بسم الله الرحمن الرحيم

Acknowledgement

First and foremost, we thank Allah Almighty who paved the path for us in achieving the desired goal. I would like to express my sincere gratitude to my supervisor, **Dr. Mahmoud Wahdan**, for his guidance and support throughout this project. Their insights and feedback were invaluable in helping me shape and refine my ideas, I would also like to thank **Eng. Nadden Ayman** for their support and encouragement, I also would like to thank **Eng. Mennatallah Mohamed ,** Life coaches **Eman Elshami**, **Nesma Rayes** for helping us with the whole journey creating this system as well as the **El-Shorouk Academy, Dr. Mohamed Salah,** and **Prof. Ahmed Elabbasy,** and his Asistant **Ms. Hala Saied Hamed Elgarf** for providing resources and facilities that made this project possible. Also, we want to thank **Eng. Mohamed Mahmoud,** Also**,** we are grateful to my family and friends for their unwavering support and understanding during this challenging but rewarding journey. Finally, I would like to acknowledge the participants who generously gave their time and shared their experiences with me. Without their contributions, this project would not have been possible.

Thank you all for your support and guidance.

Abstract

Psychological diagnosis and examination system (INFLOW) is Web Application that represents faster communication between the patient and life coach, providing that this communication through the internet or face to face, checking online by reservation, and performing some tests to know more about themselves, And this system is a Customized Psychological Diagnosis System is platform that enables users to accurately diagnose emotional health conditions, Based on Lüscher Color Test System, And there is another system is Mobile Application. Its purpose is managing client's meetings and data for life coaches to follow meetings and data to be up to date and to take notes.

Table of Contents

Chapter One: Introduction **5**

1.1 Motivation: 6

1.2 Project objectives: 6

1.3 Project Development Methodology: 6

1.4 Project Plan: 8

1.5 Tools & Technology: 9

1.6 Definitions, Acronyms & Abbreviations: 10

Chapter Two: Background and Literature Survy **11**

2.1 Lüscher Color Test System: 12

2.2 Similar Projects: 13

2.3 Tools Survey: 18

Chapter Three: System Specification **30**

3.1 System Description: 31

3.3 System Models: 38

Chapter Four: System Design **40**

4.1 System Architecture: 41

4.2 Database Design: 42

4.3 User Interface Design: 44

Chapter Five: Implementation **65**

5.1 Website 66

5.2 Chatbot Code: 99

5.3 Data Analysis code: 102

5.4 Mobile code: 112

Chapter Six: Testing **115**

6.1 Testing Plan: 116

6.2 Test Objective: 116

6.3 Testing Approach: 117

Chapter Seven: Conclusion and Future Work **124**

7.1 Summary of achieved result: 125

7.2 further development 125

References **127**

Chapter One
Introduction

1.1 Motivation:

According to the **World Health Organization** Emotional health is about how we think and feel. It is about our sense of wellbeing, our ability to cope with life events and how we acknowledge our own emotions as well as those of others, Emotional health disorders can have a significant impact on individuals and society, and accurate diagnosis is an essential first step in addressing them. Without proper diagnosis, individuals may suffer needlessly, and their condition may worsen over time. Moreover, psychological diagnosis and examination systems can inform research on mental health by providing a common language and classification system for emotional health disorders. This can help researchers compare findings across studies and advance our understanding of the causes and treatments of emotional health issues.

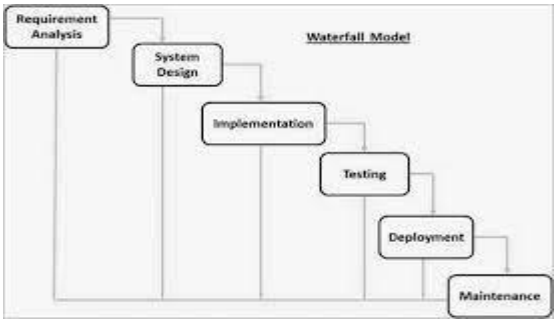
1.2 Project objectives:

Psychological diagnosis and examination system (INFLOW) is Web Application that represents faster communication between the patient and life coach, providing that this communication through the internet or face to face, checking online by reservation, and performing some tests to know more about themselves, And this system is a Customized Psychological Diagnosis System platform that enables users to accurately diagnose emotional health conditions, Based on Lüscher Color Test System, And there is another system is Mobile Application. Its purpose is managing client's meetings and data for life coaches to follow meetings and data to be up to date and to take notes.

1.3 Project Development Methodology:

Different software engineering life cycles have been proposed over the years, these projects follow the **waterfall model**, the basic cycle remains:

1. Requirement Analysis.
2. System Design.
3. Implementation.
4. Testing.
5. Deployment.
6. Maintenance.



The **waterfall model** is the popular version of the systems development life cycle model for software engineering, it`s the first Process Model to be introduced, it is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed fully before the next phase can begin. This type of model is basically used for the project which is small and there are no uncertain requirements. At the end of each phase, a review takes place to determine if the project is on the right path and whether to continue or discard the project. In this model the testing starts only after the development is complete. in the waterfall model Once a phase of development is completed, the development proceeds to the next phase and there is no turning back.

Advantages of waterfall model:

- This model is simple and easy to understand and use.
- It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
- In this model phases are processed and completed one at a time. Phases do not overlap.
- The waterfall model works well for smaller projects where requirements are very well understood.

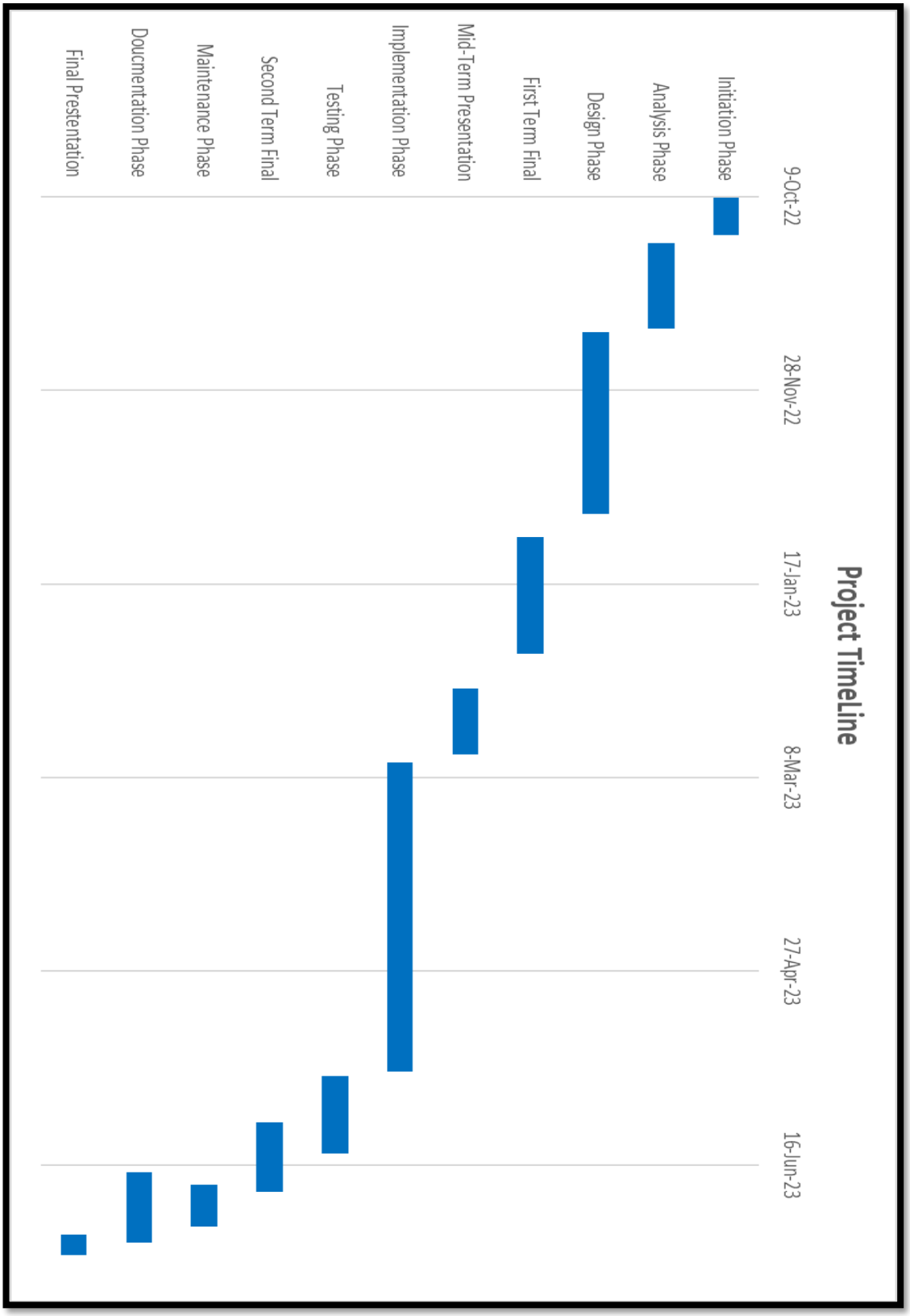
Disadvantages of waterfall model:

- Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.
- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects
- Not suitable for projects where requirements are at a moderate to high risk of changing.

Why we used waterfall model for this project:

- The requirements are very well known, clear and fixed.
- Product definition is stable.
- Technology is understood.
- There are no ambiguous requirements.
- Ample resources with required expertise are available freely.
- The project is uncomplicated.

1.4 Project Plan:



1.5 Tools & Technology:

1. **IDEs:**
    - *Visual Studio*
    - *Android Studio*
    - *Jupiter Notebook*

2. **Programming Language:**
    - *Python*
    - *Java*
    - *Kotlin*
    - *PHP*
    - *Java Script*
    - *HTML, CSS*

3. **Database Engines:**
    - *MySQL*

4. **Frameworks and Libraries:**
    - *React*
    - *Bootstrap*
    - *Laravel*
    - *Streamlit*
    - *Junit*
    - *Cucumber*
    - *Postman*
    - *Firebase*
    - *Google play Service*
    - *IntelliJ DEA*
    - *TestNG*
    - *Selenium*
    - *NumPy*

5. **Design Tools:**
    - *Figma*

6. **Technologies:**
    - *Git*
    - *Google meet*
    - *GitHub*
    - *Google Collab*

1.6 Definitions, Acronyms & Abbreviations:

Term	Definition
IDE	Integrated Development Environment
API	Application Programming Interface
MySQL	a Relational Database Management System
NumPy	python library for scientific computing. It provides a high- performance multidimensional array object written in topof C Programming with Python C.
Firebase	App development platform provide set of tools to build better apps
Git	Version Control System (VCS)
GitHub	Version Control System host
Google meet	Virtual workspace for communication and collaboration
Google Collab	Cloud service to Execute Jupyter Notebook Files
Postman	Is an API development tool which helps to build, test and modify APIs.
Cucumber	Is a testing tool that supports Behavior Driven Development
Selenium	Is a free automated testing framework used to validate web applications across different browsers and platforms
IntelliJ DEA	Is IDE for JVM languages designed to maximize developer productivity.
Streamlit	Is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science.
Laravel	Is a free and open-source PHP web framework.
Bootstrap	Is a free and open-source CSS framework directed at responsive, mobile-first front-end web development.
React	Is a JavaScript-based UI development library.

Chapter Two

Background& literature survey

2.1 Lüscher Color Test System:

The Lüscher Color Test System is a psychological assessment tool has been used for over 50 years by psychologists, psychiatrists, and other mental health professionals to measure a person's emotional state. It consists of eight colored cards, each with a distinct hue, saturation, and brightness. The test is administered by asking the participant to select the color they feel best represents their current emotional state.

The colors are arranged in a circle, and the participant is asked to select the color that best represents their current emotional state. The order in which the colors are selected is also important, as it can provide insight into the individual's emotional state at the time of the test. The results are then analyzed to provide insight into the individual's emotional state.

The results of the Lüscher Color Test System are interpreted by a psychologist or other mental health professional. The results are analyzed to gain insight into the individual's emotional state, as well as to identify any underlying causes of emotional distress. The results can also be used to develop a treatment plan for the individual.

The Lüscher Color Test System is a valuable tool for mental health professionals, as it can provide insight into a person's emotional state and assist in the development of an effective treatment plan. It is also a useful tool for individuals, as it can help them gain insight into their own emotional state and provide a better understanding of their feelings and motivations, The Lüscher Color Test System is a simple and effective tool that can be used to gain insight into a person's emotional state and provide insight into the underlying causes of their emotional distress. It is an effective way for mental health professionals to gain insight into a person's emotional state, as well as for individuals to gain insight into their own emotional state.

The Lüscher Color Test System limitations is that it is not a diagnostic tool, and the results should not be used to diagnose a mental disorder. It is also important to note that the results of the test can be affected by the individual's emotional state at the time of the test, so the results should be interpreted with caution.

2.2 Similar Projects:

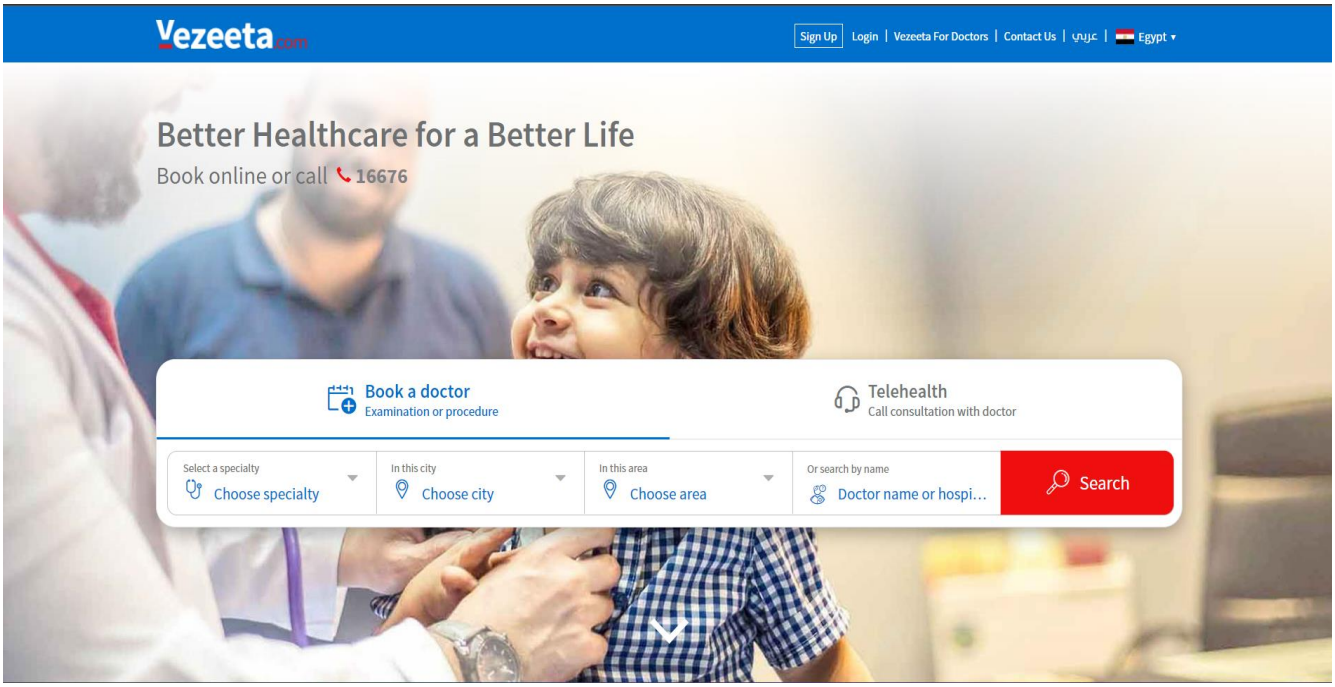
1. Vezeeta:

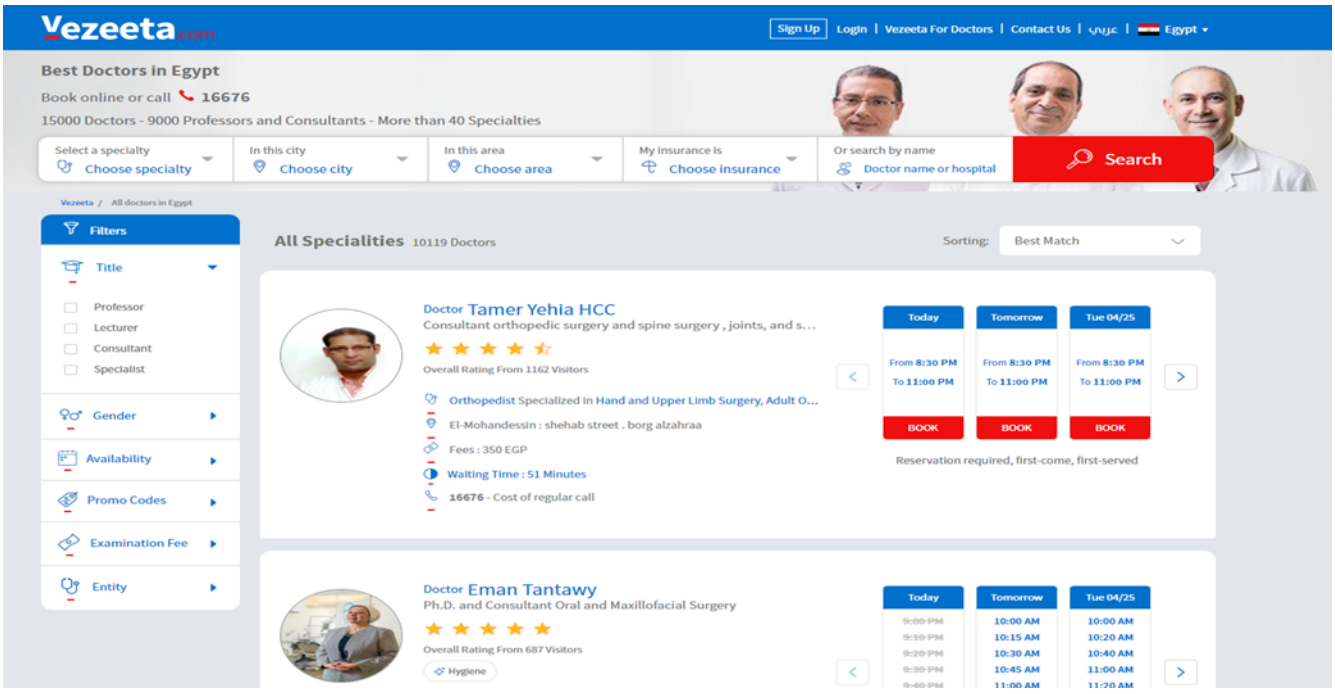
- **Vezeeta** is a digital healthcare platform that provides an online booking service for healthcare appointments. It operates in several countries in the Middle East and Africa, including Egypt, Saudi Arabia, Jordan, and Lebanon. The platform allows patients to search for and book appointments with healthcare providers such as doctors, dentists, and other specialists. In addition to booking appointments, Vezeeta also offers features such as virtual consultations, online prescription renewals, and access to patient medical records. Vezeeta's aim is to make healthcare more accessible and convenient for patients, while also helping healthcare providers to better manage their schedules and patient care. The platform has won several awards for its innovative approach to healthcare and has received funding from several investors.

- **Advantages of Vezeeta:**
  1. One of the biggest advantages of Vezeeta is its convenience. Patients can book appointments with healthcare providers from the comfort of their homes or offices, without the need to physically visit a clinic or hospital.
  2. It saves time for both patients and healthcare providers.
  3. It makes healthcare services more accessible to patients.
  4. It offers virtual consultations.
  5. It empowers patients by giving them more control over their healthcare choices and by providing access to their medical records.

- **Disadvantages of Vezeeta:**
  1. Limited to certain regions.
  2. Dependence on technology.
  3. Reliance on internet connectivity.
  4. Potential for misdiagnosis.
  5. Privacy concerns.

- **Screen Shots of Vezeeta:**





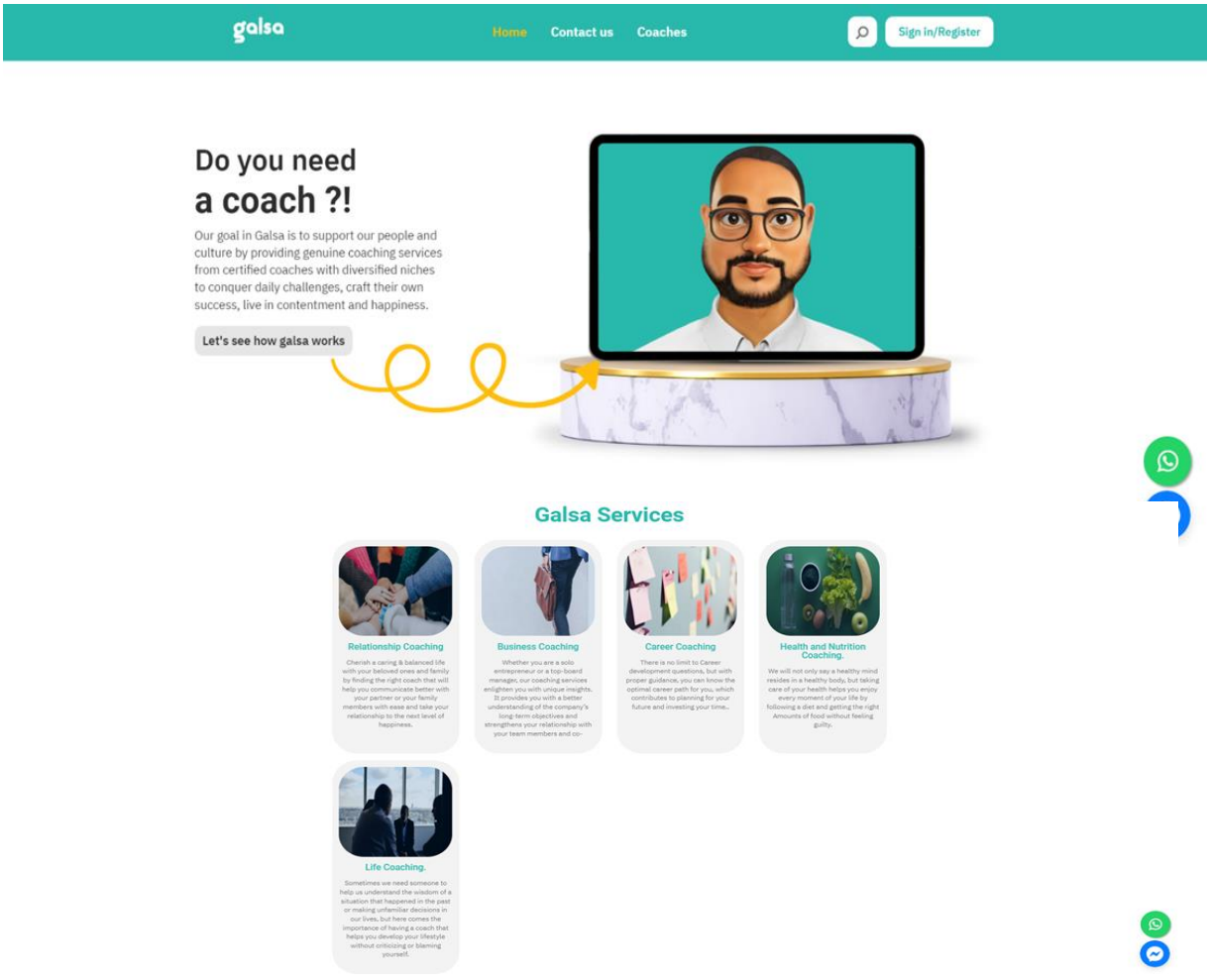
2. Galsa:

- **Galsa's** goal is to support people and culture by providing genuine coaching services from certified coaches with diversified niches to conquer daily challenges, craft their own success, live in contentment and happiness, and look like Vezeeta.

