standardizes the way in which groups of objects are handled by your programs. It presents a set of standard utility classes to manage such collections.

The collection classes are the fundamental building blocks of the more complicated data structures used in the other Java packages in your own applications. There are several types of collections. They vary in storage mechanisms used, in the way they access data, and in the rules about what data may be stored.

Advantages of Collections:

* Reduces programming effort by providing useful data structures and algorithms so you do not have to write them yourself.
* Increases performance by providing high-performance implementations of useful data structures and algorithms. Since the various implementations of each interface are interchangeable, programs can be easily tuned by switching implementations.
* Provides interoperability between unrelated APIs by establishing a common language to pass collections back and forth.
* Reduces the effort required to learn APIs by eliminating the need to learn multiple ad hoc collection APIs.
* Reduces the effort required to design and implement APIs by eliminating the need to produce ad hoc collections APIs.
* Fosters software reuse by providing a standard interface for collections and algorithms to manipulate them.

**Interfaces and Implementation:**

Collection Interfaces :

Following are the four major interfaces:

* Set Interface: holds only unique values and rejects duplicates.
* List Interface: represents an ordered list of objects, meaning the elements of a List can be accessed in a specific order, and by an index too. List can hold duplicates.
* Queue Interface: represents an ordered list of objects just like a List. However, a queue is designed to have elements inserted at the end of the queue, and elements removed from the beginning of the queue. Just like a queue in a supermarket!
* Map Interface: represents a mapping between a key and a value. The Map interface is not a subtype of the Collection interface. A Map cannot contain duplicate keys; each key can map to at most one value. The Map implementations let you do things like search for a value based on the key, ask for a collection of just the values, or ask for a collection of just the keys.
* SortedSet Interface: is a Set that maintains its elements in ascending order. Several additional operations are provided to take advantage of the ordering.
* SortedMap Interface: is a Map that maintains its mappings in ascending key order. This is the Map analog of SortedSet. Sorted maps are used for naturally ordered collections of key/value pairs, such as dictionaries and telephone directories.

**Collection Implementations:**

The Java Collections Framework provides several general-purpose implementations of the Set, List, and Map interfaces. The general-purpose implementations are summarized in the table above.

* HashSet: is an unsorted, unordered Set. It uses the hash code of the object being inserted, so the more efficient your hash Code () implementation is, the better access performance you will get. Use this class when you want a collection with no duplicates and you do not care about order when you iterate through it. Implements the Set interface.
* Linked HashSet: differs from HashSet by guaranteeing that the order of the elements during iteration is the same as the order they were inserted into the Linked HashSet.
* Tree Set: implements the SortedSet interface. Like Linked HashSet, TreeSet also guarantees the order of the elements when iterated, but the order is the sorting order of the elements. This order is determined either by their natural order (if they implement Comparable), or by a specific Comparator implementation.
* Array List: Think of this as a growable array. It gives you fast iteration and fast random access. It is an ordered collection (by index). However, it is not sorted. Array List now implements the new Random Access interface —a marker interface (meaning it has no methods) that says, “this list supports fast (generally constant time) random access.” Choose this over a Linked List when you need fast iteration but are not as likely to be doing a lot of insertion and deletion.
* Linked List: A Linked List is ordered by index position, like Array List, except that the elements are doubly-linked to one another.

**ANGULAR**

**Introduction To HTML:**

HTML is a language for describing web pages.

* It stands for Hyper Text Markup Language
* HTML is a markup language and not a programming language
* HTML uses markup tags to describe web pages.

Rules for HTML5 Tags are:

The document must be included with an HTML5 DOCTYPE.

* Tags and attributes are case-insensitive.
* Attributes do not need to be quoted.
* End tags are not required for every element.
* Some attributes may be empty such as checked and disabled

**Introduction To CSS:**

Cascading Style Sheets (CSS) is a style sheet language used to describe the presentation (that is, the look and formatting) of a document written in a markup language.

CSS was created by Hakon Wium Lie and Bert Bos and was adopted as a W3C Recommendation in late 1996.

**CSS Syntax:**

A CSS rule has two main parts:

* A selector
* One or more declarations

The selector is normally the HTML element you want to style.

Each declaration consists of a property and a value.

