**Womanista**

|  |
| --- |
|  |

SRE

PUCIT

Group Members

|  |  |
| --- | --- |
| Alishba Siddique  Fatima Shahid  Khadija Zubair | BSEF19A007  BSEF19A008  BSEF19A043 |

29-March-21

SE F19 AFT

Submitted to:

Ma'am Amina

**Womanista**  This is an app that every woman that wishes for. As the name suggests it’s all about woman so our app provides services that are necessary for a woman’s life. It makes a female’s living more comfortable as it includes services that are that based on reality, every woman has to go through the difficult times and have these reality checks like menstruation at those times Womanista stood by the female as a friend in need to survive through the reality hassle comfortably and easily

therefore, it includes following services:

* Gynecologist Home Appointment
* Self-Defense Training Tutorial
* Self-Defense Tools on Sale
* Sanitary Products Emergency Delivery
* Women Protection Laws Awareness
* Ladies Driver Cab Bookings

1. **Gynecologist Home Appointment:**

This feature allows the user to book Gynecologist Appointment for to visit her at home and visiting Doctor must be a Lady doctor as most female prefer and is more comfortable and the Gynecologist visits the user’s home both in Emergency cases or for regular checkup. User can set the particular time and date on which she needs a doctor

1. **Self-Defense Training Tutorial:**

In this unsafe society, every woman should learn the art of self-defense to protect herself in critical situation like molestation or harassment so This feature allows the provide some self-defense training tutorial to the learn some self-defense techniques like Groin kick, Heel palm kick, Elbow strike etc. to defend and protect herself by her own self and to be stronger enough to face tense situations.

1. **Self-Defense Tools on Sale:**

This feature allows the user to buy various self-defense tools which are portable, easy to use and inexpensive like pepper spray etc. so these tools can help them in defending themselves in critical situations.

1. **Sanitary Products Emergency Delivery:**

This feature allows the user to order to deliver sanitary products like sanitary pads etc., in case of emergency or not, from whichever location and the ordered sanitary product is delivered to them within 15 to 20 minutes.

1. **Women Protection Laws Awareness:**

Most of the female in community, didn’t aware of the Laws for Women Protection that why they bear any type of violence. This feature provides a detailed information about the Laws for Women Protection to user so a woman should aware of them and get justice through them.

1. **Female Driver Cab Booking:**

This feature allows the user to book transport with Female driving the cab as a Driver so any women who travel comfortably even late at night. She can enter the destination point and pickup point and the App will show her the available drivers near her location. She can select the driver of her choice and can Add emergency number over there; the

app will automatically dial call on that number if the ride is taking wrong route. She can share her ride with her friends or family.

**Features in app:**

* After opening the app, the first option appear will be SIGNUP or LOGIN
* The sign-up option will consider the following details which user will enter:
* Name
* Number
* Address
* Email
* Gender
* CNIC
* Password (for future login)

After signing up user can easily access all the functions in the app. Once user has made an account on the app, now every time the user will open the app, she will enter the email id and the password they have set.

**Technology Stack:**

* Flutter

**Design Patterns:**

The following are the design patterns which we’ll apply on the application listed as feature by feature:

1. **Visitor design pattern**

This design we’ll apply on **Gynecologist Home Appointment** as this feature includes the personal visit of the doctor so Visitor design pattern will be the best because Visitor design pattern is one of the **behavioral design patterns**. It is used when we have to perform an operation on a group of similar kind of Objects. With the help of visitor pattern, we can move the operational logic from the objects to another class

The visitor pattern consists of two parts:

* A method called **Visit()** which is implemented by the visitor and is called for every element in the data structure
* Visit-able classes providing  **Accept()** methods that accept a visitor

1. **State design pattern**

This design we’ll apply on **Self-Defense Training Tutorial** as this feature includes the video calling with the trainer so State design pattern will be the best because the state design pattern is one of the **behavioral design patterns** — it helps to solve problem that involve interaction between classes, the classes being the state of the object — used when objects behave in different ways based on the current state, they are in. So, can our system be represented as having a finite number of states? Yes. For the sake of this case study, they include: IDLE, RECEIVING, CALLING, ONCALL.