- **Advantages of Galsa:**
  1. The ability to give rate for the doctor so for next users to know doctor's skills.
  2. Not only offline treatment exists in the system.
  3. It saves time for both patients and healthcare providers.
  4. It makes healthcare services more accessible to patients.

- **Disadvantages of Galsa:**
  1. It doesn't contain tests for users entering system.
  2. It doesn't contain the idea of publishing articles for users.
  3. Limited to certain regions.

- **Screen Shots of Galsa:**



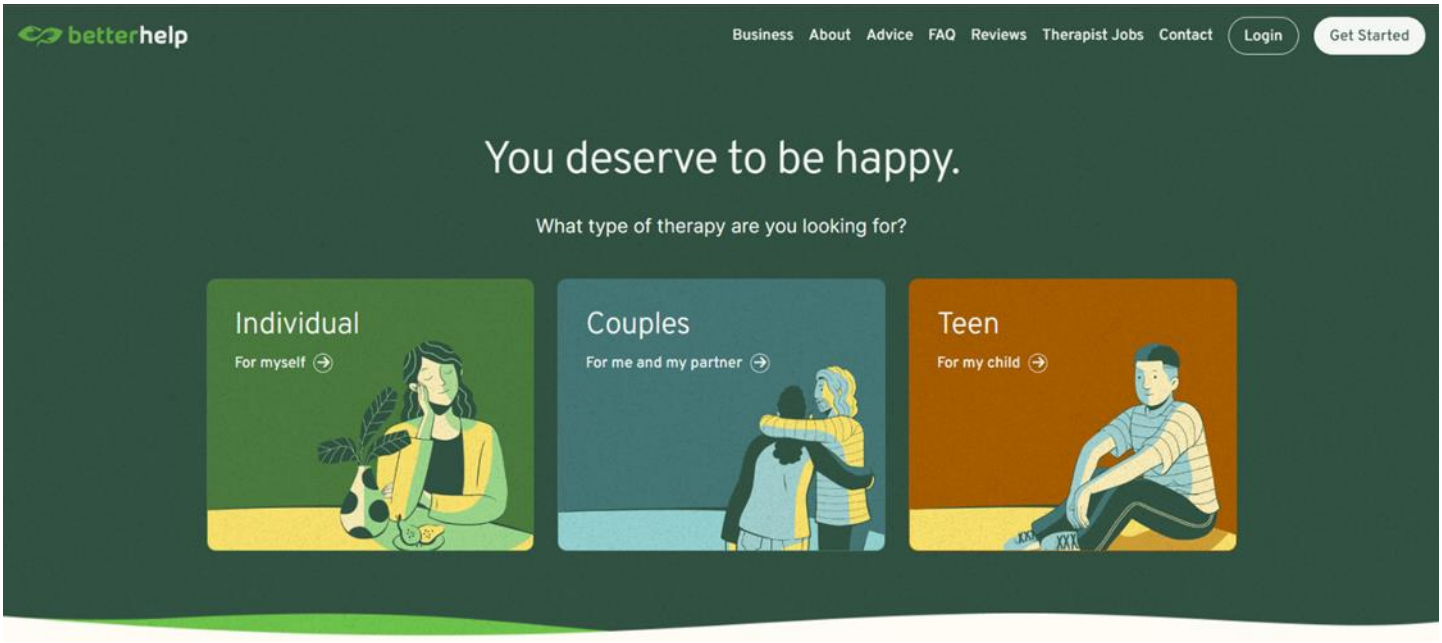
3. BetterHelp:

- **BetterHelp** is an online platform that provides mental health counseling and therapy services to individuals. It was founded in 2013 and is headquartered in Sunnyvale, California, BetterHelp offers a range of counseling and therapy services, including individual therapy, couples therapy, and counseling for teenagers, The platform connects individuals with licensed therapists through a secure online platform, which includes messaging, live chat, phone calls, and video conferencing, BetterHelp also offers a range of therapeutic approaches to address a variety of mental health issues, including depression, anxiety, stress, and relationship issues. The platform provides users with personalized counseling experience, matching them with a licensed therapist who specializes in their specific mental health concerns.

- **Advantages of BetterHelp:**
  1. Provided by a credentialled therapist.
  2. It contains tests for users entering the system.
  3. the idea of publishing articles for users.
  4. Chat, Phone, Video, and group sessions.
  5. Access therapy from anywhere.

- **Disadvantages of BetterHelp:**
  1. Dependence on technology.
  2. Reliance on internet connectivity.
  3. Potential for misdiagnosis.
  4. The ability to give rate for the doctor so for next users to know doctor's skills.

- **Screen Shot of BetterHelp:**



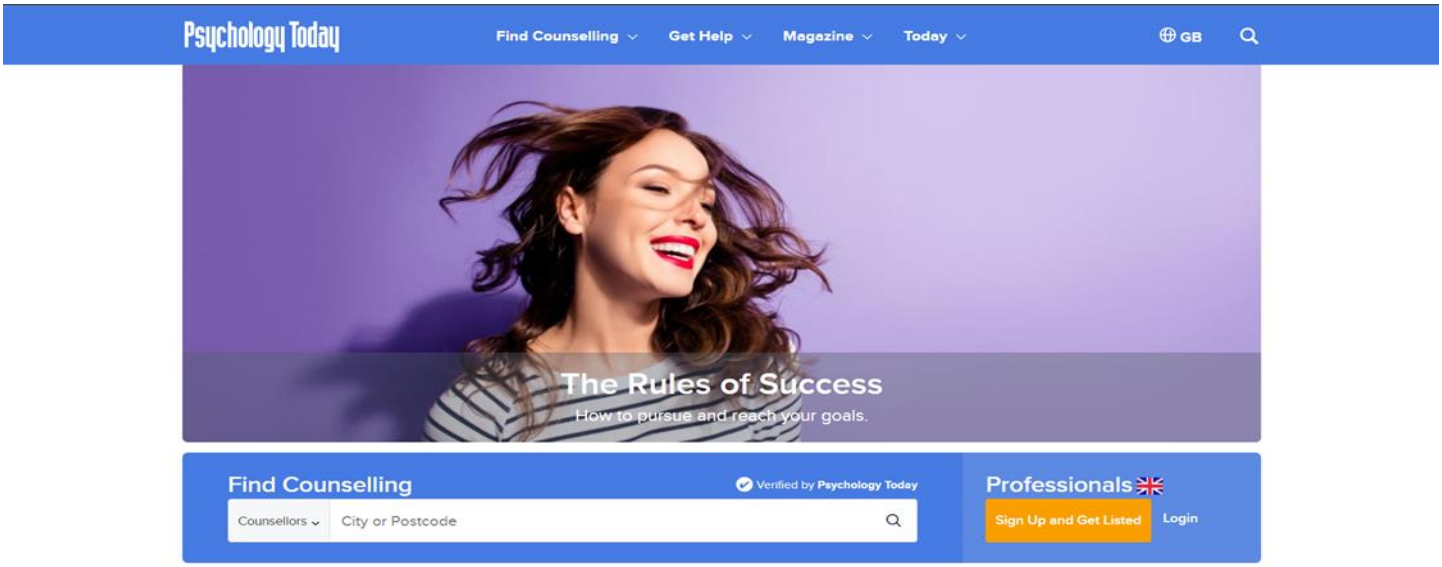
4. Psychology Today:

- Psychology Today is a valuable resource for individuals seeking information, support, and resources related to mental health and psychology. Its easy-to-use interface, comprehensive directory, and wealth of content make it a popular destination for individuals seeking to improve their mental health and wellness.

- **Advantages of Psychology Today:**
  1. Provides a wealth of information on a wide range of mental health and psychology-related topics.
  2. Offers a comprehensive directory of mental health professionals.
  3. Features an easy-to-use search tool that allows users to filter results based on location, specialty, insurance, and more.
  4. Offers self-help resources, such as quizzes and worksheets.
  5. The idea of publishing articles for users.

- **Disadvantages of Psychology Today:**
  1. Some of the information on Psychology Today may not be scientifically rigorous or evidence based.
  2. The quality of the mental health professionals listed in the directory can vary widely.
  3. The ability to give rate for the doctor so for next users to know doctor's skills.
  4. The platform does not provide a comprehensive assessment or diagnosis of mental health conditions.

- **Screen Shot of Psychology Today:**



2.3 Tools Survey:

1. Web Application tools:

1.1 Frontend Tools:

1.1.1 Figma: Figma is a web-based design and prototyping tool that allows designers to create user interfaces, websites, mobile apps, and other digital products. It's a popular design tool that is used by designers, developers, and other creatives to create high-quality designs and prototypes.

1.1 Why we used Figma:

1. Collaboration: Figma is a great tool for teams working together on design projects. It allows multiple people to work on the same project at the same time, which can save a lot of time and lead to better collaboration.

2. Cross-platform: Figma is a web-based tool, meaning it works on both Mac and PC, as well as on mobile devices. This makes it very versatile and accessible.

3. Prototyping: Figma has a built-in prototyping tool, which allows you to create interactive prototypes right within the app. This can save time and make it easier to communicate your design ideas to others.

4. Components: Figma has a powerful component system that allows you to create reusable design elements. This can save time and make it easier to maintain consistency across your designs.

1.2 Difference between Figma and Adobe XD:

1. Integration with other Adobe tools: If you're already using other Adobe tools like Photoshop or Illustrator, then Adobe XD will likely be a good fit for you. It integrates seamlessly with these other tools, making it easy to switch between them.

2. Native prototyping: Adobe XD has a built-in prototyping tool that allows you to create interactive prototypes right within the app. This can save time and make it easier to communicate your design ideas to others.

3. Design specs: Adobe XD has a feature called Design Specs, which allows you to easily share your designs with developers. This feature generates a web link where developers can view your designs and access all the information, they need to implement them.

4. Repeat Grid: Adobe XD has a powerful Repeat Grid feature that allows you to create repeating elements like lists or grids with ease.

1.1.2 React: is a free and open-source front-end JavaScript library for building user interfaces based on components. It is maintained by Meta and a community of individual developers and companies.

1.1 Why we use react:

1. It uses a component-based architecture, which allows developers to build modular and reusable components that can be used throughout an application. This can make development faster and more efficient and can also make code more maintainable and easier to debug.
2. It uses a virtual DOM, which is a lightweight representation of the actual DOM. This can help improve performance by minimizing the number of actual DOM manipulations needed when an application updates.
3. It uses a declarative syntax, which means that developers describe what they want an application to do, rather than how to do it. This can make code more readable and easier to reason about.
4. It has a large and active community of developers, which means that there are many resources available for learning and troubleshooting. This can be helpful for developers who are new to React or who are working on complex projects.
5. It can be used with other libraries and frameworks, such as Redux and React Native, which can help developers build full-stack web applications and mobile applications.
6. It is designed to be efficient and fast, which can help improve the performance of web applications.

1.2 Difference between React and Angular:

- React is a JavaScript library, whereas Angular is a TypeScript-based JavaScript framework. React uses one-way data binding and virtual DOM trees, whereas Angular uses two-way data binding and real DOM. Moreover, react is faster than Angular as it has a smaller bundle size.

	React JS	Angular JS
Advantage	The functions can be expanded in the future limitlessly.	It manages the dependencies automatically.
Demerit	Needs time to lay the groundwork.	The learning curve is steeper.
Recommended When	You are ready for a slow initiation of the development phase.	You need a guided and holistic framework to help with initiating the development.
	You want a UI oriented website with infinite customization.	A robust and fulfilling framework for your project
Popular Brands Using It	Facebook, Airbnb, Instagram, Whatsapp,Uber, Netflix, Dropbox	Google, Nike, General Motors, HBO, Forbes, Upwork, Sony

2. Backend tools

2.1. PHP: A Server-Side Scripting Language for Web Development.

2.1.1. Overview

PHP (Hypertext Preprocessor) is a widely used server-side scripting language designed for web development. This document serves as an introduction to PHP, providing an overview of its purpose, features, and applications in web development.

2.1.2. Why we use PHP:

1. Server-Side Scripting:

   - PHP is primarily executed on the server side, enabling the creation of dynamic web pages and web applications.

   - Server-side execution allows interaction with databases, file systems, and other server resources.

2. Integration with HTML:

   - PHP seamlessly integrates with HTML code, allowing the embedding of PHP scripts within HTML files.

   - This integration enables the mixing of dynamic PHP-generated content with static HTML content.

3. Database Connectivity:

   - PHP provides built-in support for various databases, including MySQL, PostgreSQL, Oracle, and more.

   - It allows connecting to databases, executing queries, and manipulating data, making it easy to create database-driven applications.

4. Web Application Development:

   - PHP offers a wide range of functionalities and frameworks for building web applications.

   - It provides features such as form processing, file handling, session management, and security mechanisms.

5. Extensive Library Support:

   - PHP has a vast collection of libraries and extensions available, providing additional functionalities and capabilities.

   - These libraries cover areas like image processing, data manipulation, API integrations, and more.

6. Ease of Use:

   - PHP has a simple and intuitive syntax, making it easy to learn and use, especially for beginners.

   - Its syntax is like C and other programming languages, making it familiar to developers with prior programming experience.

2.2 Laravel: A PHP Framework for Web Application Development

2.2.1. Overview

Laravel is an open-source PHP framework known for its elegant syntax, powerful features, and extensive ecosystem. This document serves as an introduction to Laravel, providing an overview of its key features, benefits, and use cases.

2.2.2. Why we use Laravel:

1. Model-View-Controller (MVC) Architecture:

    - Laravel follows the MVC architectural pattern, enabling developers to separate business logic, presentation, and data manipulation.

    - MVC enhances code organization, scalability, and maintainability.

2. Routing:

    - Laravel offers a concise and expressive routing system.

    - Developers can define application URLs and map them to corresponding controllers and action

3. Database Abstraction:

    - Laravel's Eloquent ORM (Object-Relational Mapping) provides a simple yet powerful way to interact with databases.

    - Eloquent supports query building, active record implementation, relationships, and database migrations.

4. Blade Templating Engine:

    - Laravel employs the Blade templating engine, offering a clean and intuitive syntax for designing views.

    - Blade allows for efficient separation of presentation logic from application logic.

5. Artisan CLI:

    - Laravel provides a command-line interface called Artisan.

    - Artisan offers a wide range of useful commands for code generation, database management, and other common development tasks.

6. Authentication and Authorization:

- Laravel simplifies user authentication and authorization.

- It includes built-in features for user registration, login, password reset, and role-based access control.

7. Caching and Performance:

- Laravel supports multiple caching mechanisms, including popular backends like Memcached and Redis.

- Caching improves application performance by storing frequently accessed data.

2.2.3. Combined Benefits of PHP and Laravel

- When used together, PHP and Laravel offer a range of advantages, including:

- Rapid development through PHP's flexibility and Laravel's pre-built functionalities.

- Scalability and maintainability through Laravel's MVC architecture and code organization.

- Database connectivity and integration capabilities of PHP combined with Laravel's ORM and database migration tools.

- Access to a large community and ecosystem that provide ongoing support, resources, and updates for both PHP and Laravel.

2.3. Comparison with Other Languages and Tools

3.1. PHP with Custom Code:

- PHP without a framework allows complete customization and flexibility.

- However, it requires developers to build functionalities from scratch and manage code organization independently.

3.2. Other Server-Side Languages (Python, Ruby, etc.):

- Languages like Python and Ruby offer alternative server-side options for web development.

- While they provide their own frameworks and libraries, PHP with Laravel often excels in terms of community support, ecosystem, and ease of use.

3.3. JavaScript and Front-End Frameworks (React, Angular, etc.):

- JavaScript and front-end frameworks focus on client-side development.

- PHP with Laravel complements these frameworks by serving as the backend, handling server-side operations, and integrating with front-end technologies.

3.4. Content Management Systems (WordPress, Drupal, Joomla, etc.):

- Content management systems (CMS) provide pre-built platforms for website creation and content management.

- PHP with Laravel, on the other hand, offers more flexibility and customization options, making it suitable for complex web applications beyond traditional CMS-based websites.

3. Database Tools:

- **What is the data:**
  In general, data usually refers to digital information that can be stored and processed using computers. This can include data entered by users on websites or applications, data generated by sensors and devices, and data collected from various sources such as social media, web analytics, and customer feedback.

- **Types of data:**
  - Structured data
  - Semi-structured data
  - Unstructured data

Technology	**Relational database table**	**XML / RDF**	**Character and binary data**
Transaction management	Matured transaction, Various concurrency techniques	Transaction management adapted from RDBMS not matured	No transaction management, no concurrency
Version management	Versioning over tuples, rows, tables, etc.	Versioning over tuples or graphics is possible	Versioned.
Flexibility	Schema dependent rigorous schema	Flexible, tolerant schema.	Very flexible, absence of schema.
Scalability	Scaling DB schema is difficult.	Schema scaling is simple.	Very scalable
Robustness	Very robust	New technology not widely spread	----

- **Why we use MySQL** is an open-source relational database management system (RDBMS) that uses structured query language (SQL) to manage and manipulate data. It is one of the most widely used databases in the world, with a large community of developers and users contributing to its development.

  Here are some of the key features and benefits of MySQL:

  - It is free and open source, which means that it can be downloaded and used without any licensing fees.
  - It is a relational database, which means that data is organized into tables with columns and rows, making it easy to manage and manipulate data.
  - It is designed to be scalable and can handle large amounts of data and high traffic websites.
  - It is optimized for speed and can quickly retrieve and manipulate data, making it ideal for applications that require fast response times.
  - It includes built-in security features, such as encryption and user authentication, to protect sensitive data from unauthorized access.
  - It is compatible with a wide range of platforms and programming languages, making it easy to integrate with existing systems and applications.
  - It has a large and active community of developers and users, which means that there are many resources available for learning and troubleshooting.

4. Mobile Application Tools:

4.1 Kolten: is a cross-platform, statically typed, general-purpose high-level programming language with type inference. Kotlin is designed to interoperate fully with Java, and the JVM version of Kotlin's standard library depends on the Java Class Library, but type inference allows its syntax to be more concise.

4.1.1 Why we use Kolten:

1. It is designed to be a concise and expressive language, allowing developers to write code in fewer lines than traditional programming languages. This can help improve productivity and reduce the likelihood of errors.
2. It is designed to be interoperable with existing Java code, which means that developers can use both languages together in the same project. This can be helpful for migrating legacy Java code to Kotlin or for integrating Kotlin into existing Java projects.
3. It is a statically typed language, which means that the compiler checks the types of variables and expressions at compile time. This can help catch errors earlier in the development process, reducing the likelihood of runtime errors.
4. It has built-in null safety features that can help prevent null pointer exceptions. This can be helpful for writing more robust and stable code.
5. It includes modern language features such as lambdas, extension functions, and coroutines, which can make it easier to write code that is more readable, maintainable, and scalable.
6. It allows developers to access platform-specific features such as Android SDKs and APIs, which can be important for building high-quality Android applications.

4.1.2 Difference between Kolten and Flutter:

	Flutter	Kotlin
Performance	App speed is fast because of the Hot Reload functionality.	Apps build with Kotlin are efficient, secure, optimal and perform better.
Popularity	Flutter is gaining almost near popularity as Kotlin, despite being very new.	Kotlin is trending as compared to Flutter on Google trends.
Community	Flutter has more substantial community support on Github	Kotlin way lacks beat the Flutter community on Github
UI Experience	User can build a customer user interface with Flutter.	With Kotlin, users can have their native experience.
Testing Support	It works as a B-a-a-S and provides some in-built widget tasting and app testing features.	It provides a rich set of unit testing functionality for Android apps.
Intergration	Flutter integrates well with the Android and iOS platforms.	Kotlin provides multiplatform availability for every native environment.
Minimum Support Platform	Android v4.1x+, v10(jelly bean), v16, and iOS 8+	Kotlin support seven the older version of Android and iOS 8+

5. Chatbot and Data analysis Tools:

5.1. Python:

5.1.1 Why we use Python:

1. Readability: Python's syntax is designed to be easy to read and understand. Its clean and visually uncluttered code layout, along with the use of indentation for code blocks, promotes readability and enhances code comprehension.
2. Simplicity: Python emphasizes simplicity and minimalistic design. It aims to provide a straightforward and intuitive programming experience, making it accessible for beginners while still being powerful enough for complex applications.
3. Interpreted Language: Python is an interpreted language, which means that it is executed line by line, without the need for explicit compilation. This allows for quick development and testing cycles, as code changes can be immediately executed and evaluated.
4. Object-Oriented Programming (OOP): Python supports object-oriented programming principles, allowing developers to create classes, objects, and utilize inheritance, polymorphism, and encapsulation. OOP enables modular and reusable code organization, making it easier to manage complex projects.
5. Large Standard Library: Python comes with a comprehensive standard library that provides a wide range of modules and packages covering various functionalities. These modules offer ready-to-use tools for tasks such as file I/O, networking, web development, data manipulation, and more, significantly reducing the need for external dependencies.

6. Cross-Platform Compatibility: Python is a cross-platform language, meaning that Python code can run on different operating systems (such as Windows, macOS, and Linux) without modification. This portability makes it easy to develop applications that can be deployed on multiple platforms.

7. Extensive Ecosystem: Python has a vast ecosystem of third-party libraries and frameworks. These libraries cover a wide range of domains, including scientific computing (NumPy, Pandas), machine learning (Scikit-learn, TensorFlow), web development (Django, Flask), and more. The extensive ecosystem provides developers with a wealth of tools and resources to accelerate development and solve complex problems.

Python's versatility and ease of use make it suitable for a wide range of applications, including web development, data analysis, scientific computing, machine learning, automation, scripting, and more. Its community-driven development, active user base, and extensive documentation contribute to its popularity and make Python a powerful and flexible programming language.

5.2 Streamlit:

5.2.1 Here are some key aspects and features of Streamlit:

1. Simplified Web Development: Streamlit abstracts away the complexities of web development, allowing you to focus on the logic and functionality of your application. With Streamlit, you can create web apps using only Python code, without the need to write HTML, CSS, or JavaScript. This simplicity makes it accessible to developers with varying levels of web development experience.

2. Interactive User Interfaces: Streamlit provides a variety of interactive UI components that can be easily integrated into your application. These components include buttons, sliders, checkboxes, and text inputs, among others. By leveraging these UI components, you can create interactive menus and forms for your users to interact with your application. This interactive nature enhances the user experience and enables seamless communication with your Streamlit app.

3. Rapid Prototyping and Iterative Development: Streamlit's live reload functionality enables real-time updates in the browser as you modify your code. This feature allows for rapid prototyping and iterative development, as you can instantly see the effects of your changes without the need for manual page refreshes. This streamlined workflow boosts productivity and accelerates the development process.

4. Data Visualization Integration: Streamlit seamlessly integrates with popular data visualization libraries such as Matplotlib, Plotly, and Altair. This integration allows you to create dynamic and interactive visualizations directly within your Streamlit app. You can easily generate charts, graphs, and plots to visualize data and present it in a compelling manner to your users.

5. Deployment and Sharing: Streamlit simplifies the deployment process, enabling you to easily share your web app with others. With just a few lines of code, you can deploy your Streamlit app on platforms like Heroku or host it on your own server. This deployment flexibility makes it effortless to share your application with collaborators, stakeholders, or a wider audience.