The property is the style attribute you want to change. Each property has a value.

**Types of CSS:**

Three CSS implementations

* Inline: Affects only the element applied to
* Embedded: Affects only the elements in a single file
* External: Linked to an unlimited number of files

Inline Style Sheets:

* All style attribute are specified in the tag itself.
* It gives desired effect on that tag only. It does not affect any other HTML tag.

Embedded Style Sheet:

* Set of style definitions placed within <STYLE> tags.
* Added to the <HEAD> area of file

External CSS:

The <LINK> element is used to attach an external CSS document to an HTML document

* All style definition are store in one file (.css file)
* This file gets called by the HTML file during page loading
* Syntax: <link rel=“stylesheet” href=“filename.css” type=“text/css”>

**Introduction To Bootstrap:**

Bootstrap is the most popular, open source HTML, CSS, JavaScript framework for developing responsive, mobile first projects on the web.

**Advantages of Bootstrap:**

* Speed of Development **-**The speed of development is one of the major reason of utilizing Bootstrap. If you want to develop an application or a website quickly, it is vitally important to consider using Bootstrap. Why? Because, It helps to save your coding effort by offering less(prior version 4) CSS functionality and pre-built blocks of code rather than need of structuring your code from the scratch. You can buy ready-made themes of Bootstrap and use them either as it is or you can even alter them to fit your requirements which will help to achieve efficient results in development.
* Responsiveness **–**With the huge growth in usage of varied kinds of mobile devices, over the period of time, the requirement to have a responsive website that will fit in every size has become compulsory and important. Bootstrap is equipped with responsive layout and 12-column grid system that help dynamically adjust the website to a suitable screen resolution. The ‘responsive utility classes’ feature of Bootstrap enables you to hide / show a certain section of content for a particular screen size.
* Consistency **-**Consistency was the fundamental principle behind the introduction of Bootstrap. It ensures the ultimate consistency regardless of designer/developer, who is working on it. Moreover, the results work uniformly across various browsers and the output remains same.
* Customizable **-**Bootstrap facilitates you to customize it as per the designs of your project and helps in designing tailor made websites, according to your specifications. It has the facility to select any feature that is actually needed to create a customized website (The web developers can make a choice to select the aspects which are required which can be simply complete by utilizing Bootstrap customize page). With this feature, one can get rid of what they do not require.
* Support **-**Bootstrap helps to fix issues promptly with an immense support community. Bootstrap also releases continual updates to fix any new issues. Currently, it is being developed, hosted and maintained by GitHub with over 9000 commits and 500 contributors.

**Angular Framework:**

* Extendable:It is easy to figure out how even a complex Angular app works once you understand the basics—and that means you can easily enhance applications to create new and useful features for your users.
* Maintainable: Angular apps are easy to debug and fix, which means that long-term maintenance is simplified.
* Testable:Angular has good support for unit and end-to-end testing, meaning that you can find and fix defects before your users do.

**HTTP Requests:**

* Angular applications often obtain data using http
* Application issues http get requests to a web server which returns http response-Observable to the application.
* Application then processes that data.

**Web Applications:**

Desktop applicationsare those that have stand-alone applications running on them. They are maintenance heavy.

Server-side applicationson the other hand run on the server side of a client-server system. The server is then able to access server side resources like databases, server components, and provide these services to multiple clients concurrently.

**Client-side scripting versus Server-side scripting:**

Client-side scriptsrun on the client-side of a client-server application. For example, in a web application, we have written Javascript that runs on the browser to perform presentation validation.

Server-side scriptsrun on the server-end of a client-server application. For example, when a HTML form is submitted, the data may be persisted on a database at the server side.

From a web application perspective, there are many advantages to this:

The response is in HTML form, so complex code that is executed to generate the response stays at the server-side.

Server is able to access server side resources like databases, server components, which ensures centralized control.

**Session Tracking:**

In Session tracking, client first makes a request for any servlet or any page, the container receives the request.

The container then generates a unique identification, called Session ID, for that client and gives it back to the client along with the response. This ID gets stores on the client machine. Thereafter when the client sends a request again to the server, it also sends the session Id with the request. The container sees the Id and sends back the response. While seeing the ID the container recognizes that it is the same previous client making a request.