1. **Observer design pattern**

This design we’ll apply on **Self-Defense Tools on Sale** as this feature includes the buying self-defense products so it required filters when searching the products so Observer

design pattern**, behavioral design pattern,** will be the best because the goal of the observer design pattern is to create a relationship between a subject and their observers. The observers are set to wait for updated data. If the subject changes, the observers are notified almost instantly.

1. **Facade design pattern**

This design we’ll apply on **Sanitary Products Emergency Delivery** as this feature includes the buying sanitary products so it required delivery when ordered the products so Facade design pattern**, structural design pattern,** will be the best because **Facade** is a structural design pattern that provides a simplified interface to a library, a framework, or any other complex set of classes.

1. **Immutable design pattern (Read-Only Method)**

This design we’ll apply on **Women Protection Law Awareness** as this feature includes the opening the law articles for reading only Immutable Interface so design patternis a [pattern](https://en.wikipedia.org/wiki/Design_pattern_(computer_science)) for designing an [immutable object](https://en.wikipedia.org/wiki/Immutable_object). The immutable interface pattern involves defining a type which does not provide any [methods](https://en.wikipedia.org/wiki/Method_(computer_science)) which mutate state. Objects which are referenced by that type are not seen to have any mutable state, and appear immutable.

1. **Command design pattern**

This design we’ll apply on **Cab booking as** this feature includes booking a cab of female drivers so Command design pattern will be the best because Command Pattern comes under **behavioral pattern**. It encapsulates all the information in an object for a particular action, that action to be performed at a later time. Cab Booking System can be developed using Command Design Pattern and using some scheduling algorithm along with.

1. **Proxy design pattern**

This design we’ll apply on **Alternate Payment Method Selection** as this feature includes the buying products so it required selection of payment method when ordered the products so Proxy design pattern will be the best because **Proxy** is a **structural design pattern** that lets you provide a substitute or placeholder for another object. A proxy

controls access to the original object, allowing you to perform something either before or after the request gets through to the original object.

1. **Domain Inventory pattern**

This design we’ll apply on **Inventory Management** as this feature includes the storage of the products so it inventories which have to be managed so Domain inventory design pattern will be the best because Domain Inventory is a [design pattern](https://en.wikipedia.org/wiki/Design_pattern), applied within the [**service-orientation**](https://en.wikipedia.org/wiki/Service-orientation) [design paradigm](https://en.wikipedia.org/wiki/Design_paradigm), whose application enables creating pools of services, which correspond to different segments of the enterprise, instead of creating a single enterprise-wide pool of services. This design pattern is usually applied when it is not possible to create a single inventory of services for whole of the enterprise by following the same design standards across the different segments of the enterprise.

1. **Interceptor pattern**

This design we’ll apply on **Authentication** as this feature includes the authentication after login into the application so Interceptor pattern will be the best because an interceptor pattern is a [software design pattern](https://en.wikipedia.org/wiki/Software_design_pattern) that is used when software systems or [frameworks](https://en.wikipedia.org/wiki/Software_framework) want to offer a way to change, or augment, their usual processing cycle.  the Interceptor pattern is used for intercepting a request made to a service. It’s mostly being used for **security purposes (authentication & authorization)**, but also for logging.

1. **Strategy pattern**

This design we’ll apply on **Log-in/log-out** as this feature includes logging into the application so Strategy pattern will be the best because strategy pattern a class behavior or its algorithm can be changed at run time. This type of design pattern comes under **behavior pattern.**

In Strategy pattern, we create objects which represent various strategies and a context object whose behavior varies as per its strategy object. The strategy object changes the executing algorithm of the context object.

1. **Flyweight pattern**

This design we’ll apply on **Location** as this feature includes location for cab booking in the application so flyweight pattern will be the best because Flyweight pattern is primarily used to reduce the number of objects created and to decrease memory footprint and increase performance. This type of design pattern comes under **structural pattern** as this pattern provides ways to decrease object count thus improving the object structure of application