Streamlit's intuitive Python API, interactive UI components, rapid prototyping capabilities, seamless data visualization integration, and simplified deployment make it an excellent choice for building menu button-based chatbots or any other data-driven web application. By leveraging Streamlit's strengths, you can create engaging and interactive applications that provide a seamless user experience while minimizing the complexities of traditional web development.

- **Why did we use Python and Streamlit?**
    - Python and Streamlit are ideal choices for building a menu button-based chatbot due to several reasons:

- **Python for Chatbot Development:**

1. Simplicity: Python has a simple and readable syntax, making it easy to understand and write code. This simplicity accelerates the development process and reduces the chances of errors, making it an excellent choice for chatbot projects.
2. Vast Ecosystem: Python has a vast ecosystem of libraries and frameworks that cater specifically to natural language processing (NLP) and machine learning tasks. These libraries provide pre-built algorithms and models for tasks such as language understanding, sentiment analysis, and text generation, enabling developers to leverage existing resources and accelerate development.
3. NLP and ML Capabilities: Python's robust NLP libraries, such as NLTK, SpaCy, and TensorFlow, offer advanced functionalities for processing and analyzing natural language. These capabilities enable the chatbot to understand and generate human-like responses, improving the overall user experience.

- **Streamlit for Menu Button-Based Chatbot Development:**

1. Simplified Web Development: Streamlit simplifies the process of building web applications by providing a user-friendly Python API. It eliminates the need to write complex HTML, CSS, or JavaScript code, allowing developers to focus on the chatbot's logic and functionality.
2. Interactive User Interfaces: Streamlit offers a variety of interactive UI components, including buttons, sliders, and checkboxes, which can be easily integrated into the chatbot. These components enable users to interact with the chatbot through menus and buttons, providing a user-friendly and intuitive interface.
3. Rapid Prototyping: Streamlit's live reload functionality allows for real-time updates in the browser as you modify the code. This feature speeds up the development process by providing immediate feedback, making it easier to prototype and iterate on the chatbot's features and design.
4. Data Visualization: Streamlit seamlessly integrates with data visualization libraries such as Matplotlib and Plotly. This integration allows developers to visualize chatbot responses, display graphs, and present data in an interactive and visually appealing manner.
5. Deployment and Sharing: Streamlit simplifies the deployment process, enabling you to easily deploy and share your menu button-based chatbot as a web application. Whether you choose to deploy it on platforms like Heroku or host it on your own server, Streamlit provides straightforward options for sharing your chatbot with others.

By combining the strengths of Python and Streamlit, developers can create a menu button-based chatbot that leverages Python's powerful NLP and machine learning capabilities while benefiting from Streamlit's simplicity, rapid prototyping features, interactive UI components, and easy deployment options. This combination results in an efficient and user-friendly chatbot application that provides an intuitive menu-driven conversation experience.

- **Difference Between Python with Streamlit and Other tools:**

1. **Python with Streamlit:**

- Python is a versatile programming language known for its simplicity and readability.
- Streamlit is a Python library specifically designed for building interactive web applications.
- Difference: Python, as a general-purpose language, provides a wide range of functionalities and libraries that can be used to develop menu button-based chatbots. Streamlit, on the other hand, simplifies the web development process by offering a user-friendly API and pre-built UI components that facilitate the creation of interactive interfaces for your chatbot.
- Use Case: Python with Streamlit is an excellent choice for developing menu button-based chatbots due to Python's versatility and Streamlit's simplified development and interactive UI components. It allows you to create an intuitive and user-friendly interface with menu buttons to drive the conversation flow of your chatbot.

2. HTML/CSS/JavaScript with JavaScript Frameworks (e.g., React, Vue.js):

- HTML, CSS, and JavaScript are the core technologies used for web development.
- JavaScript frameworks like React and Vue.js provide powerful tools for building interactive web applications.
- Difference: Unlike Python and Streamlit, which are primarily focused on the backend development, using HTML/CSS/JavaScript and JavaScript frameworks gives you full control over the frontend development of your chatbot. You can design and customize the user interface, including menu buttons, to create a highly tailored and visually appealing chatbot experience.
- Use Case: HTML/CSS/JavaScript with JavaScript frameworks are suitable when you need fine-grained control over the front-end design and behavior of your chatbot. If you have specific UI/UX requirements or want to create a highly customized user interface, these technologies provide flexibility and extensive customization options.

3. Chatbot Development Platforms (e.g., Chatfuel, ManyChat):

- Chatbot development platforms offer drag-and-drop interfaces and pre-built components to create chatbot applications.
- Difference: Chatbot development platforms are designed to simplify the chatbot development process by providing a visual interface and pre-built functionalities. They typically offer menu button options and other interactive elements to create chatbot flows without writing code.
- Use Case: Chatbot development platforms are suitable when you want to quickly create a menu button-based chatbot without extensive coding or web development knowledge. These platforms offer simplicity and ease of use, making them ideal for beginners or non-technical users who want to build chatbots with menu-driven interactions.

4. **Microsoft Bot Framework Composer:**

- The Microsoft Bot Framework Composer is a visual development tool for building chatbots.
- Difference: The Bot Framework Composer provides a visual interface that allows you to design and create conversational flows using a graphical editor. It offers a drag-and-drop interface for defining conversation paths, managing user input, and integrating with external services. The Composer supports menu button-based interactions, enabling you to create chatbots with intuitive menu-driven navigation.
- Use Case: The Bot Framework Composer is suitable for developers who prefer a visual approach to building chatbots. It offers a low-code development environment and simplifies the process of creating menu-driven chatbots. The Composer also integrates with Microsoft services, making it a good fit for those leveraging the Microsoft ecosystem.

5. **Twilio Autopilot:**

- Twilio Autopilot is a conversational AI platform for building chatbots and virtual assistants.
- Difference: Autopilot provides a platform for creating conversational experiences using a combination of natural language understanding (NLU), dialogue management, and task fulfillment. It supports menu button-based interactions using predefined tasks and system actions. You can design the chatbot's menu options and responses using the Autopilot interface.
- Use Case: Twilio Autopilot is suitable for developers who want to leverage a powerful conversational AI platform for building menu-driven chatbots. It provides robust NLU capabilities, allowing you to create chatbots that understand user intents and respond accordingly. Autopilot's flexibility and integration options make it a good fit for various industries and use cases.

Each of these tools has its own advantages and considerations. Choosing the right tool depends on factors such as your development expertise, desired level of customization, and the specific requirements of your menu button-based chatbot. Consider the simplicity, customization options, and development capabilities of each tool to make an informed decision.

Chapter Three:
System Specification

3.1 System Description:

3.1.1 Functional Requirements:

1. User:

1. Registration:

- Passing through the "Welcome Page" The Lawyer will be directed to a page where he is asked to enter the following (First Time):
    - e-mail address
    - Full name
    - Password
    - Phone number

  Then click on Register after that a number is sent to his email to verify who he is, and he is asked to write that number in the application.

2. Login:

- Passing through the "Welcome Page" The Lawyer will be directed to a page where he is asked to enter the following:
    - username/email
    - password Click.

  After login he directed to his home page after checking the username/email and password.

3. View contact us:

- Where the user can say his opinion about the process and give us feedback.

4. View FAQs:

- Where user can see what other users asked about and learn more about system.

5. View / search for articles:

- Where users can see our coaches' articles and know more about life coaching.

6. Fill information before test:

- Where we take more information from the user that we'll need before he takes the test.

7. View / take test:

- Where user can take Lüscher test by choosing colors and show out his preferences.

8. View test results:

- Where user can see coach's opinion about user's choices and diagnose his case.

9. Schedule appointment:

- Where the user can choose the date and time for his appointment with doctor.

10. Test payment:

- Where the user can pay for his journey with us.

2. Coach:

1. **Login:**
    - Passing through the “Welcome Page” The Lawyer will be directed to a page where he is asked to enter the following:
        - username/email
        - password Click

      on login then he is directed to his home page after checking the username/email and password.
2. **View her schedule:**
    - Where the coach can see his appointments for the week.

3. View patient data:

- Where coaches can see patient test results and data.

4. Add / edit articles:

- Where doctor can add articles for patients or edit on previous articles

5. View dashboard:

- Where the coach can see statistics and insights about his patients.

3. Secretary:

1. Login:

- Passing through the “Welcome Page” The Lawyer will be directed to a page where he is asked to enter the following:
    - username/email
    - password Click

  on login then he is directed to his home page after checking the username/email and password.

2. Manage appointments:

- Add appointments.
- Delete appointments.
- Edit appointments.
- Search for appointments.

3. View patient data.

4. Change appointments prices.

3.1.2 Non-Functional Requirements:

1. Performance Requirements

- Coaches don’t have to deal with customers offline.
- Coaches can check the user’s result at any time.
- Speed up the workflow and reduce the time required to finish the case.

2. Availability

- Our system is available to any customer to be treated online.
- Our system is expandable to support future developments and additions to it
- Our system is portable for all web applications and android operating systems from 5.1 to present.

3. Reliability

- Our system is operable in all lighting conditions. Regardless of brightness level in the user's operating environment

4. Usability

- Our system is easy to use for all users with minimal instructions. 100% of Languages in the Graphical User Interface (GUI) should be intuitive and it can be understood by non-technical users.

5. Correctness:

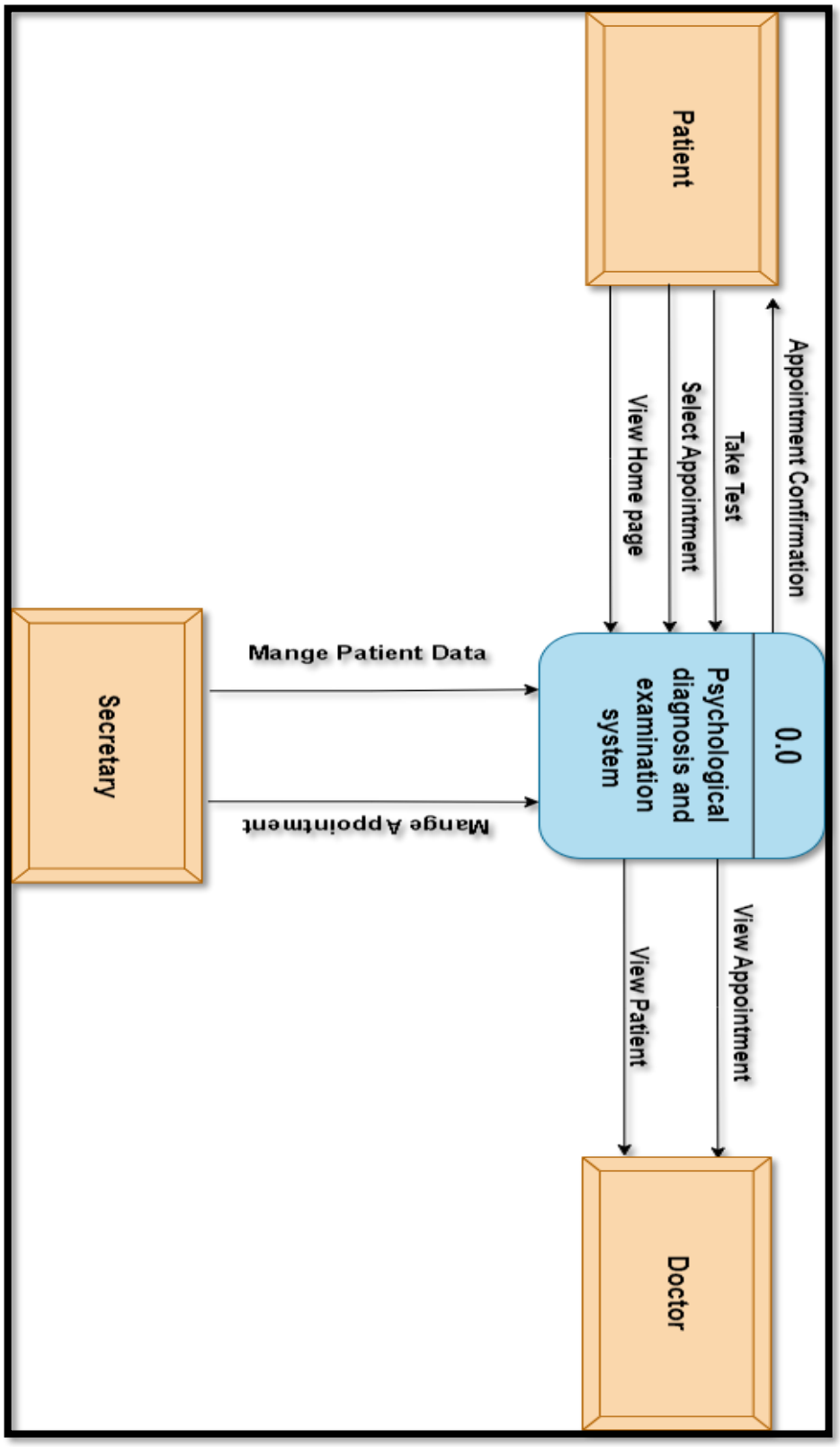
- Our system achieves all the goals of coaches and facilitates their work.

6. Instability

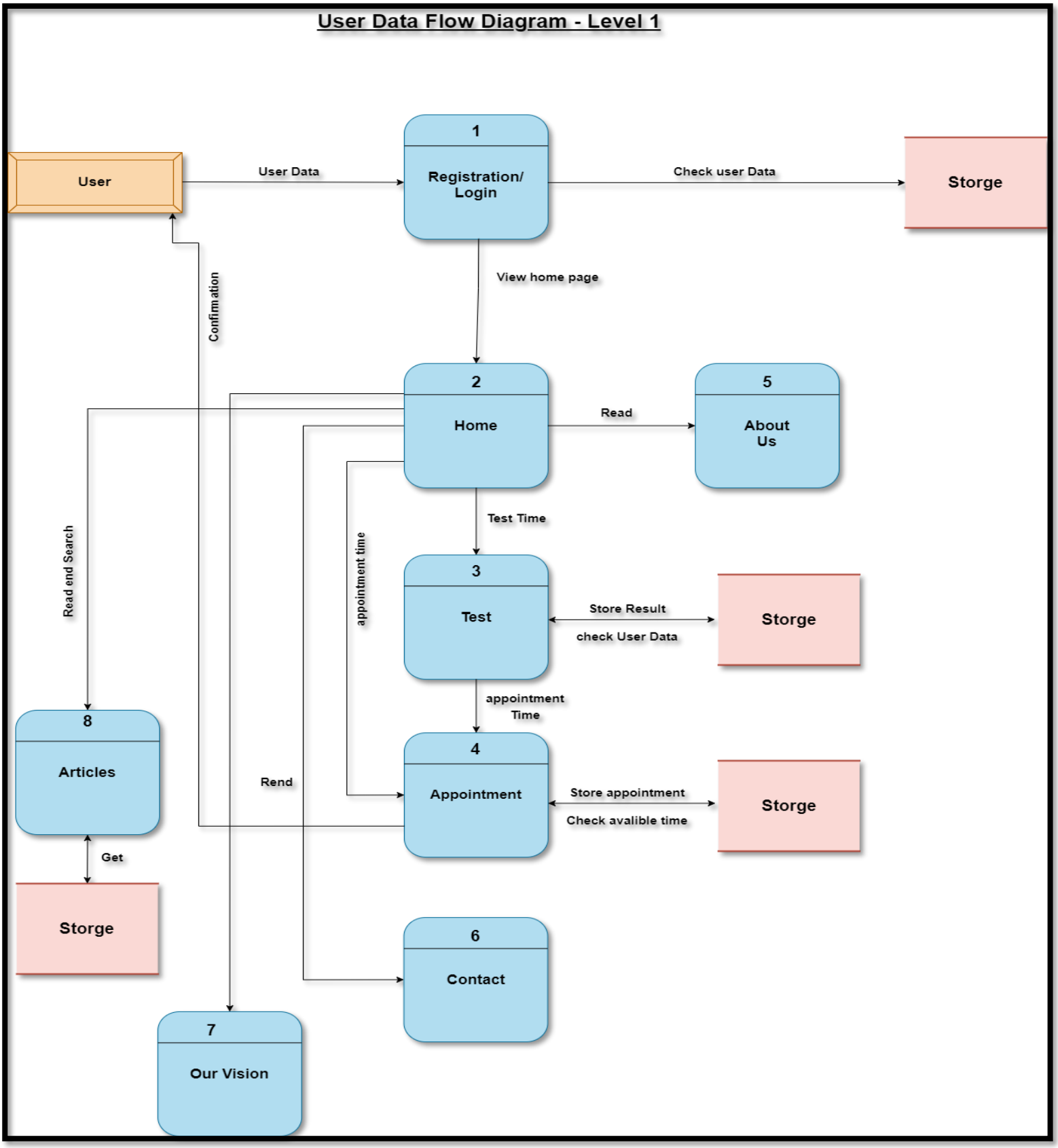
- Our system can be easily installed on most devices if the basic requirements are met.

3.2 Context and DFD Diagram:

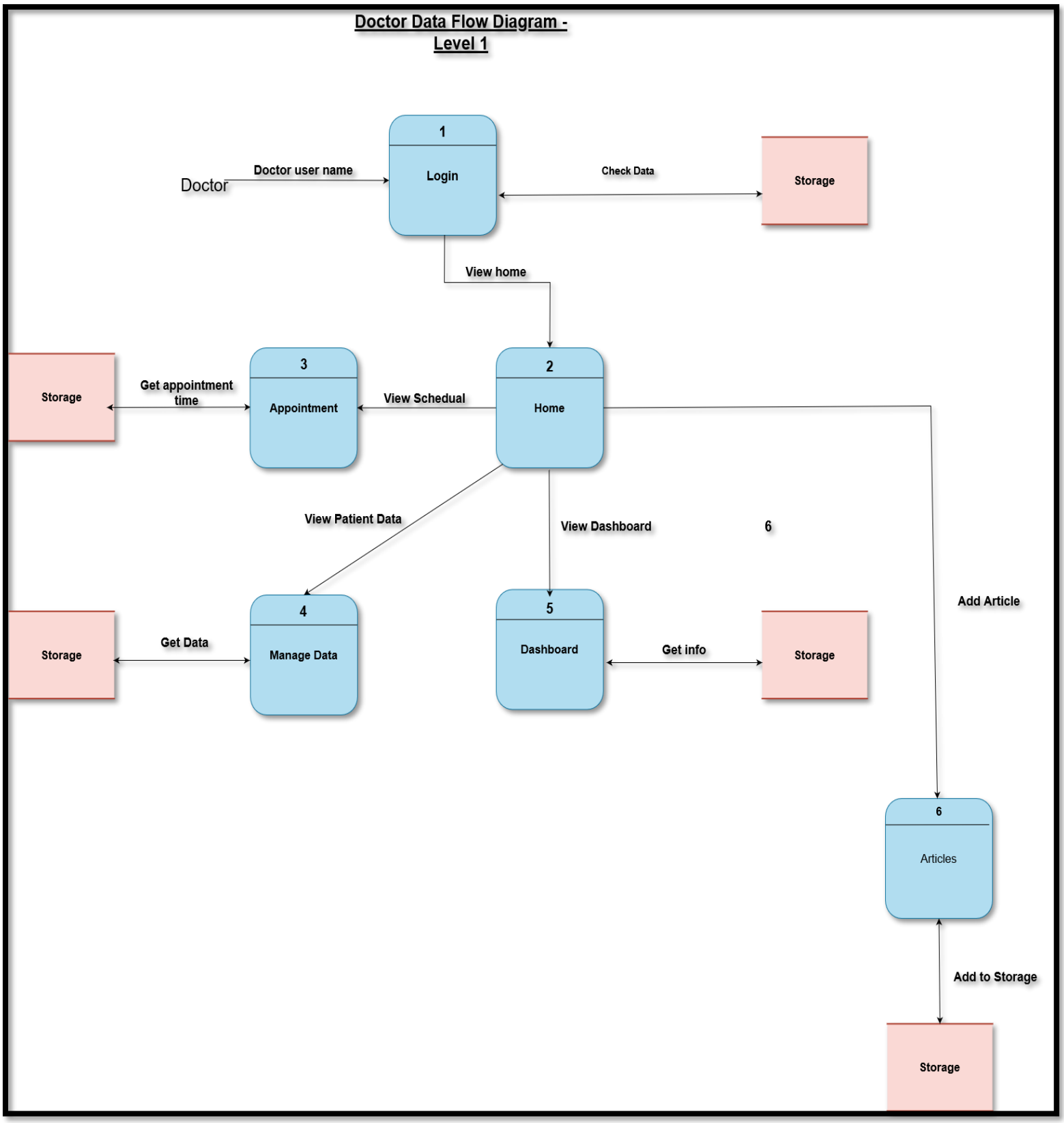
3.2.1 Context Diagram:



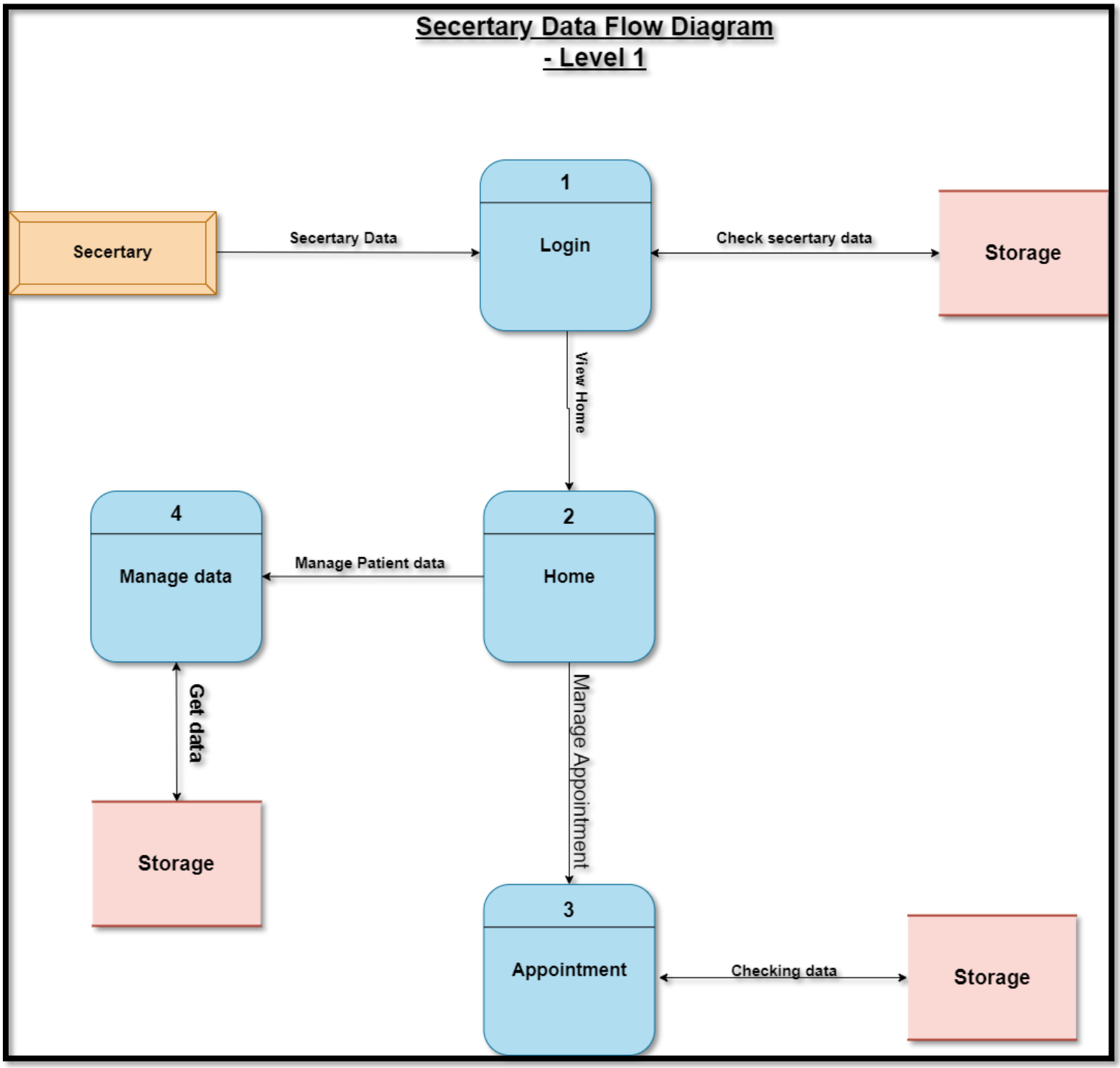
3.2.2 DFD for (use):