HTTP is a stateless protocol, that is it cannot persist the information.

HTTP always treats each request as a new request. Each request made by a Web browser (client), for an image, an HTML page, or other Web object, is made via a new connection. In other words, in HTTP, the client makes a connection to the server, sends the request, gets the response, and closes the connection. Hence to maintain the state across pages we need some sort of session tracking mechanism.

There are several ways in which session tracking can be done in JEE.

* Hidden Form Fields: It is the simplest way of saving user-specific information where hidden parameters are encoded inside an HTML form, such as the username and type of transaction being made
* URL Rewriting: URL rewriting involves placing a session id in the URL.
* Cookies: It is a small text file storing client information, cookies are created by the server and saved on client’s machine.
* Session Tracking API: Servlets API provides Session Tracking API

**JPA WITH HIBERNATE**

**Entity Manager:**

The EntityManager is the primary interface used by application developers to interact with the JPA runtime.

**Persistence Context:**

Persistence context defines a scope under which particular entity instances are created, persisted, and removed.

Every EntityManager manages its ownpersistence context. In short, persistence context is a memory area for EntityManager to work on entity instance.

The slide diagram shows a sample JPA runtime. JPA uses EntityManager instance to manage objects which required to be persisted. Such objects are called Entities. Entities managed by EntityManager travels through different life cycle phases.

**New State**: When an entity object is initially created its state is New. In this state the object is not yet associated with an Entity Manager and has no representation in the database.

**Managed State:** An entity object becomes Managedwhen it is persisted to the database via an EntityManager’s persist method which must be invoked within an active transaction. On transaction commit, the owning Entity Manager stores the new entity object to the database.

Entity objects retrieved from the database by an EntityManager are also in the Managedstate.

If a managed entity object is modified within an active transaction the change is detected by the owning EntityManager and the update is propagated to the database on transaction commit.

**Detached State:** Represents entity objects that have been disconnected from the EntityManager. For instance, all the managed objects of an EntityManager become detached when the EntityManager is closed.

**Removed State:** A managed entity object can also be retrieved from the database and marked for deletion, by using the EntityManager’s remove method within an active transaction. The entity object changes its state from Managed to Removed, and is physically deleted from the database during commit.

**Transactions:**

A transaction is a set of operations that either fail or succeed as a unit.

Transactions are a fundamental part of persistence.

A database transaction consists of a set of DML (Data Manipulation Language) operations that are committed or rolled back as a single unit.

An object level transaction is one in which a set of changes made to a set of objects are committed to the database as a single unit.

**Java Persistence Query Language:**

The Java Persistence Query Language (JPQL) is a platform-independent object-oriented query language defined as part of the Java Persistence API (JPA) specification.

JPQL is used to make queries against entities stored in a relational database.

The JPQL defines queries for entities and their persistent state. The query language allows you to write portable queries that work regardless of the underlying data store.

The JPQL can be considered as an object oriented version of SQL. Users familiar with SQL should find JPQL very easy to learn and use.

The main difference between SQL and JPQL is that SQL works with relational database tables, records and fields, whereas JPQL works with Java classes and objects.

**Spring 5:**

Spring is a Java platform that provides comprehensive infrastructure support for developing Java applications with development tools.

Any java application can benefit from Spring in terms of

* Automation of deployment
* Convention over configuration
* Testing is simpler
* Services to enable a cohesive technology experience not only for the developers, but also for the businesses

Addresses the complexity of enterprise application development

Spring Boot favors convention over configuration and is designed to get you up and running as quickly

Spring Framework provides a comprehensive programming and configuration model for modern Java-based enterprise applications on any kind of deployment platform

Spring Web Flow builds on Spring MVC and allows implementing the "flows" of a web application

Spring Web Services (Spring-WS) is a product of the Spring community focused on creating document-driven Web services

Spring Integration extends the Spring programming model to support the well-known Enterprise Integration Patterns

Microservices : Software architecture design pattern, in which complex applications are composed of small, independent processes communicating with each other using language-agnostic APIs. These services are small, highly decoupled and focus on doing a small task.

Spring Boot: It’s a new framework designed to simplify the bootstrapping and development of a new Spring application with opinionated approach to configuration, freeing developers from the need to define boilerplate configuration.