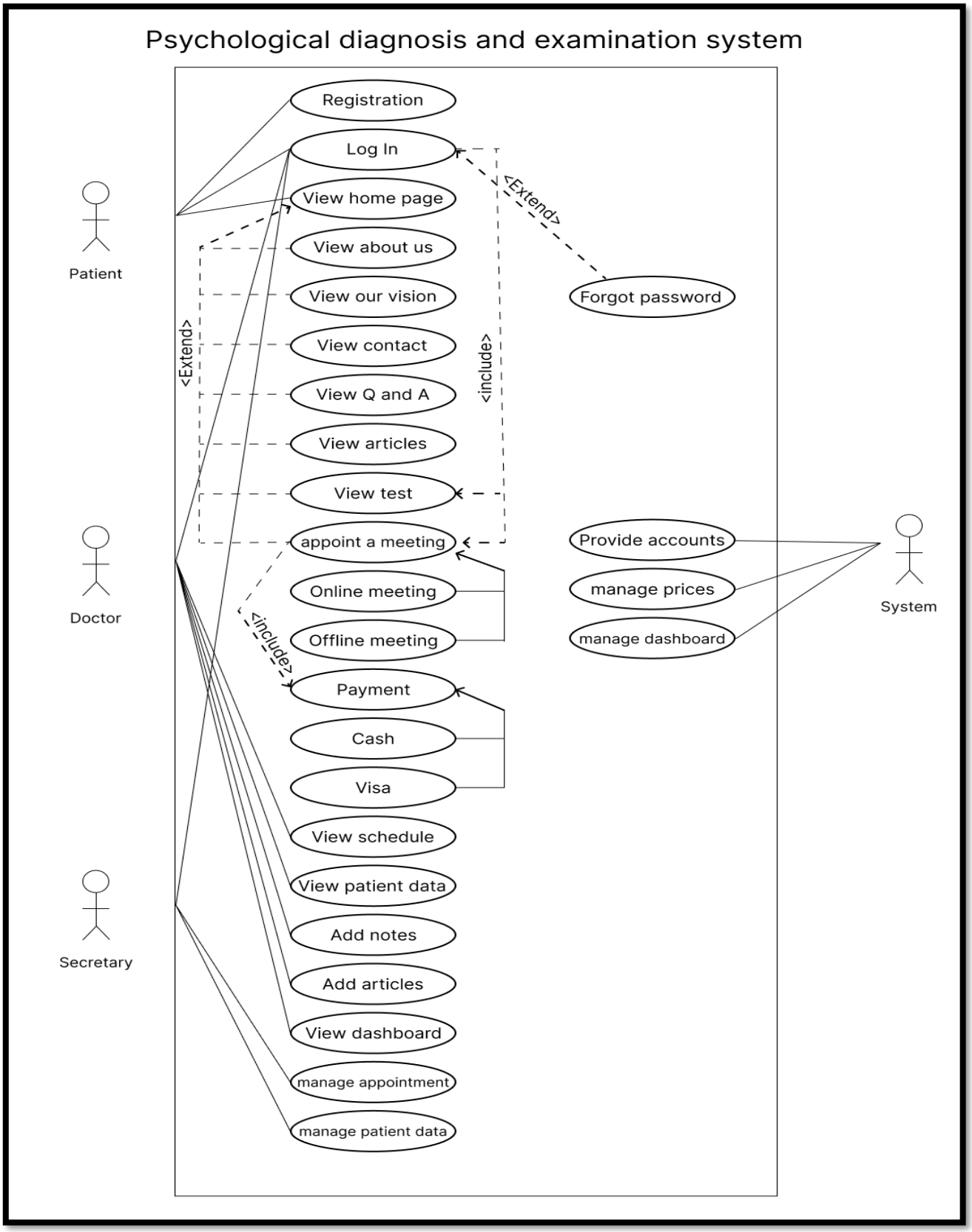
3.2.3 DFD for (Doctor):



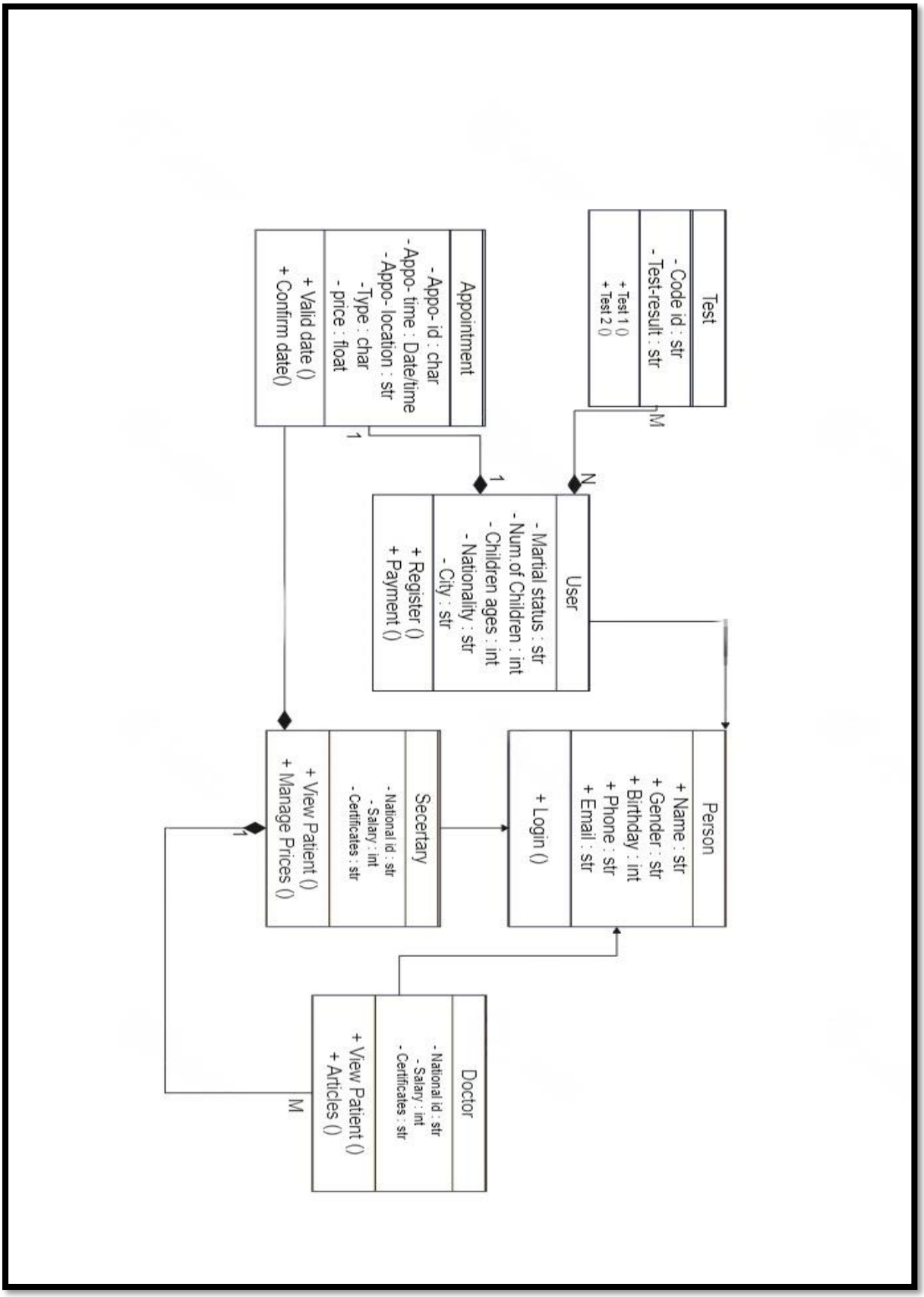
3.2.4 DFD for (Secretary):



3.3 System Models:
3.3.1 Use Case:

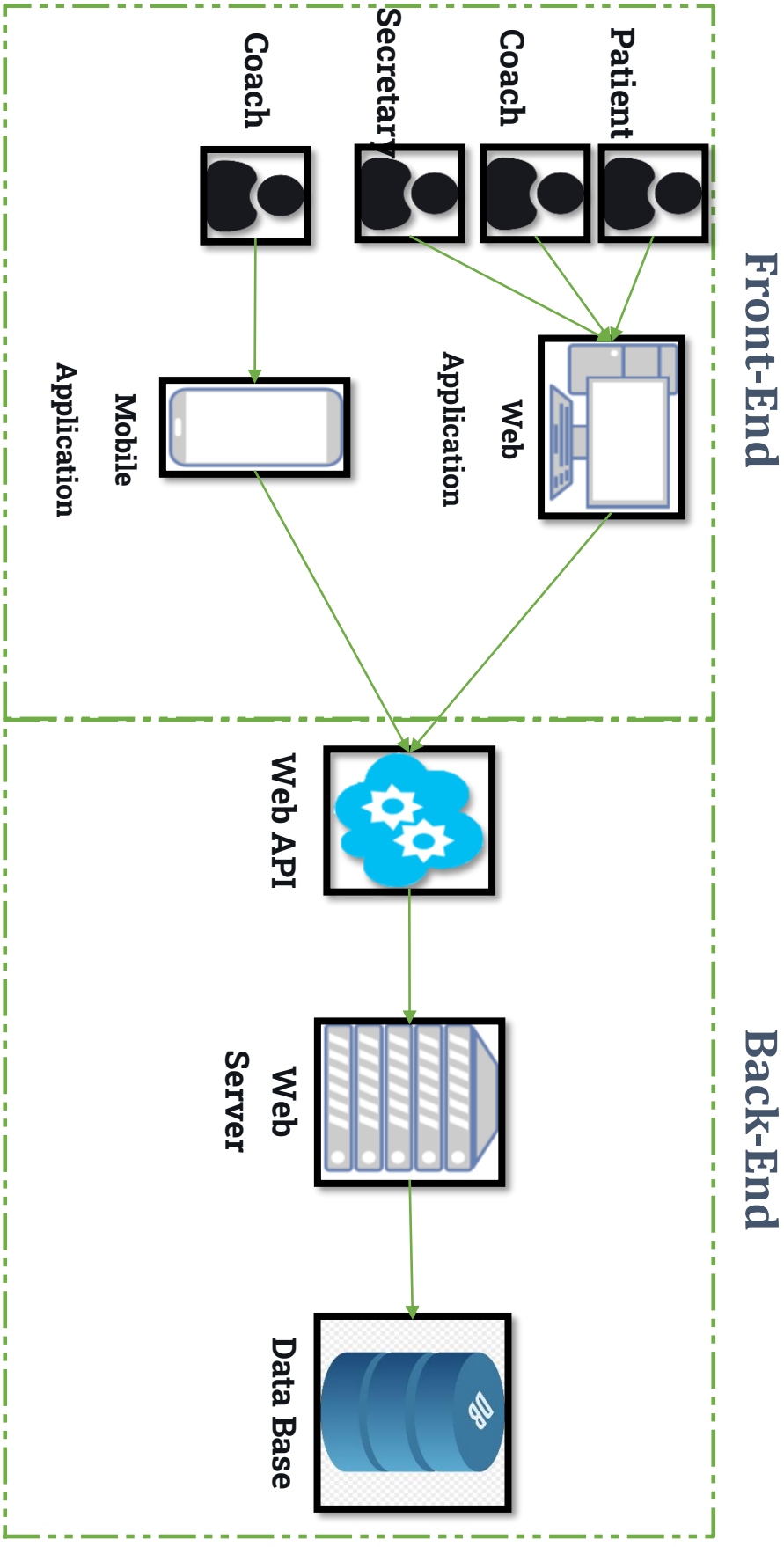


3.3.2 Class Diagram:



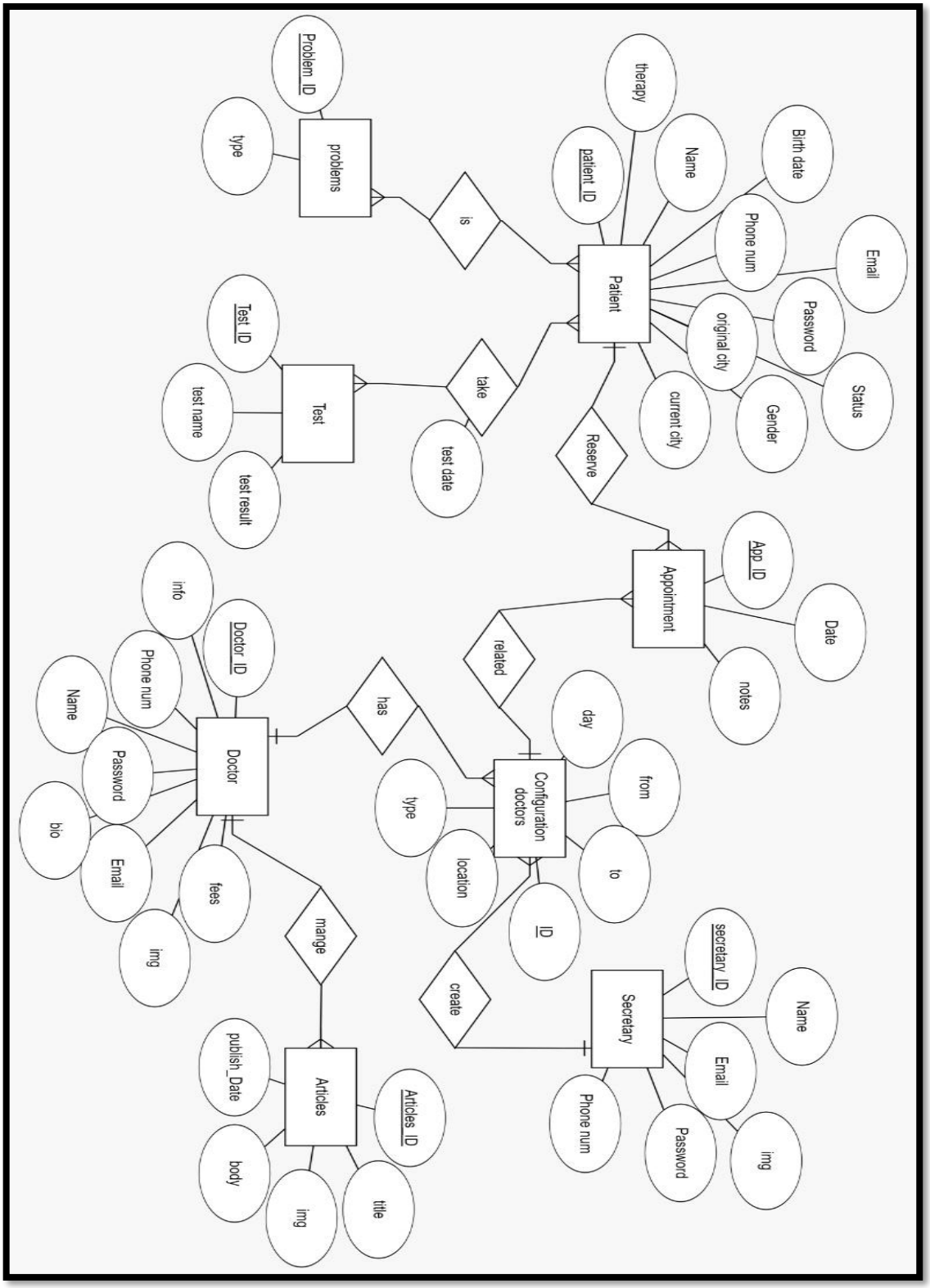
Chapter Four:
System Design

4.1 System Architecture:

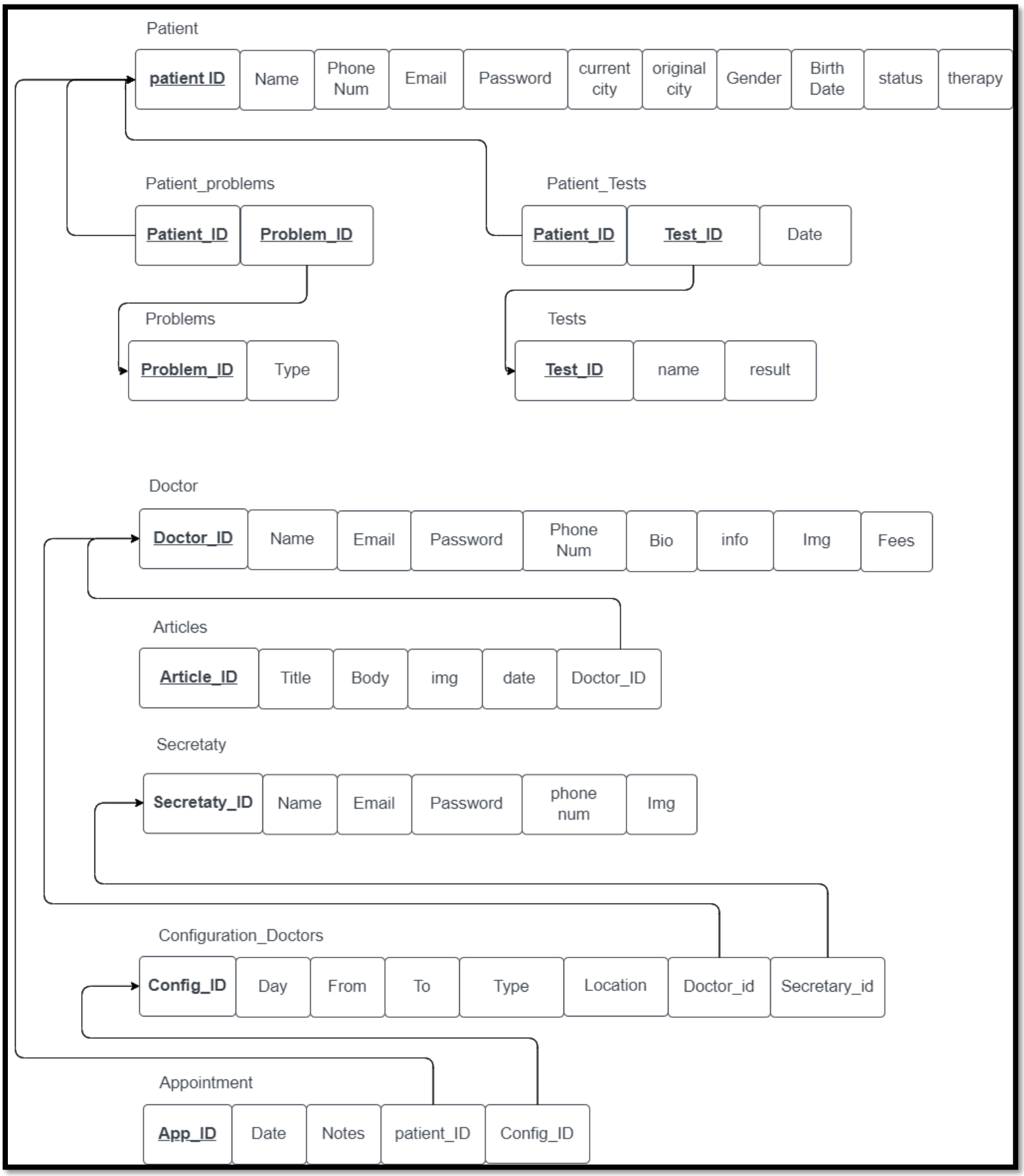


4.2 Database Design:

2.1 ERD diagram:

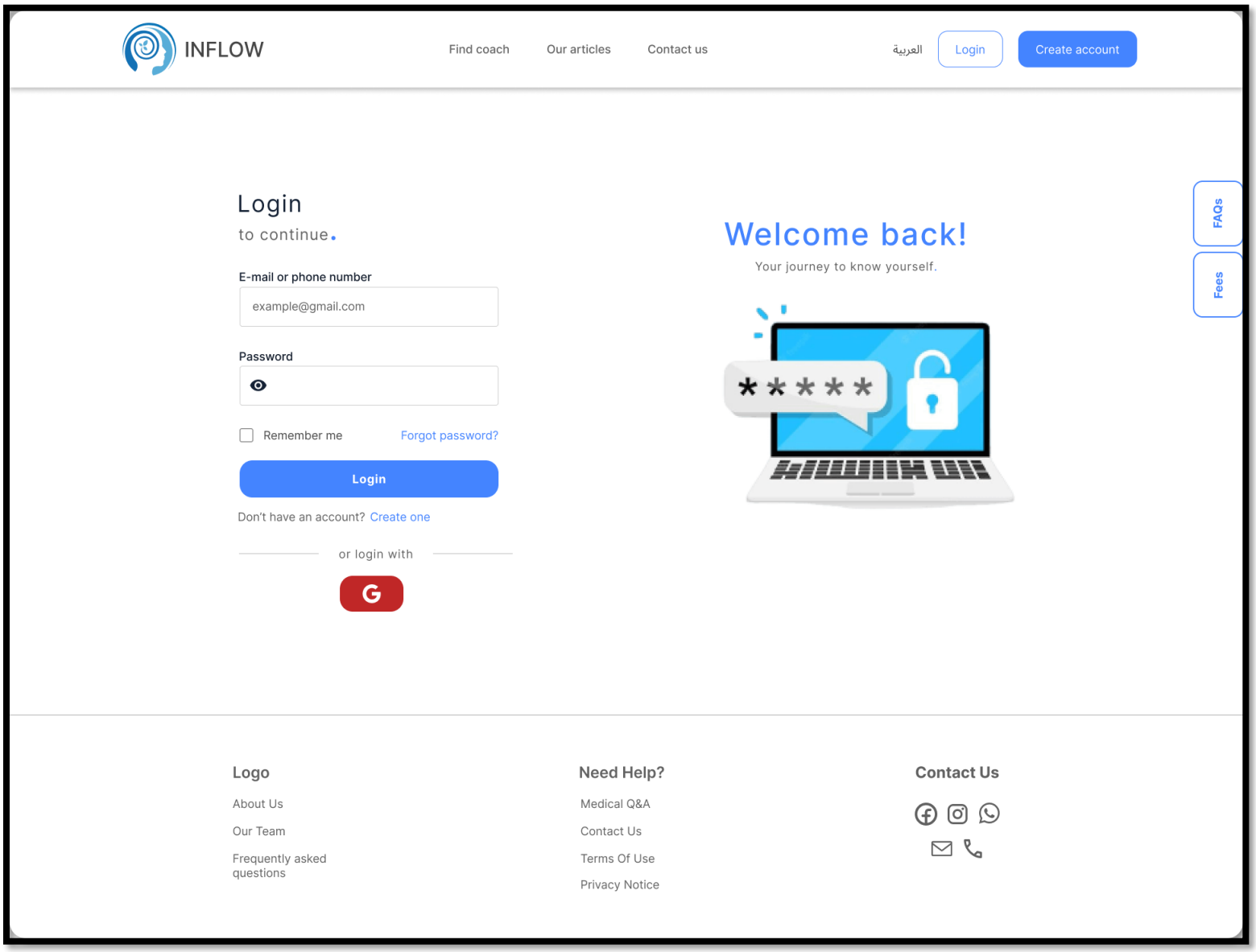


2.2 Schema

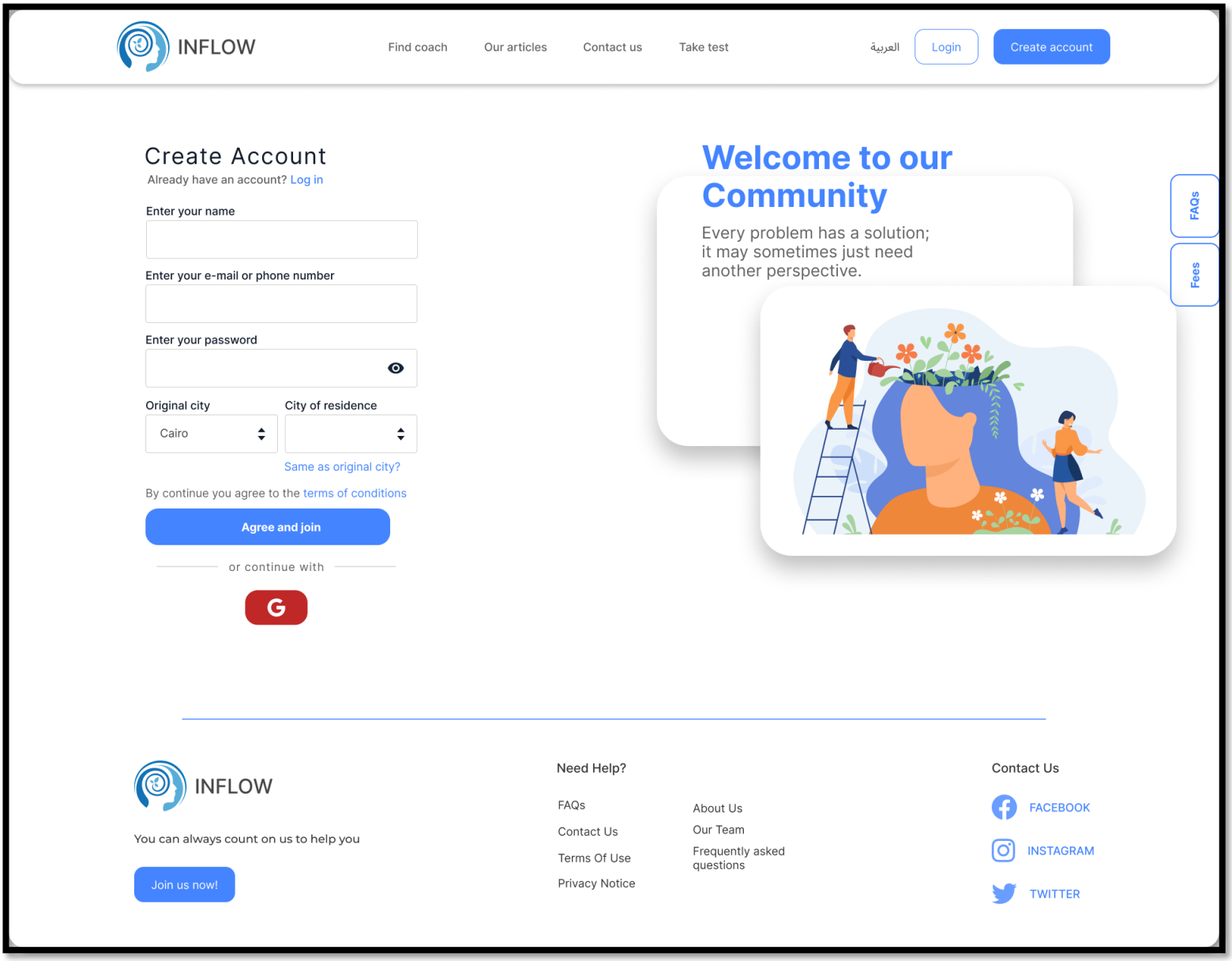

Patient
patient ID
Name
Phone Num
Email
Password
current city
original city
Gender
Birth Date
status
therapy
Patient_problems
Patient_ID
Problem_ID
Patient_Tests
Patient_ID
Test_ID
Date
Problems
Problem_ID
Type
Tests
Test_ID
name
result
Doctor
Doctor_ID
Name
Email
Password
Phone Num
Bio
info
Img
Fees
Articles
Article_ID
Title
Body
img
date
Doctor_ID
Secretaty
Secretaty_ID
Name
Email
Password
phone num
Img
Configuration_Doctors
Config_ID
Day
From
To
Type
Location
Doctor_id
Secretary_id
Appointment
App_ID
Date
Notes
patient_ID
Config_ID

4.3 User Interface Design:

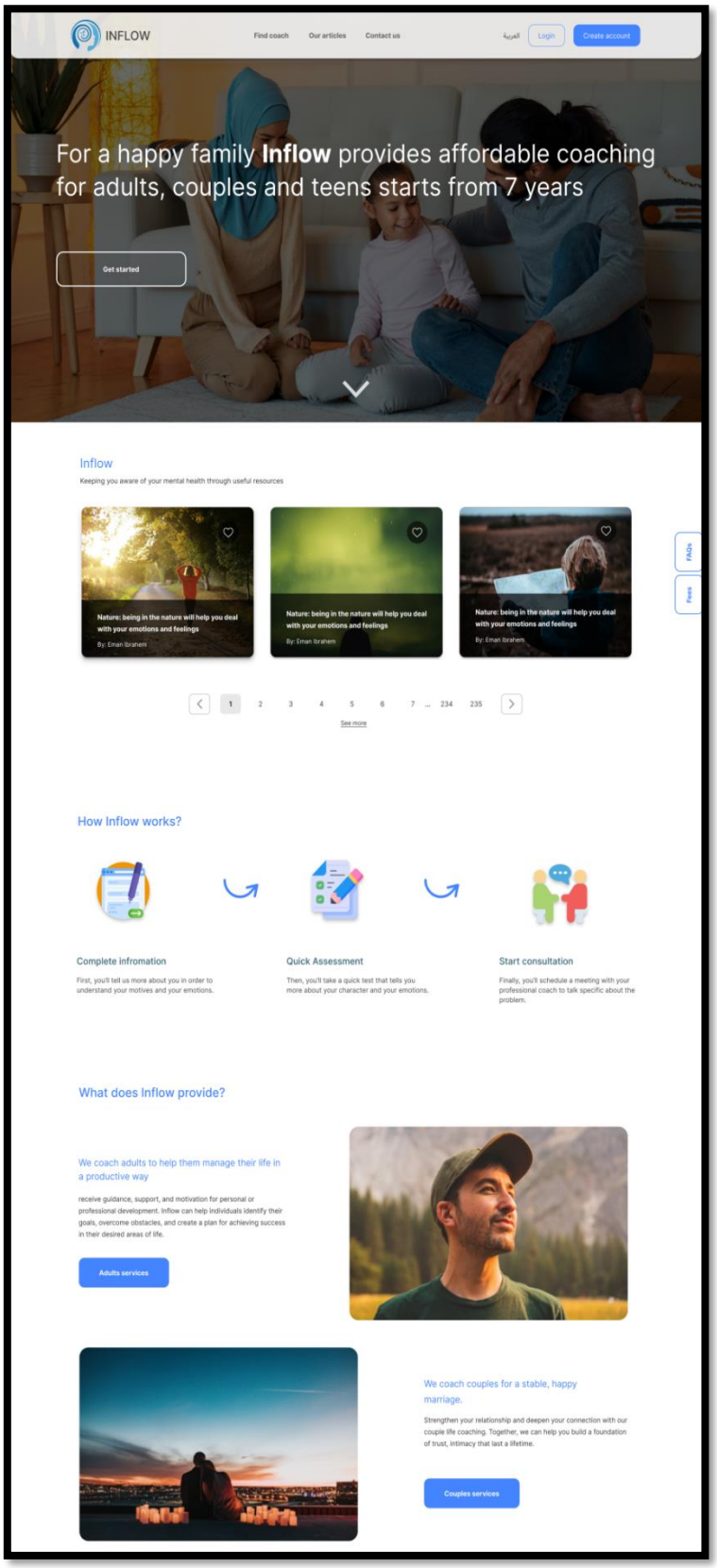
3.1 Web Application (User Interface):

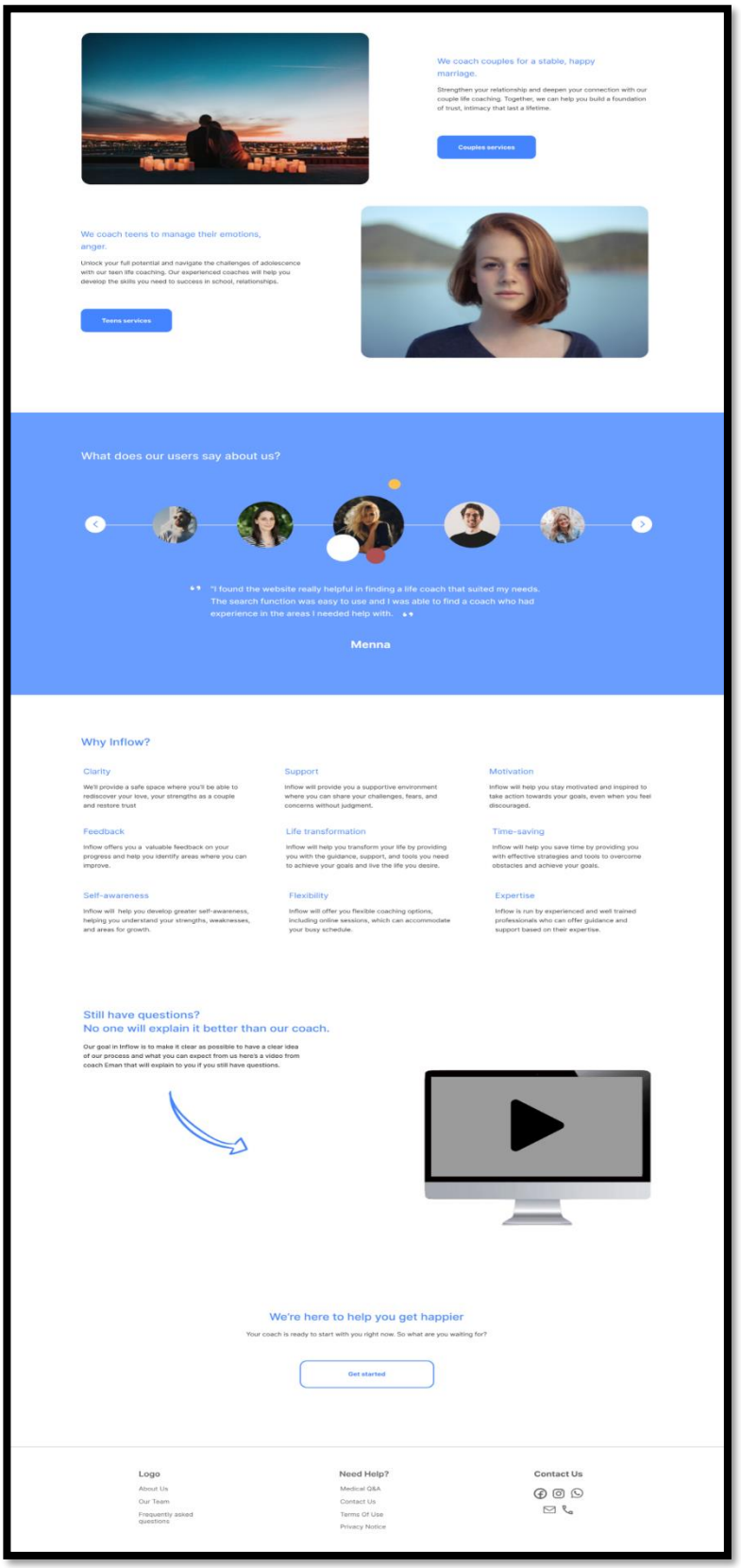


Login screen

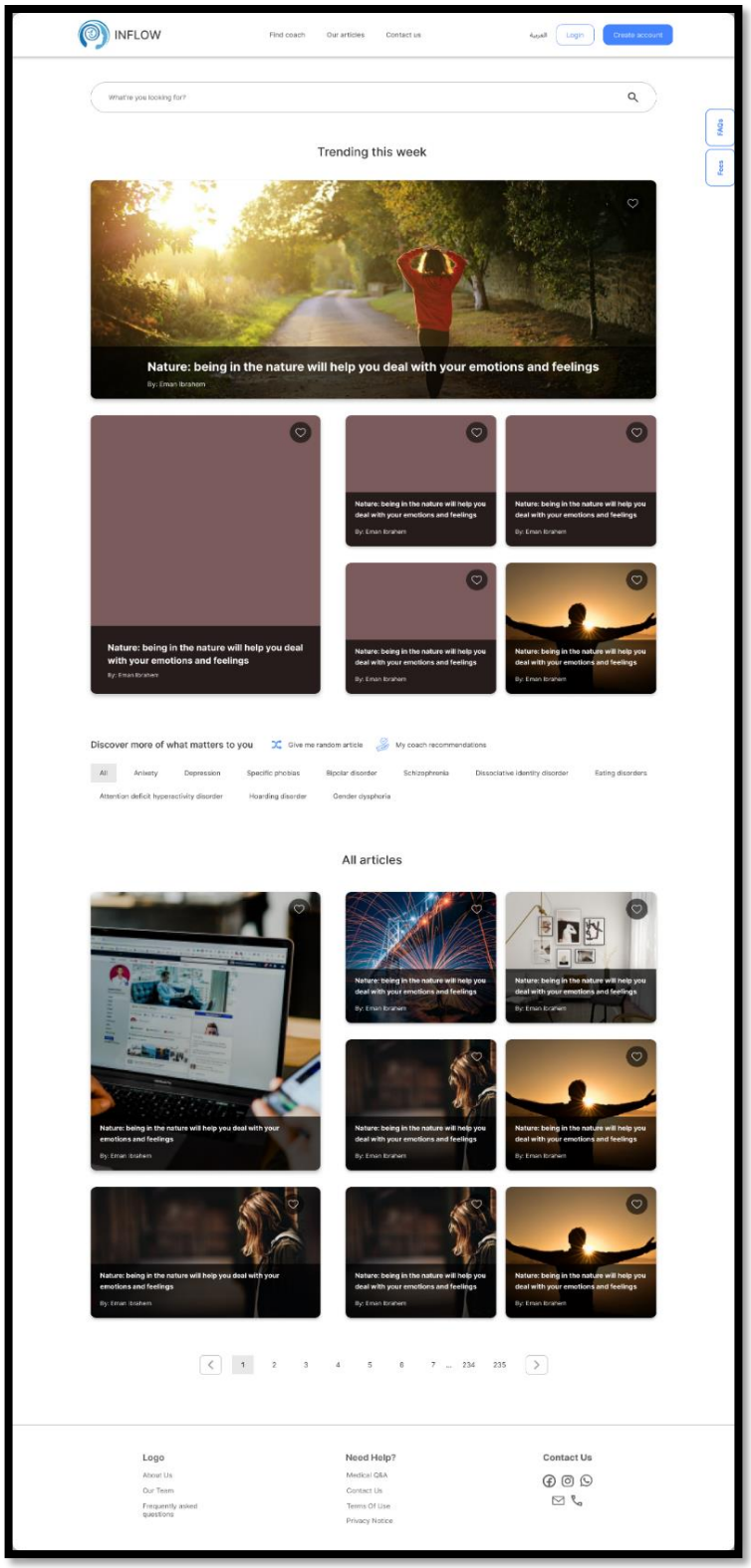


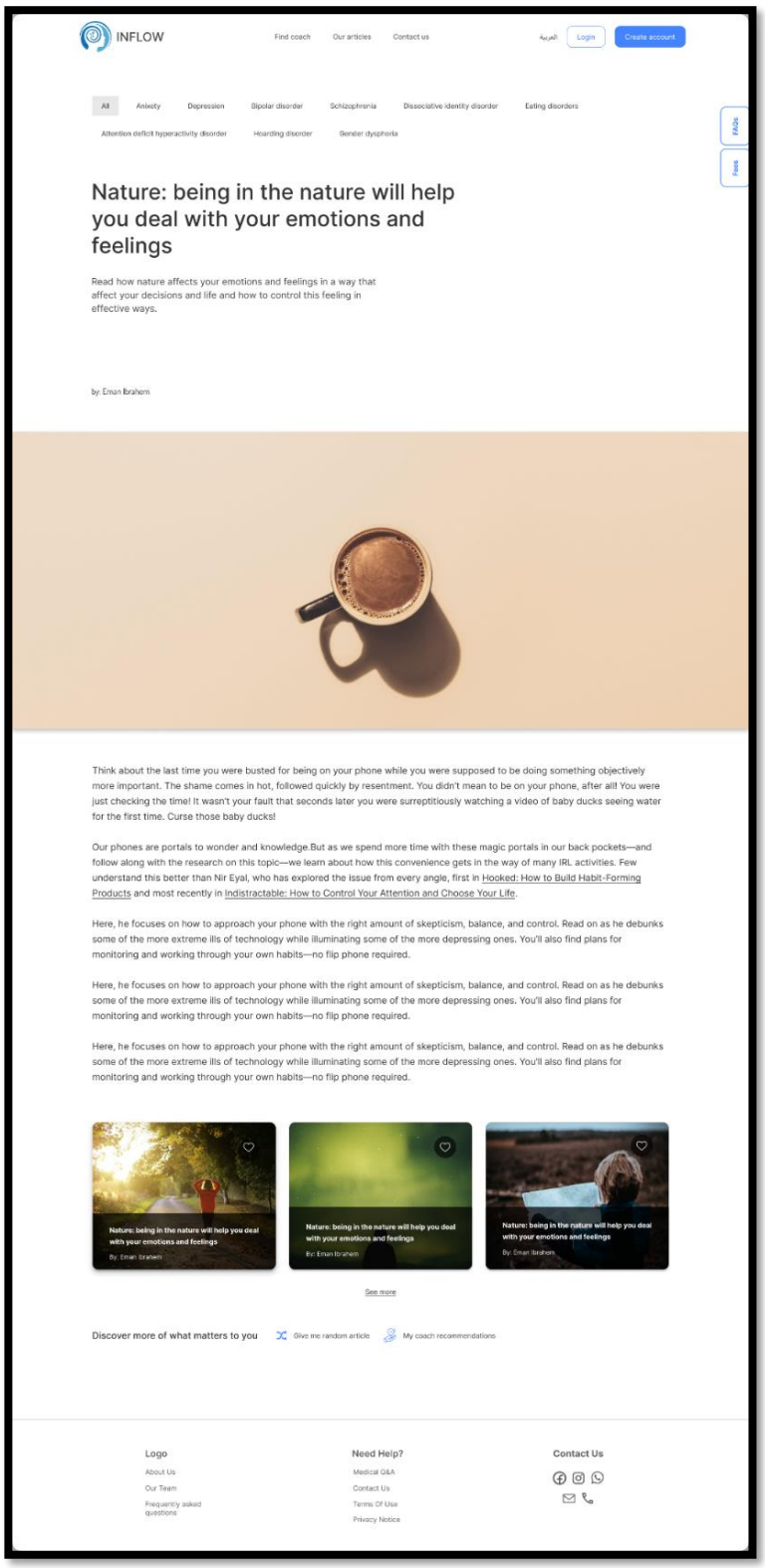
Signup Screen


INFLOW
Find coach
Our articles
Contact us
Login
Create account
For a happy family Inflow provides affordable coaching for adults, couples and teens starts from 7 years
Get started
Inflow
Keeping you aware of your mental health through useful resources
Nature: being in the nature will help you deal with your emotions and feelings
By: Eman Ibrahem
See more
How Inflow works?
Complete infromation
First, you'll tell us more about you in order to understand your motives and your emotions.
Quick Assessment
Then, you'll take a quick test that tells you more about your character and your emotions.
Start consultation
Finally, you'll schedule a meeting with your professional coach to talk specific about the problem.
What does Inflow provide?
We coach adults to help them manage their life in a productive way
receive guidance, support, and motivation for personal or professional development. Inflow can help individuals identify their goals, overcome obstacles, and create a plan for achieving success in their desired areas of life.
Adults services
We coach couples for a stable, happy marriage.
Strengthen your relationship and deepen your connection with our couple life coaching. Together, we can help you build a foundation of trust, intimacy that last a lifetime.
Couples services

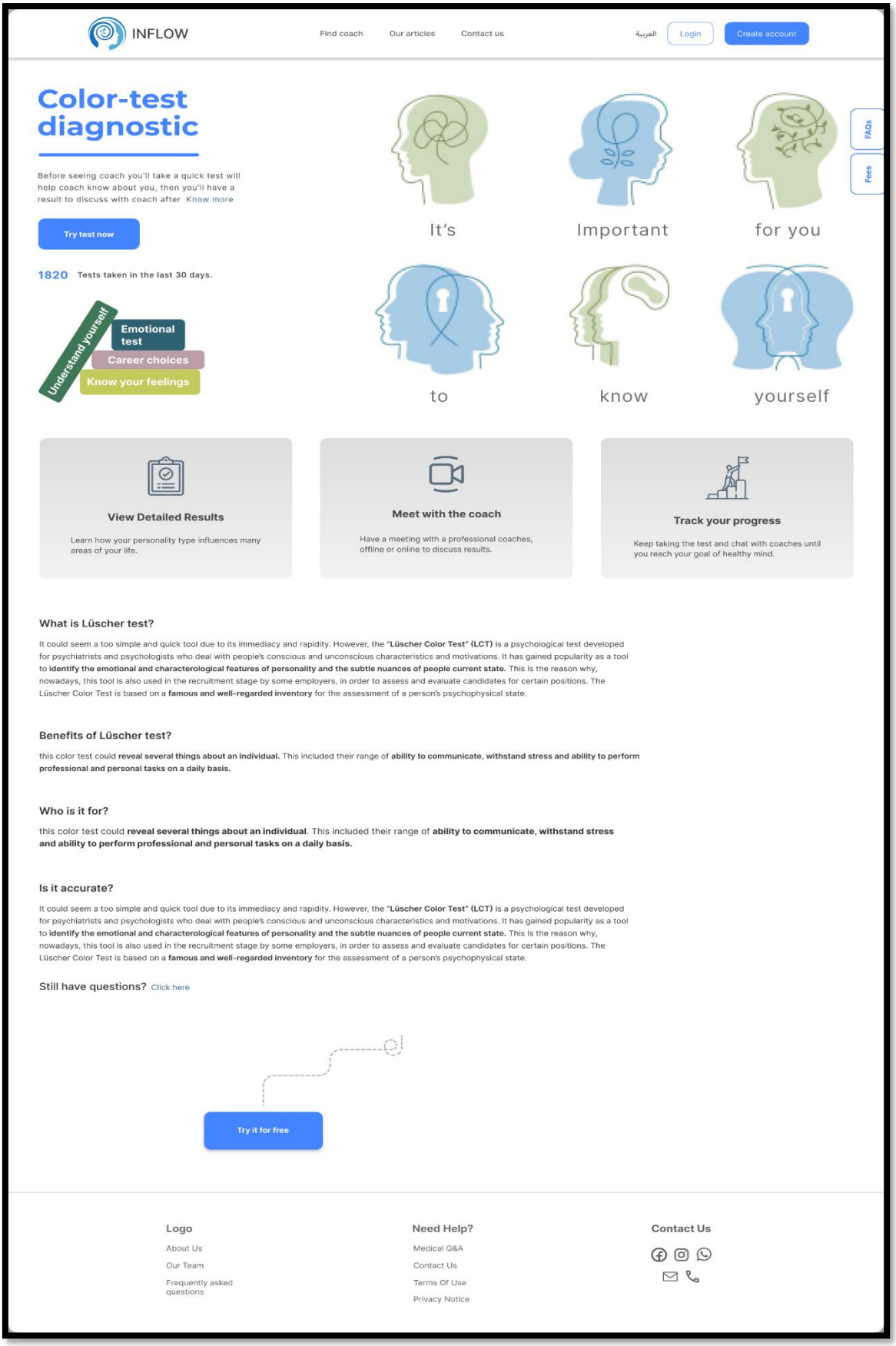

We coach couples for a stable, happy marriage.
Strengthen your relationship and deepen your connection with our couple life coaching. Together, we can help you build a foundation of trust, intimacy that last a lifetime.
Couples services
We coach teens to manage their emotions, anger.
Unlock your full potential and navigate the challenges of adolescence with our teen life coaching. Our experienced coaches will help you develop the skills you need to success in school, relationships.
Teens services
What does our users say about us?
"I found the website really helpful in finding a life coach that suited my needs. The search function was easy to use and I was able to find a coach who had experience in the areas I needed help with."
Menna
Why Inflow?
Clarity
Support
Motivation
Feedback
Life transformation
Time-saving
Self-awareness
Flexibility
Expertise
Still have questions?
No one will explain it better than our coach.
Our goal in Inflow is to make it clear as possible to have a clear idea of our process and what you can expect from us here's a video from coach Eman that will explain to you if you still have questions.
We're here to help you get happier
Your coach is ready to start with you right now. So what are you waiting for?
Get started
Logo
About Us
Our Team
Frequently asked questions
Need Help?
Medical Q&A
Contact Us
Terms Of Use
Privacy Notice
Contact Us