Advantages of Spring

* Dependency Injection
* Offers layers of abstraction
* It allows you to write less code, instead of writing more code
* Spring JDBC
* Instead of writing Jdbc using 7 steps, It allows you to write in two steps
* Spring ORM
* Integrate to any framework such as struts, jsf, Servlet and Jsp. Hibernate, jpa
* Allows you to create restful webservices
* Supports MVC pattern for web applications
* Supports Internationalization
* Supports AOP (Aspect Oriented Programming)
* Separates Aspect logic and business logic
* Aspect Logic
* Code not related to your application logic
* Logger, Validation, Conversion, Formatting and Authentication
* Easy to write test cases using Spring
* API works on top of Junit.

**Spring MVC REST Workflow:**

The following steps describe a typical Spring MVC REST workflow:

The client sends a request to a web service in URI form.

The request is intercepted by the DispatcherServlet which looks for Handler Mappings and its type.• The Handler Mappings section defined in the application context file tells DispatcherServlet which strategy to use to find controllers based on the incoming request.• Spring MVC supports three different types of mapping request URIs to controllers: annotation, name conventions and explicit mappings.

Requests are processed by the Controller and the response is returned to the DispatcherServlet which then dispatches to the view.

**HTTP Methods in Rest:**

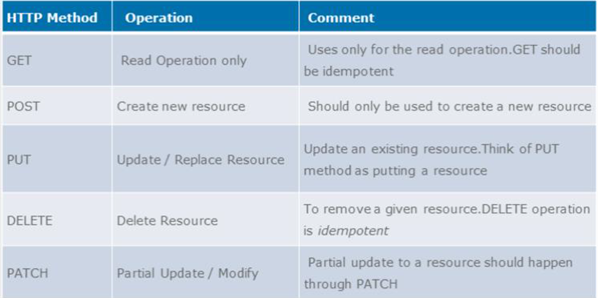


Table2.1

**HTTP Status Code:**

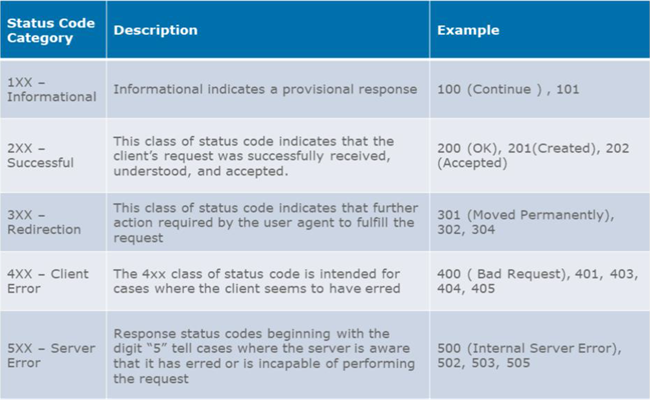


Table 2.2

**Exception Handling:**

@ExceptionHandler along with @ResponseStatus to map the exception to the custom method in controller which can handle all exception in that controller.

In @ExceptionHandler annotation we can include the Exception classes which we need to handle for this controller.

The new annotation allows the multiple scattered @ExceptionHandlerfrom before to be consolidated into a single, global error handling component.

The actual mechanism is extremely simple but also very flexible:

It allows full control over the body of the response as well as the status code

It allows mapping of several exceptions to the same method, to be handled together

It makes good use of the newer RESTful ResposeEntity response.

**Conclusion**

Throughout my internship, I could understand more about the definition of an IT technician and programmer and prepare myself to become a responsible and innovative technician and programmer in future. Along my training period, I realize that observation is a main element to find out the root cause of a problem. During my project, I cooperate with my colleagues to determine the problems. Moreover, the project indirectly helps me to learn independently, discipline myself, be considerate/patient, self-trust, take initiative and the ability to solve problems. Besides, my communication skills are strengthened as well when communicating with others. During my training period, I have received criticism and advice from my trainer when mistakes were made. However, those advices are useful guidance for me to change myself and avoid myself making the same mistakes again.

It was a good experience and memories as not only I have gained experience, but also new friends and knowledge.

Finally, I would like to conclude by saying thanks to T J Govindarajulu my trainer from whom I have learnt many things not only related to technology but also how to grow as a person.