Home page

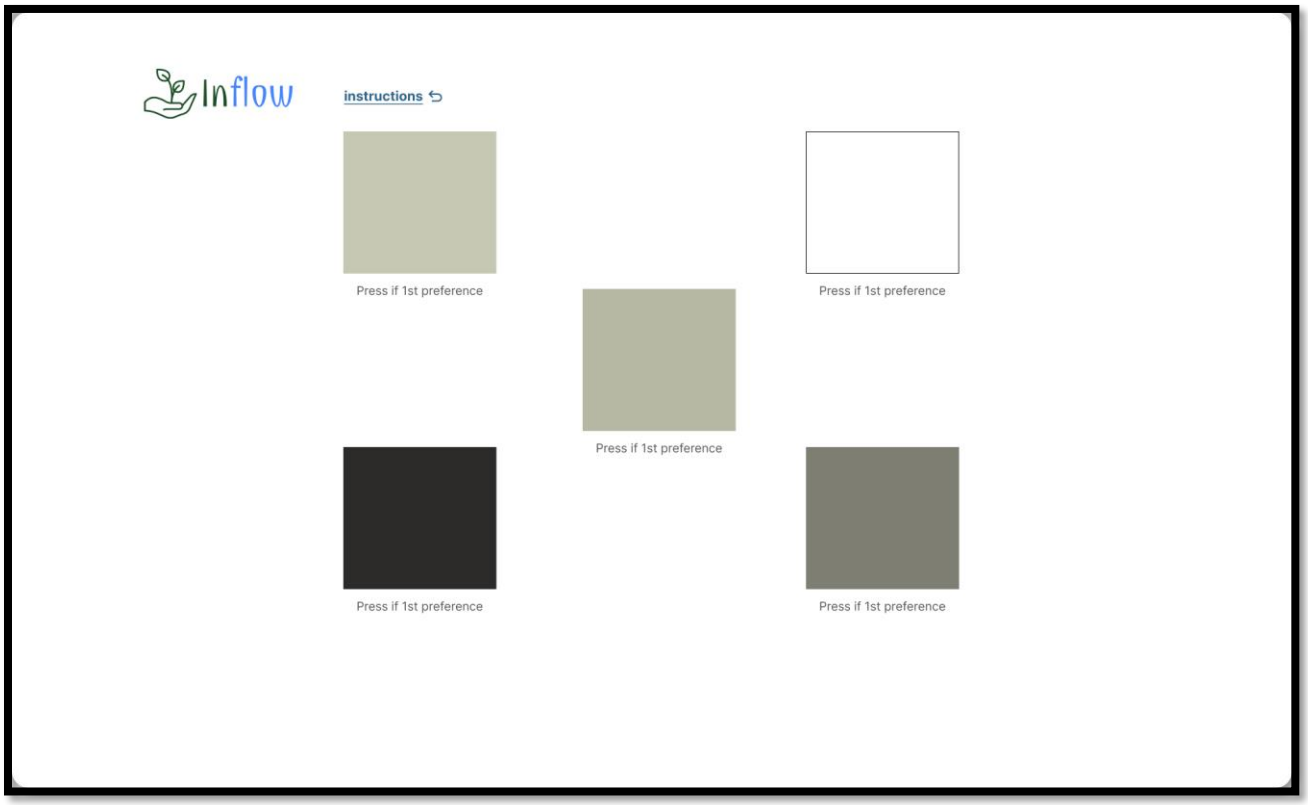

INFLOW
Find coach
Our articles
Contact us
Login
Create account
What're you looking for?
Trending this week
Nature: being in the nature will help you deal with your emotions and feelings
By: Eman Ibrahem
Discover more of what matters to you
Give me random article
My coach recommendations
All articles
Logo
About Us
Our Team
Frequently asked questions
Need Help?
Medical Q&A
Contact Us
Terms Of Use
Privacy Notice
Contact Us


INFLOW
Find coach
Our articles
Contact us
Login
Create account
Nature: being in the nature will help you deal with your emotions and feelings
Read how nature affects your emotions and feelings in a way that affect your decisions and life and how to control this feeling in effective ways.
by: Eman Ibrahem
Think about the last time you were busted for being on your phone while you were supposed to be doing something objectively more important. The shame comes in hot, followed quickly by resentment. You didn't mean to be on your phone, after all! You were just checking the time! It wasn't your fault that seconds later you were surreptitiously watching a video of baby ducks seeing water for the first time. Curse those baby ducks!
Our phones are portals to wonder and knowledge.But as we spend more time with these magic portals in our back pockets—and follow along with the research on this topic—we learn about how this convenience gets in the way of many IRL activities. Few understand this better than Nir Eyal, who has explored the issue from every angle, first in Hooked: How to Build Habit-Forming Products and most recently in Indistractable: How to Control Your Attention and Choose Your Life.
Here, he focuses on how to approach your phone with the right amount of skepticism, balance, and control. Read on as he debunks some of the more extreme ills of technology while illuminating some of the more depressing ones. You'll also find plans for monitoring and working through your own habits—no flip phone required.
Here, he focuses on how to approach your phone with the right amount of skepticism, balance, and control. Read on as he debunks some of the more extreme ills of technology while illuminating some of the more depressing ones. You'll also find plans for monitoring and working through your own habits—no flip phone required.
Here, he focuses on how to approach your phone with the right amount of skepticism, balance, and control. Read on as he debunks some of the more extreme ills of technology while illuminating some of the more depressing ones. You'll also find plans for monitoring and working through your own habits—no flip phone required.
See more
Discover more of what matters to you
Give me random article
My coach recommendations
Logo
About Us
Our Team
Frequently asked questions
Need Help?
Medical Q&A
Contact Us
Terms Of Use
Privacy Notice
Contact Us

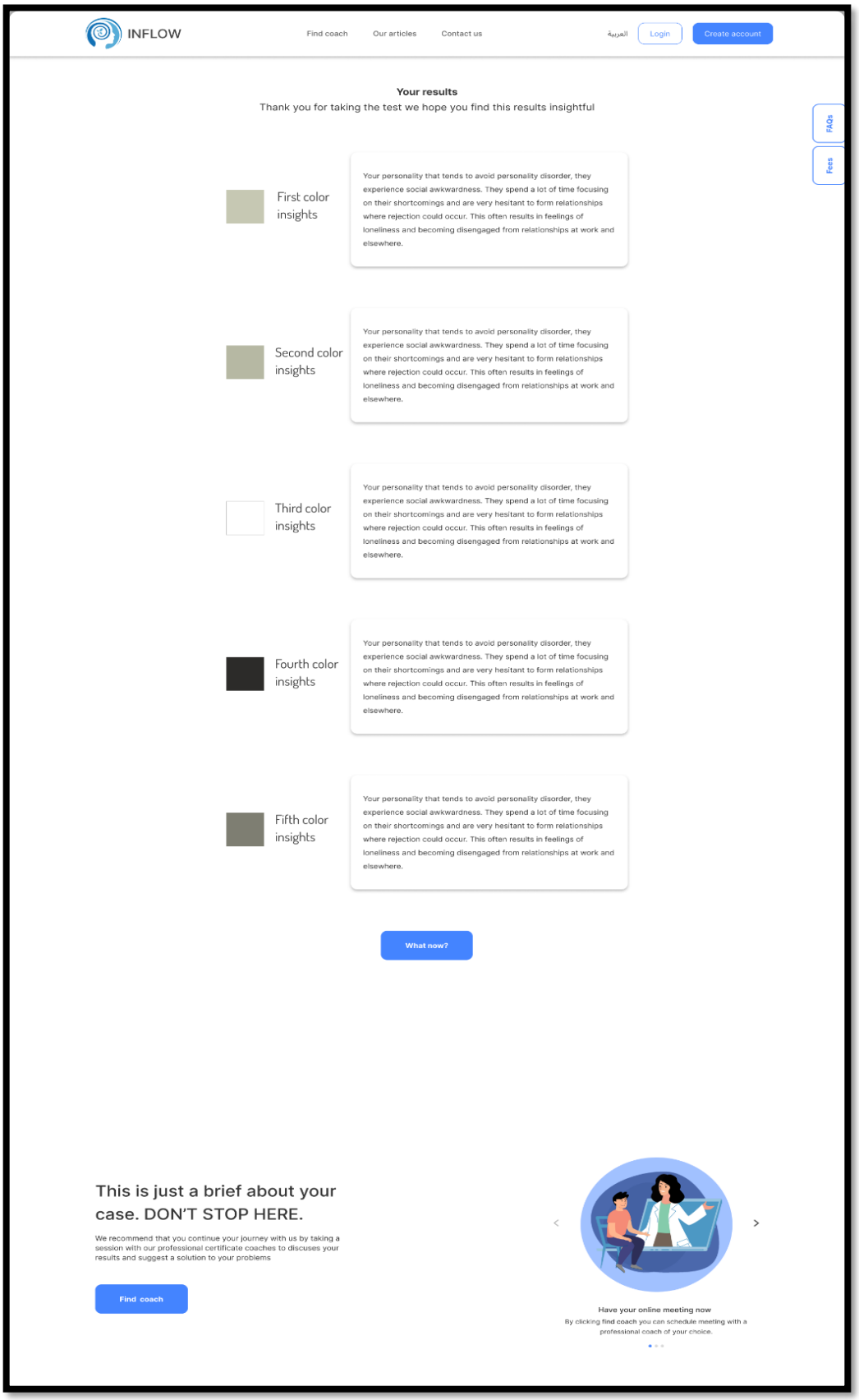
Articles Pa


INFLOW
Find coach
Our articles
Contact us
العربية
Login
Create account
Color-test diagnostic
Before seeing coach you'll take a quick test will help coach know about you, then you'll have a result to discuss with coach after Know more
Try test now
1820 Tests taken in the last 30 days.
Understand yourself
Emotional test
Career choices
Know your feelings
It's
Important
for you
to
know
yourself
FAQs
Fees
View Detailed Results
Learn how your personality type influences many areas of your life.
Meet with the coach
Have a meeting with a professional coaches, offline or online to discuss results.
Track your progress
Keep taking the test and chat with coaches until you reach your goal of healthy mind.
What is Lüscher test?
It could seem a too simple and quick tool due to its immediacy and rapidity. However, the "Lüscher Color Test" (LCT) is a psychological test developed for psychiatrists and psychologists who deal with people's conscious and unconscious characteristics and motivations. It has gained popularity as a tool to identify the emotional and characterological features of personality and the subtle nuances of people current state. This is the reason why, nowadays, this tool is also used in the recruitment stage by some employers, in order to assess and evaluate candidates for certain positions. The Lüscher Color Test is based on a famous and well-regarded inventory for the assessment of a person's psychophysical state.
Benefits of Lüscher test?
this color test could reveal several things about an individual. This included their range of ability to communicate, withstand stress and ability to perform professional and personal tasks on a daily basis.
Who is it for?
this color test could reveal several things about an individual. This included their range of ability to communicate, withstand stress and ability to perform professional and personal tasks on a daily basis.
Is it accurate?
It could seem a too simple and quick tool due to its immediacy and rapidity. However, the "Lüscher Color Test" (LCT) is a psychological test developed for psychiatrists and psychologists who deal with people's conscious and unconscious characteristics and motivations. It has gained popularity as a tool to identify the emotional and characterological features of personality and the subtle nuances of people current state. This is the reason why, nowadays, this tool is also used in the recruitment stage by some employers, in order to assess and evaluate candidates for certain positions. The Lüscher Color Test is based on a famous and well-regarded inventory for the assessment of a person's psychophysical state.
Still have questions? Click here
Try it for free
Logo
About Us
Our Team
Frequently asked questions
Need Help?
Medical Q&A
Contact Us
Terms Of Use
Privacy Notice
Contact Us

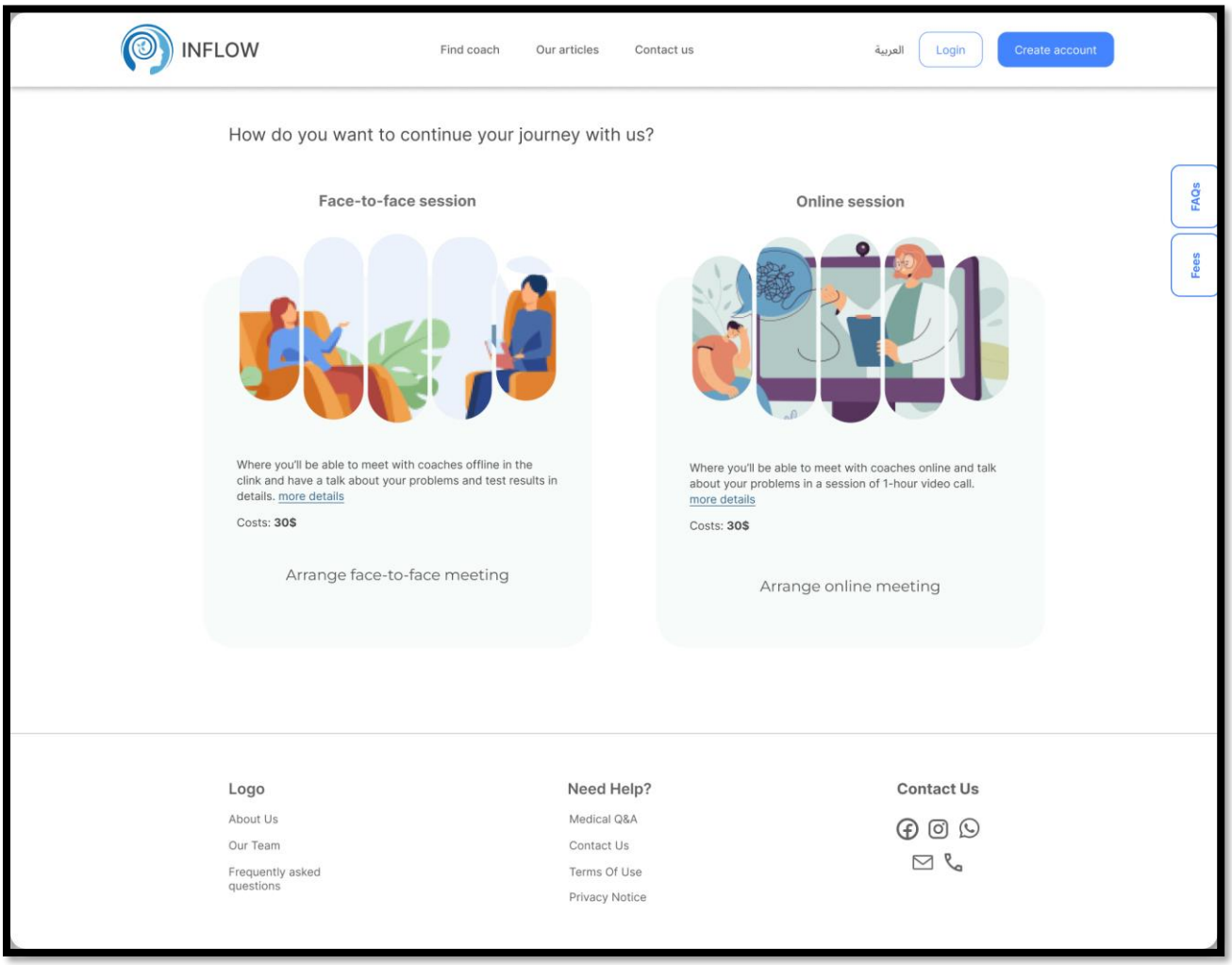
Pre-Test


Inflow
instructions
Press if 1st preference
Press if 1st preference
Press if 1st preference
Press if 1st preference
Press if 1st preference

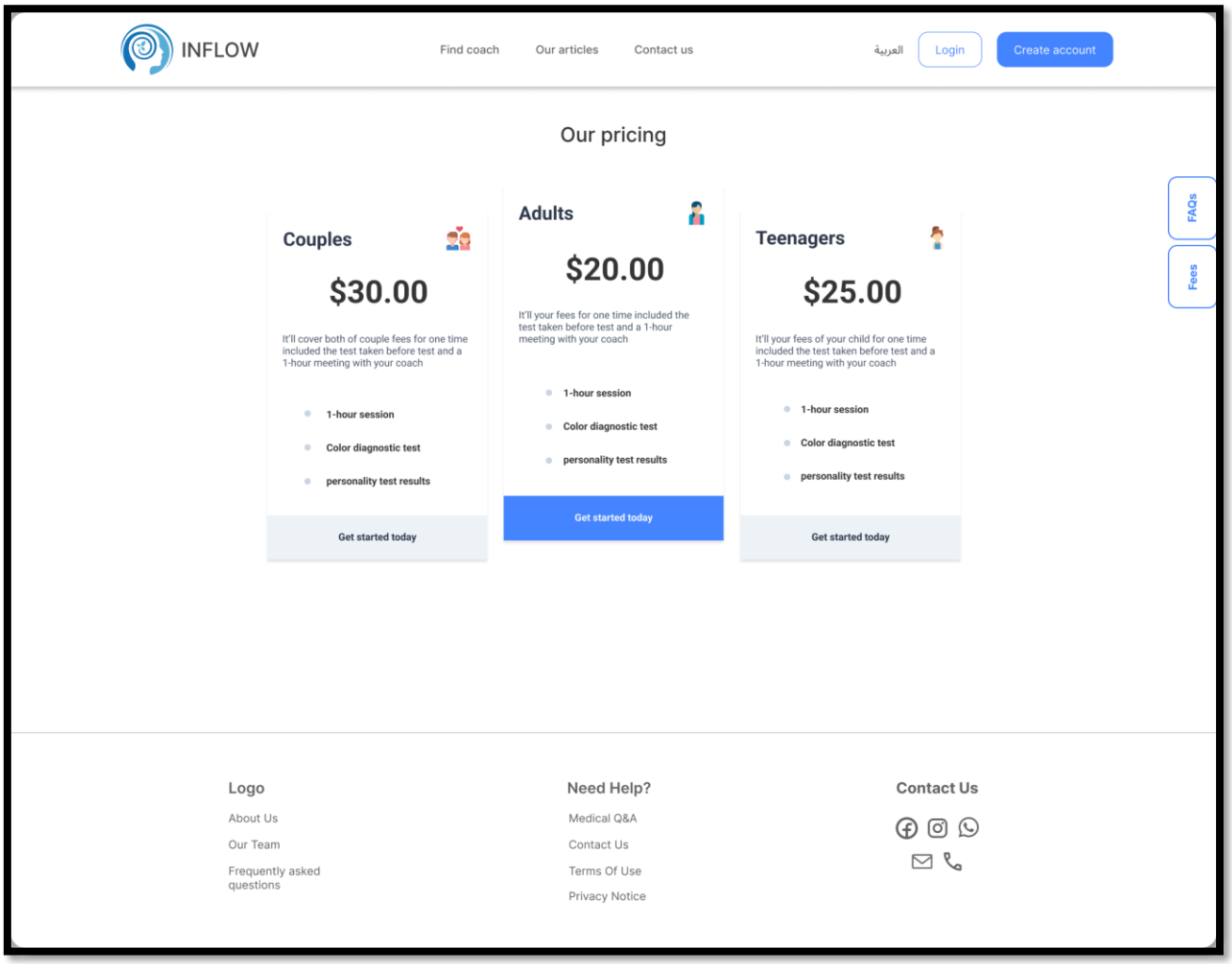
Test


INFLOW
Find coach
Our articles
Contact us
العربية
Login
Create account
Your results
Thank you for taking the test we hope you find this results insightful
FAQs
Fees
First color insights
Your personality that tends to avoid personality disorder, they experience social awkwardness. They spend a lot of time focusing on their shortcomings and are very hesitant to form relationships where rejection could occur. This often results in feelings of loneliness and becoming disengaged from relationships at work and elsewhere.
Second color insights
Your personality that tends to avoid personality disorder, they experience social awkwardness. They spend a lot of time focusing on their shortcomings and are very hesitant to form relationships where rejection could occur. This often results in feelings of loneliness and becoming disengaged from relationships at work and elsewhere.
Third color insights
Your personality that tends to avoid personality disorder, they experience social awkwardness. They spend a lot of time focusing on their shortcomings and are very hesitant to form relationships where rejection could occur. This often results in feelings of loneliness and becoming disengaged from relationships at work and elsewhere.
Fourth color insights
Your personality that tends to avoid personality disorder, they experience social awkwardness. They spend a lot of time focusing on their shortcomings and are very hesitant to form relationships where rejection could occur. This often results in feelings of loneliness and becoming disengaged from relationships at work and elsewhere.
Fifth color insights
Your personality that tends to avoid personality disorder, they experience social awkwardness. They spend a lot of time focusing on their shortcomings and are very hesitant to form relationships where rejection could occur. This often results in feelings of loneliness and becoming disengaged from relationships at work and elsewhere.
What now?
This is just a brief about your case. DON'T STOP HERE.
We recommend that you continue your journey with us by taking a session with our professional certificate coaches to discuses your results and suggest a solution to your problems
Find coach
Have your online meeting now
By clicking find coach you can schedule meeting with a professional coach of your choice.

Result

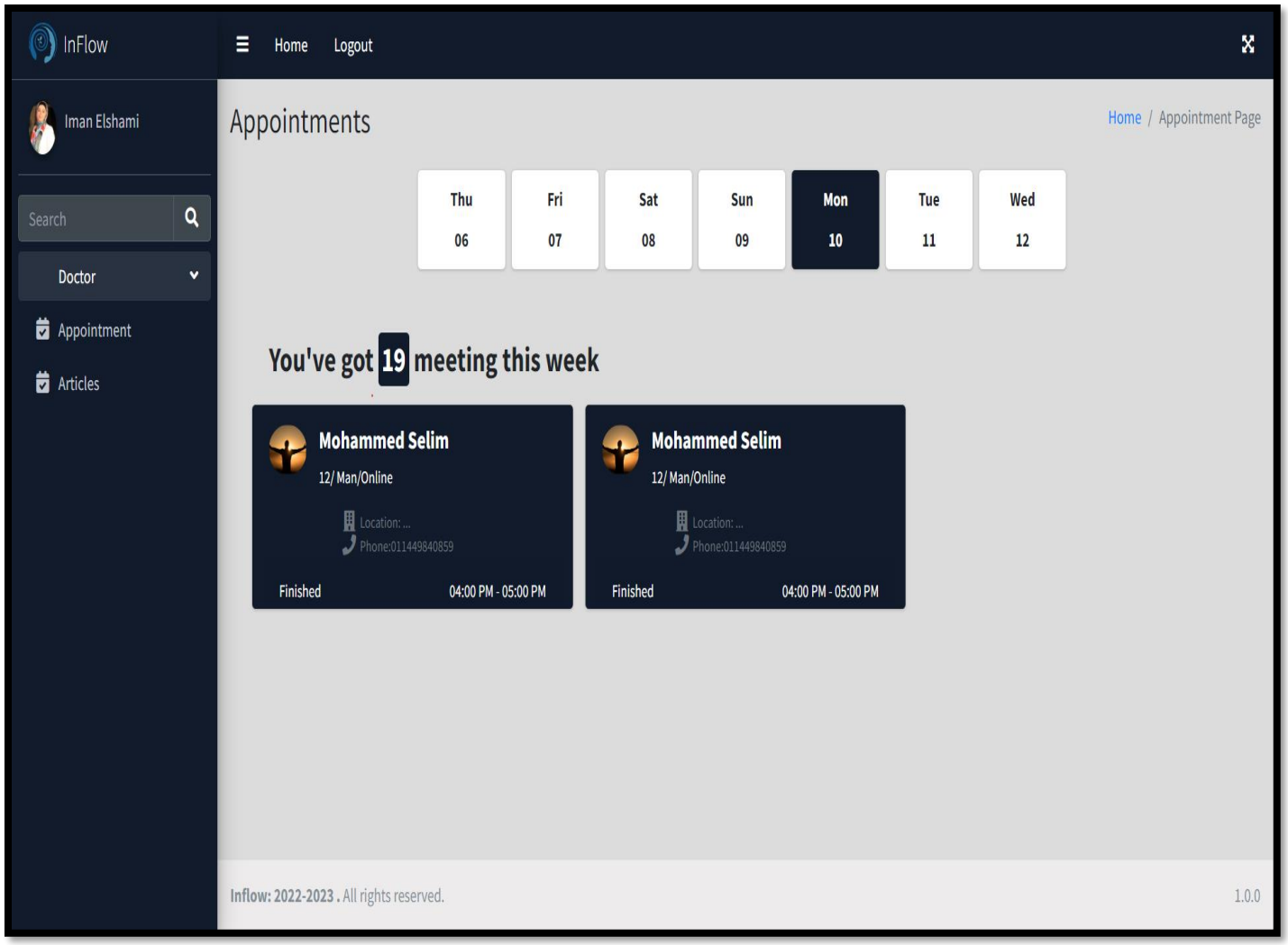


Appoint a Meeting

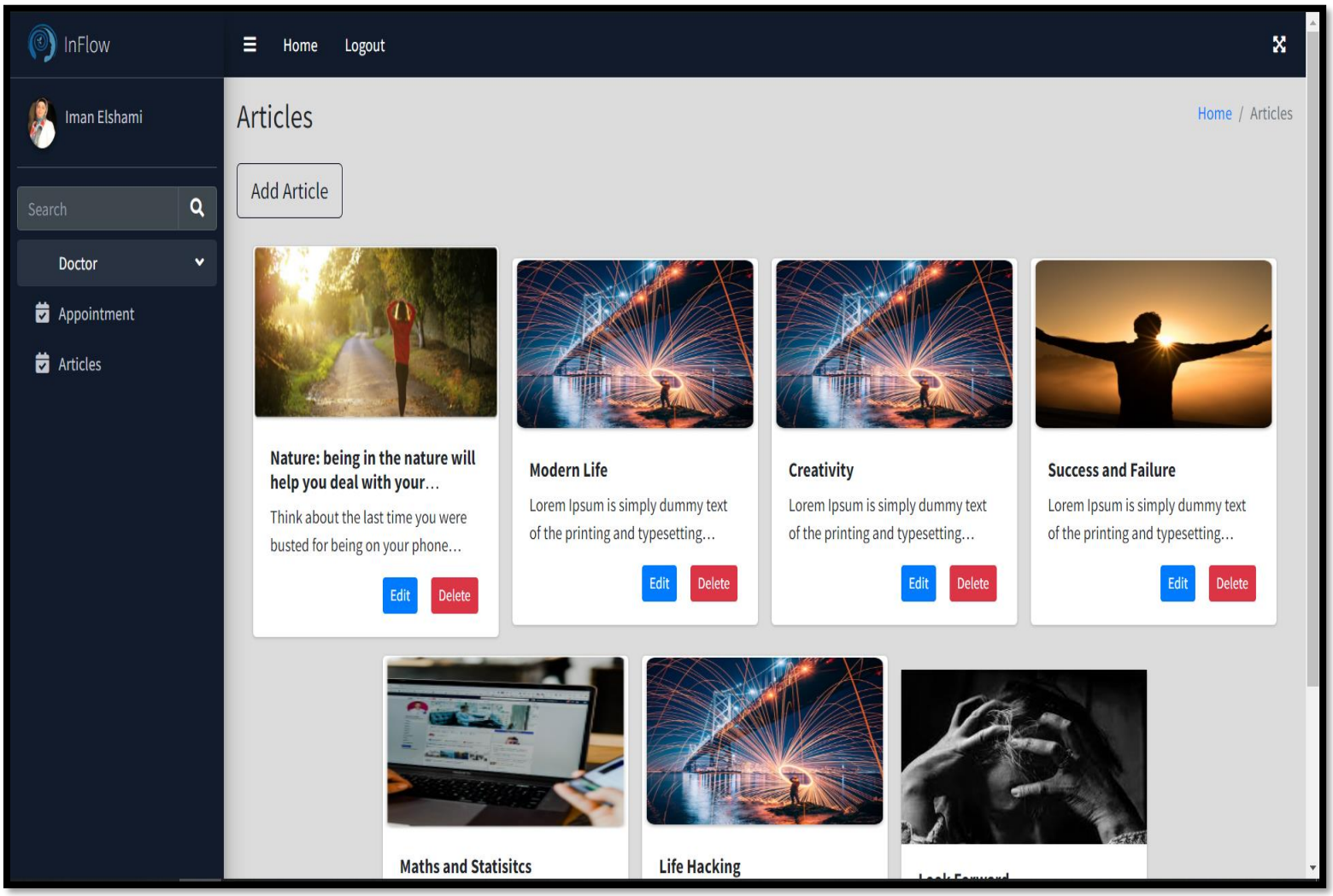

INFLOW
Find coach
Our articles
Contact us
العربية
Login
Create account
Our pricing
Couples
$30.00
It'll cover both of couple fees for one time included the test taken before test and a 1-hour meeting with your coach
1-hour session
Color diagnostic test
personality test results
Get started today
Adults
$20.00
It'll your fees for one time included the test taken before test and a 1-hour meeting with your coach
1-hour session
Color diagnostic test
personality test results
Get started today
Teenagers
$25.00
It'll your fees of your child for one time included the test taken before test and a 1-hour meeting with your coach
1-hour session
Color diagnostic test
personality test results
Get started today
FAQs
Fees
Logo
About Us
Our Team
Frequently asked questions
Need Help?
Medical Q&A
Contact Us
Terms Of Use
Privacy Notice
Contact Us

Pricing

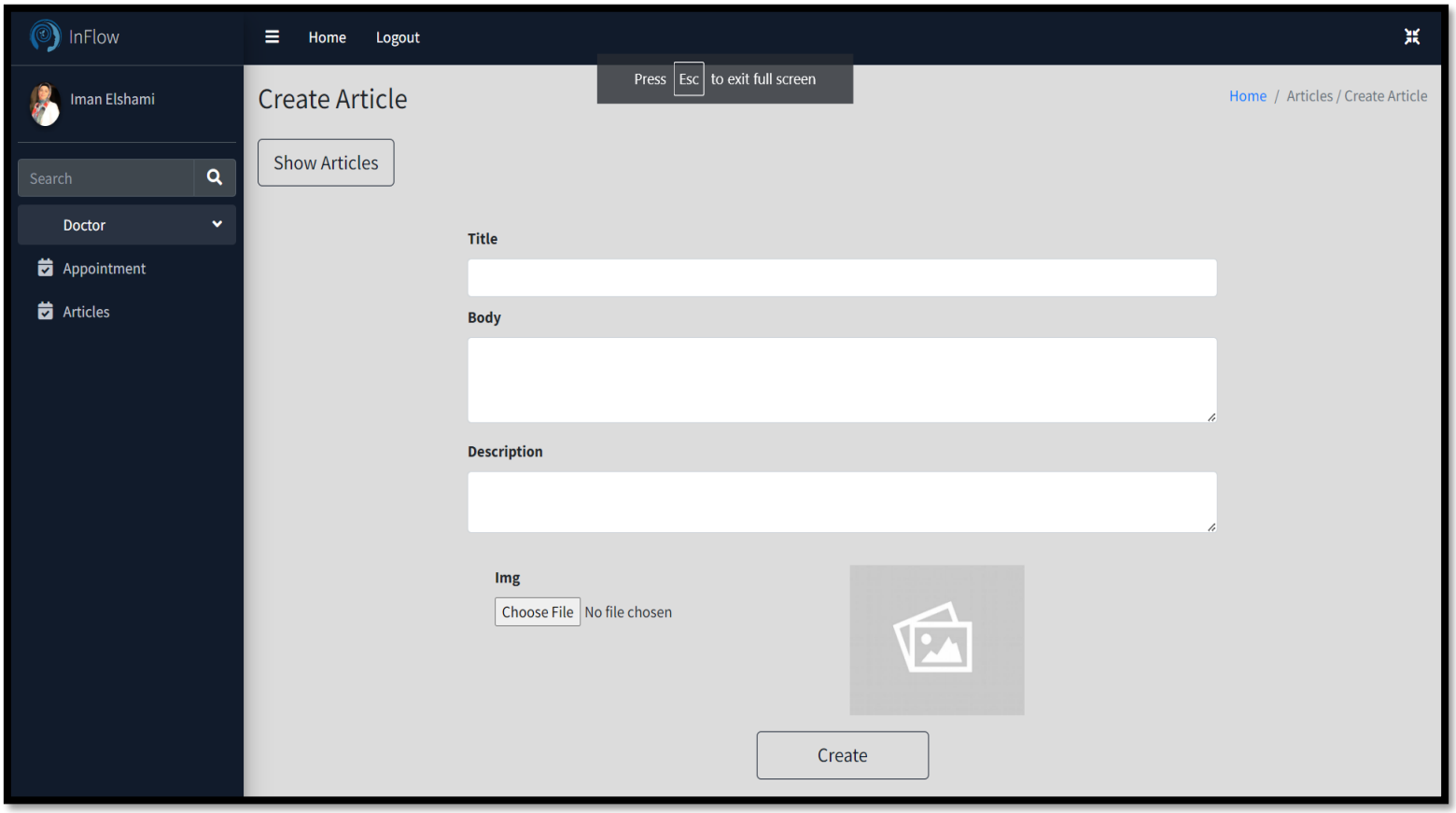
3.2 Web Application (Doctor and Secretary Interface):



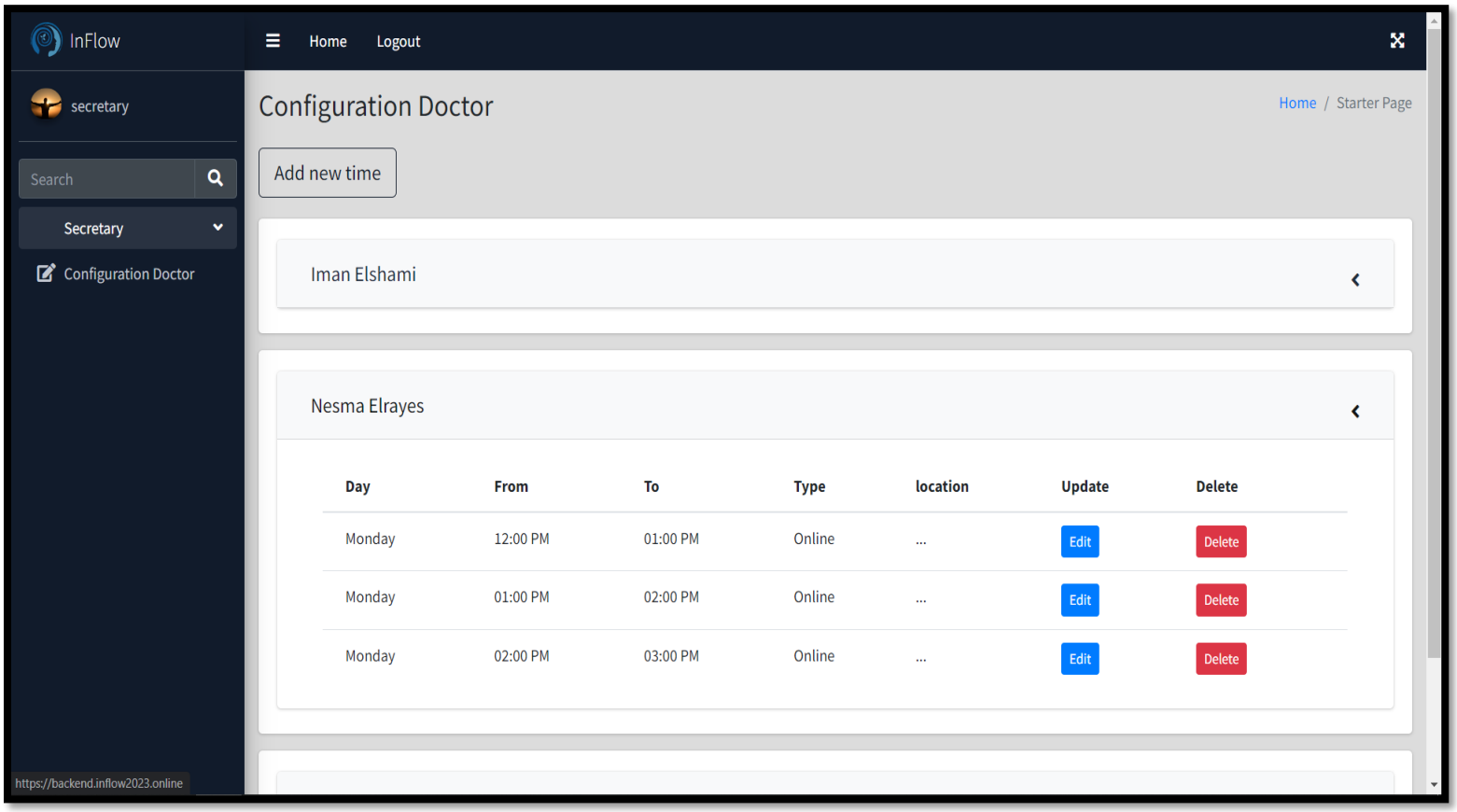
View Appointments



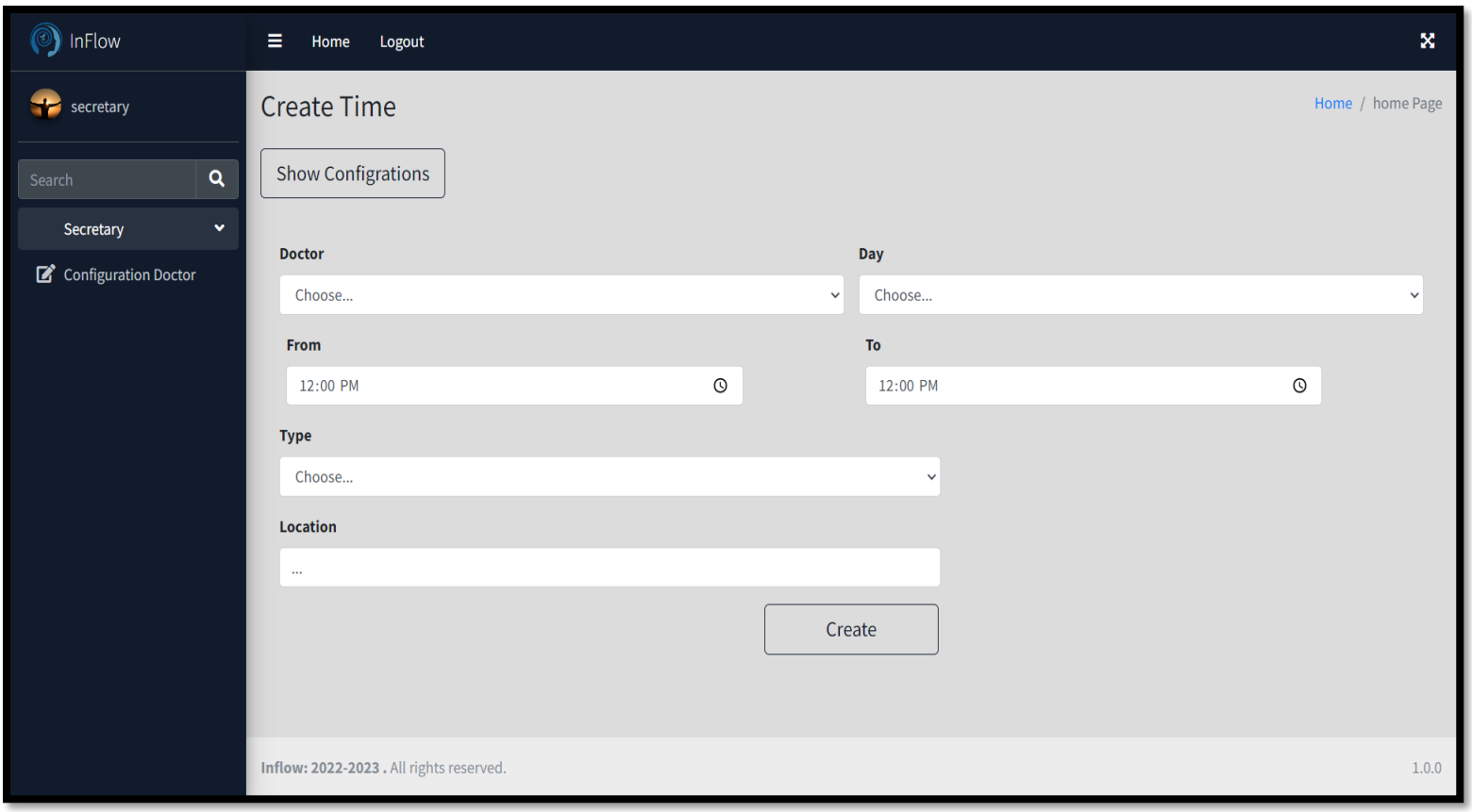
View Articles



Create Articles

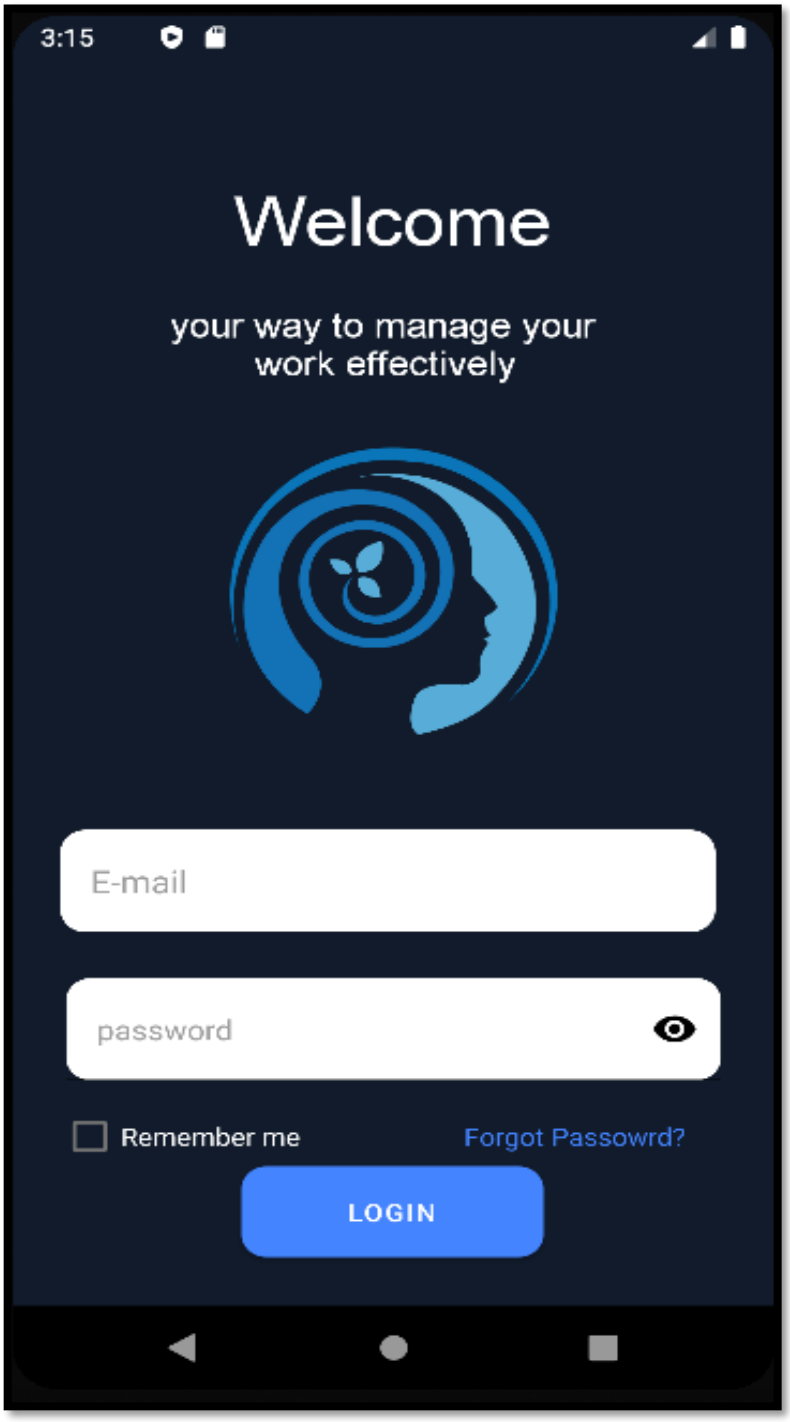


Appointment Configuration

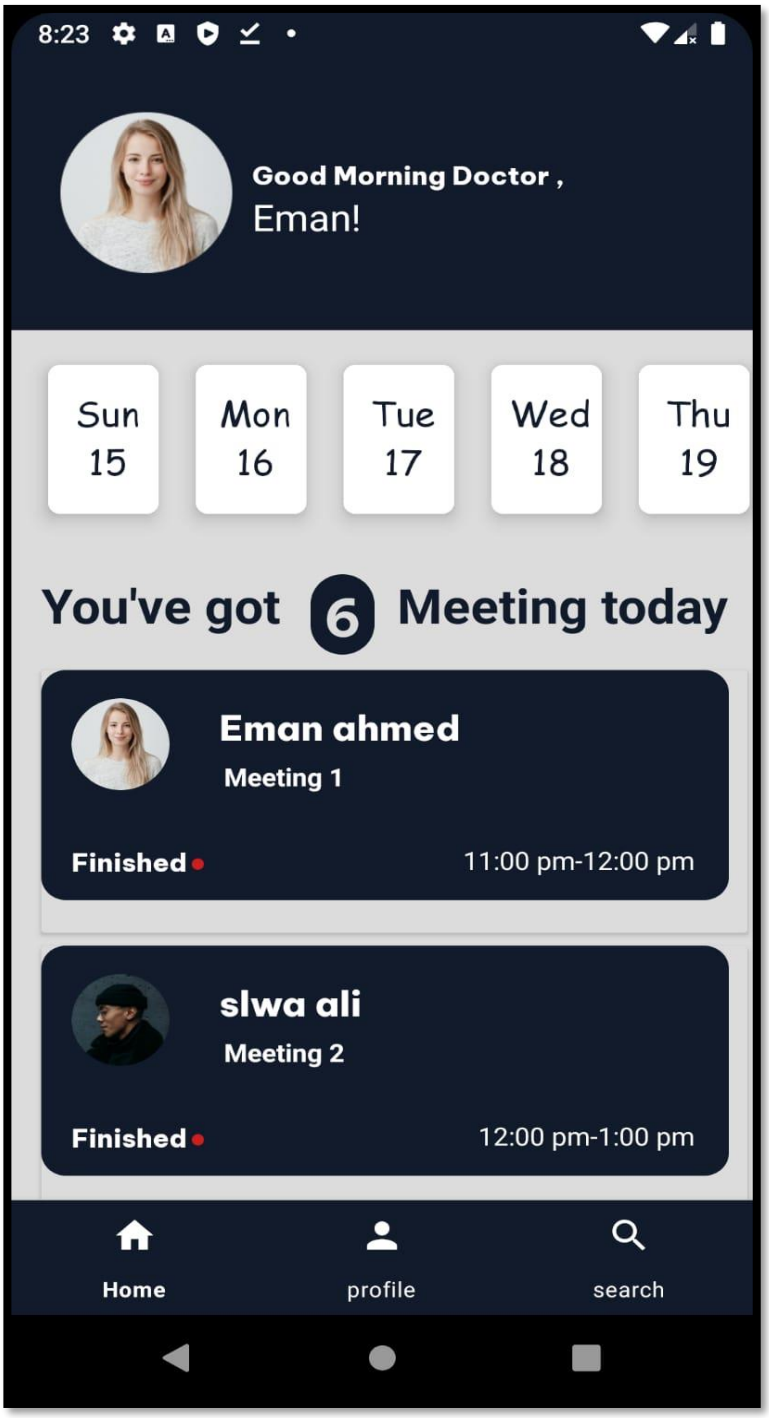


Create Appointments

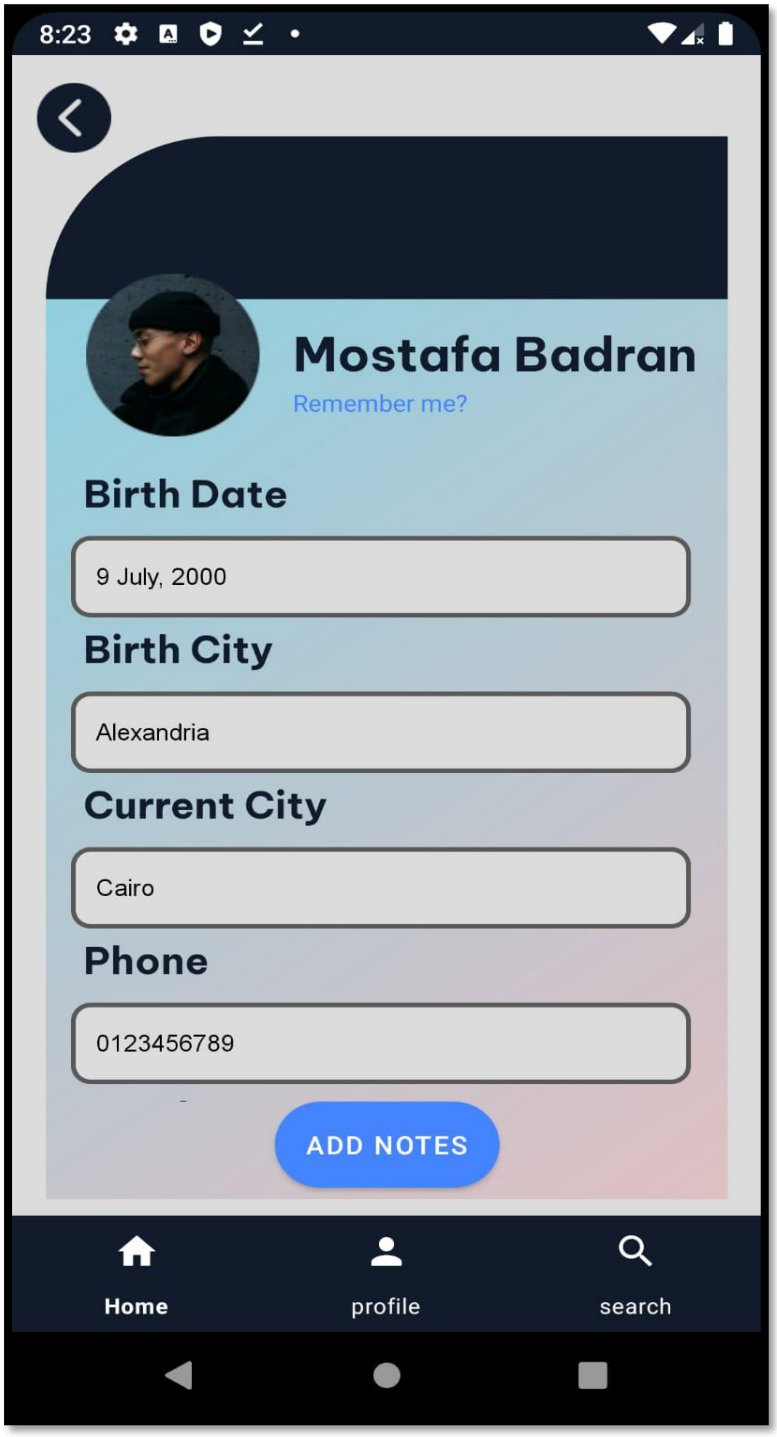
3.3 Mobile Application:



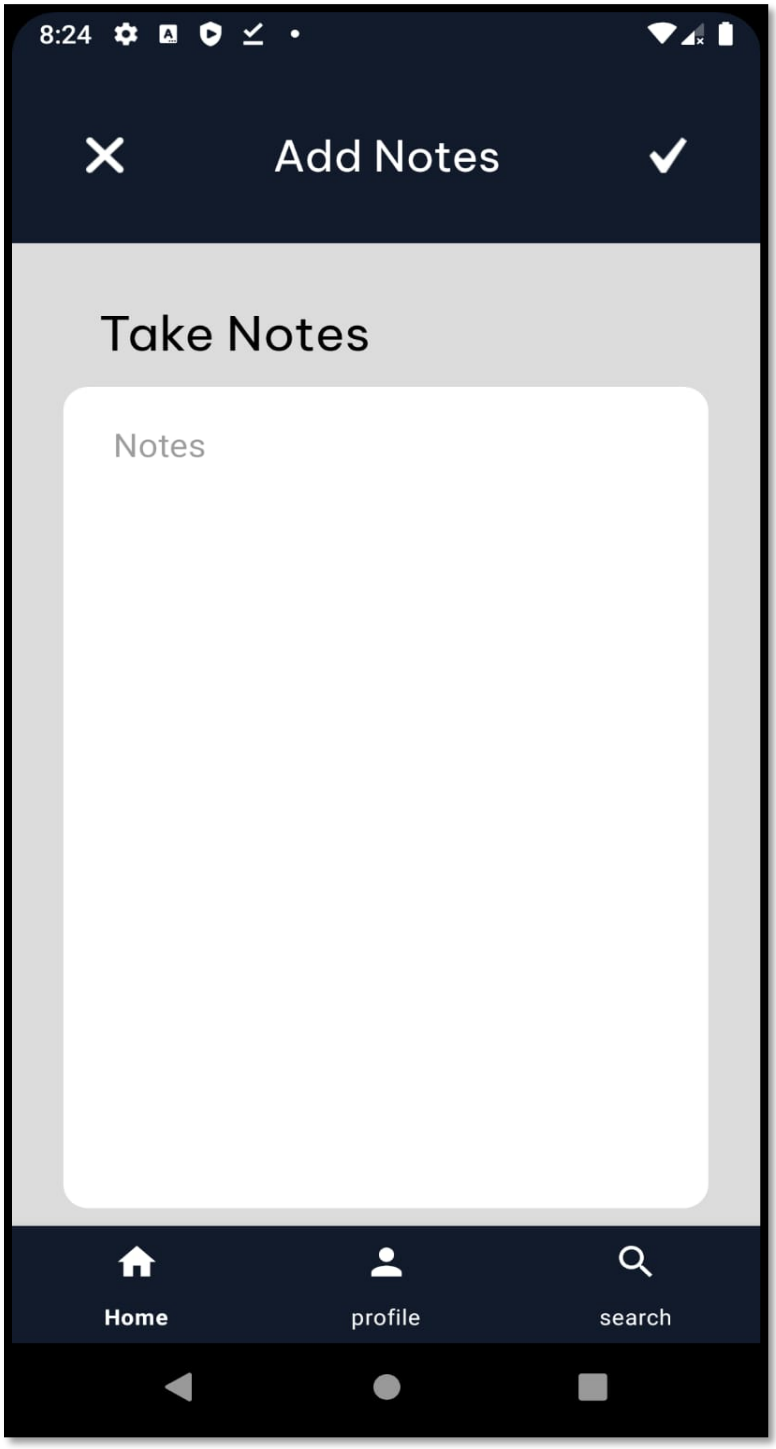
Login Screen


8:23
Good Morning Doctor ,
Eman!
Sun 15
Mon 16
Tue 17
Wed 18
Thu 19
You've got 6 Meeting today
Eman ahmed
Meeting 1
Finished
11:00 pm-12:00 pm
slwa ali
Meeting 2
Finished
12:00 pm-1:00 pm
Home
profile
search

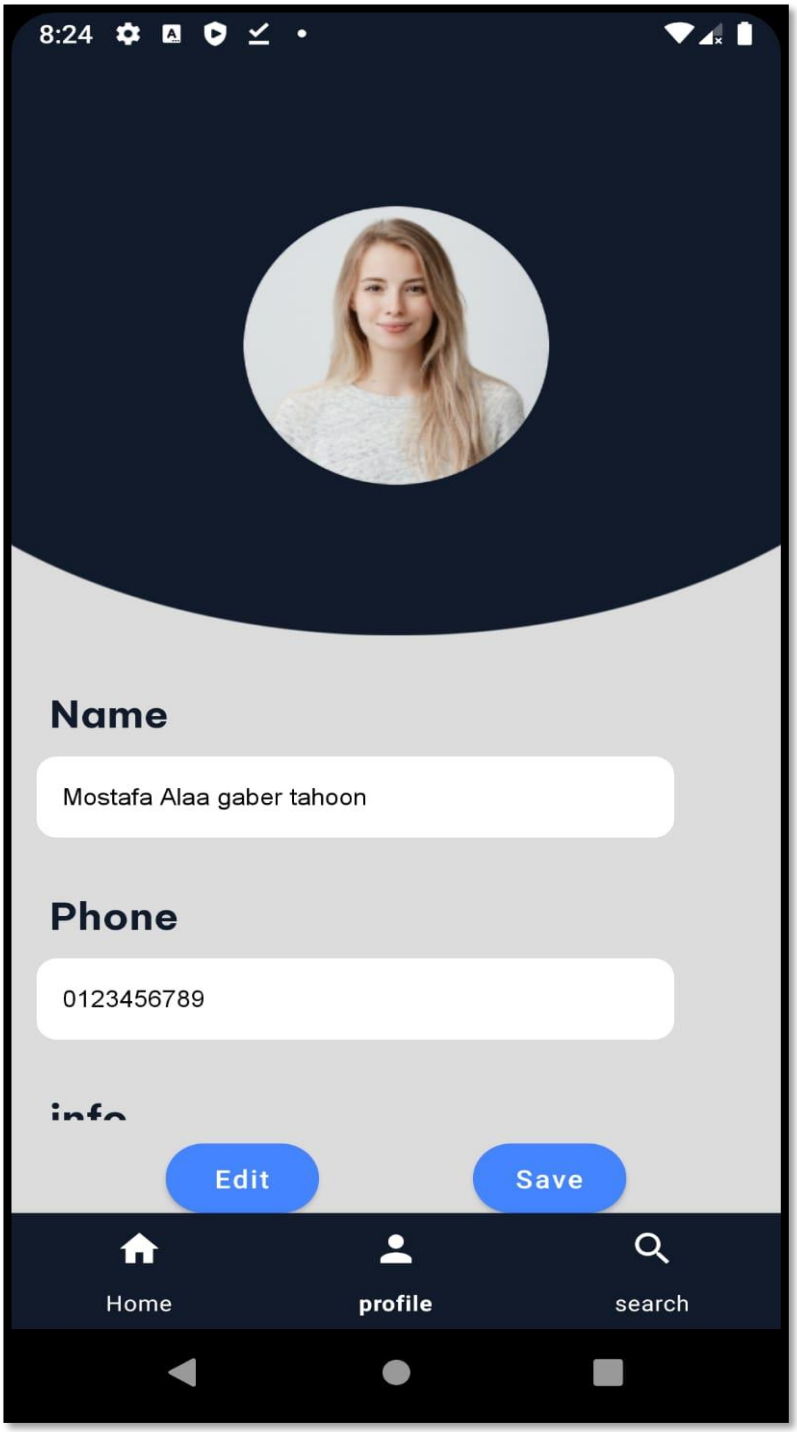
Home Page



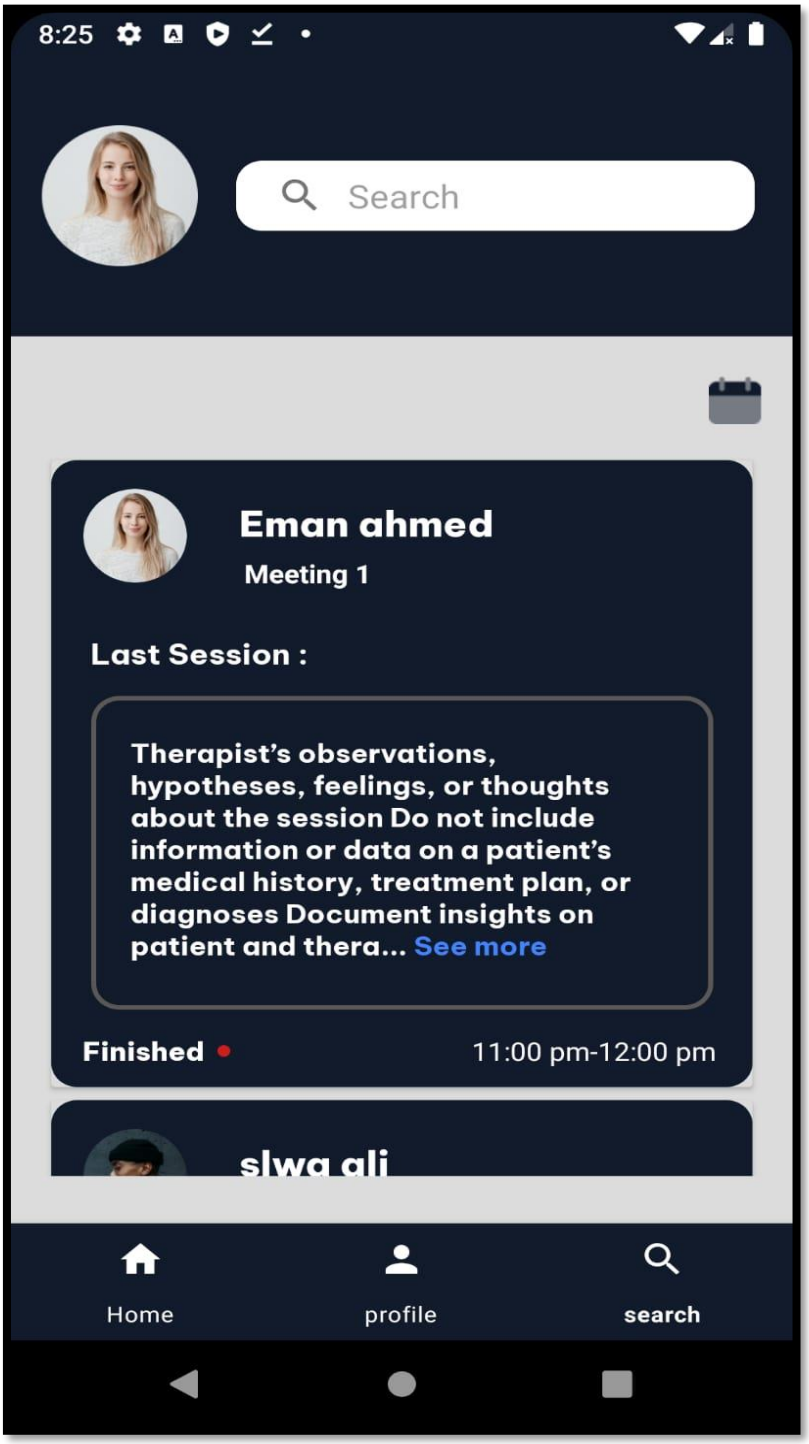
Patient data



Taking Note page



Doctor Profile



Search about patient

Remember the Meeting page.

Chapter Five:
Implementation

5.1 Website

5.1.1 Front End code:

1.1 app.jsx:

```jsx
import { BrowserRouter, Routes, Route } from "react-router-dom";
import { createBrowserRouter , RouterProvider} from 'react-router-dom'
import NewHome from "./components/Home/NewHome/NewHome";
import Signin from "./components/Signin/Signin";
import Signup from "./components/Signup/Signup";
import Home from "./components/Home/Home/Home";
import Headerguest from "./components/Header/Headerguest";
import Footer from "./components/Footer/Footer";
import Hometest from "./components/Home/Hometest/Hometest";
import AccountVerfication from "./components/AccountVerfication/AccountVerfication";
import LinkReset from "./components/LinkReset/LinkReset";
import PasswordAssistance from "./components/PasswordAssistance/PasswordAssistance";
import TestInstructions from "./components/TestInstructions/TestInstructions";
import Results from "./components/Results/Results";
import BookMeeting from "./components/BookMeeting/BookMeeting";
import Coach from "./components/coach/coach";
import Testinformation from "./components/Testinformation/Testinformation";
import TestPage from "./components/TestPage/TestPage";
import Articles from "./components/Articles/Articles";
import ArticlePage from "./components/Articles/ArticlePage/ArticlePage";
import ContactForm from "./components/Contactus/ContactForm";
import Payment from "./components/Payment/Payment";
import CompletePayment from "./components/Payment/CompletePayment/CompletePayment";
import OTP from "./components/Payment/OTP/OTP";


import 'bootstrap/dist/css/bootstrap.min.css';


export default function App() {


  const registerRouter = createBrowserRouter([
      {
        path: '/',
        element: <div><Headerguest/> <div><hr/></div> <Signin /> </div>,

      },
  ]);
```

```
40
41    const homeRouter = createBrowserRouter([
42      {
43        path: '/home', element: <Hometest/>,
44      },
45    ]);
46
47
48    return (
49      <>
50        <BrowserRouter>
51          <Routes>
52            <Route path="/" element={ <NewHome /> } />
53            <Route path="/home" element={<Home /> } />
54            <Route path="/signup" element={ <div><Signup />  <Footer /> </div> } />
55            <Route path="/signin" element={ <div> <Headerguest />  <Signin /> <hr/> <Footer /></div> } />
56            <Route path="/hometest" element={ <div><Hometest />  <Footer /> </div> } />
57            <Route path="/account-verfication" element={ <div><AccountVerfication />  <Footer /> </div> } />
58            <Route path="/Link-Reset" element={ <div><LinkReset />  <Footer /> </div> } />
59            <Route path="/password-assistance" element={ <div><PasswordAssistance />  <Footer /> </div> } />
60            {/* <Route path="/information-form" element={ <div><InformationForm />  <Footer /> </div> } /> */}
61            <Route path="/test-instructions" element={ <div><TestInstructions /> </div>} />
62            <Route path="/articles" element={ <div><Articles />  <Footer /></div>} />
63            <Route path="/articlepage" element={ <div><ArticlePage />  <Footer /></div>} />
64            <Route path="/test-information" element={ <div><Testinformation /> <Footer /> </div>} />
65            <Route path="/testpage" element={ <div><TestPage />  </div>} />
66            <Route path="/results" element={ <div><Results />  <Footer /></div>} />
67            <Route path="/bookmeeting" element={ <div><BookMeeting />  <Footer /></div>} />
68            <Route path="/contactus" element={ <div><ContactForm />  <Footer /></div>} />
69            <Route path="/coach" element={ <div><Coach />  <Footer /></div>} />
70            <Route path="/payment" element={ <div><Payment />  <Footer /></div>} />
71            <Route path="/completepayment" element={ <div><CompletePayment />  <Footer /></div>} />
72            <Route path="/otp" element={ <div><OTP />  <Footer /></div>} />
73          </Routes>
74        </BrowserRouter>
75      </>
76    );
77
78  }
79
```

1.2 Home.jsx:

```jsx
import React, { useState } from "react";
import { useEffect } from 'react';

import { useNavigate } from 'react-router-dom';

import Headerguesthome from "../../Header/Headerguesthome";
import HomeHeader from '../../Header/HomeHeader';
import ScrollContent from "./ScollContent/ScrollContent";


import { Grid } from '@mui/material';

import classes from './NewHome.module.css';

const NewHome = () => {

    const [targetSection, setTargetSection] = useState(null);
    const [showContent, setShowContent] = useState(false);
    const [removecontent, setremoveContent] = useState(false);

    const navigate = useNavigate();
    const hometestBtn = () => {
        navigate("/hometest")
    };

    useEffect(() => {
        const section = document.getElementById('target-section');
        setTargetSection(section);
    }, []);
    useEffect(() => {
        window.scrollTo(0, 0);
    }, []);

    const handleClick = () => {
        if (targetSection) {
            window.scrollTo({ top: 660, behavior: 'smooth' });
        }
    };

    const accessToken = localStorage.getItem('token');

```

```
41
42
43      return (
44          <div>
45              <div className={classes.home}>
46              {accessToken ? <HomeHeader /> : <Headerguesthome />}
47                  <Grid>
48                      <Grid container style={{ height: '80vh', padding: '0 9%' }} justify="center" alignItems="center">
49                          {!showContent && (
50                                  <div>
51                                      <Grid container >
52                                          <Grid item xl={12} xs={12} md={12} style={{ textAlign: 'center' }}>
53                                              <div className={classes.faqs}>
54                                                  <h1 style={{ textAlign: 'left' }}>For a happy family <b>Inflow </b> provides affordable online coaching for couples, individuals starts from 7 years</h1>
55                                                  <div style={{ display: 'flex', justifyContent: 'left', marginTop: "2%" }}>
56                                                      <button onClick={hometestBtn}>Get started</button>
57                                                  </div>
58                                              </div>
59                                          </Grid>
61                                          <Grid item xs={12} md={12} xl={12} style={{  justifyContent: 'center' }}>
62                                              <div className={classes.downarrow} onClick={handleClick}>
63                                                  <svg width="48" height="45" viewBox="0 0 75 45" fill="none" xmlns="http://www.w3.org/2000/svg">
64                                                      <path d="M0 6.67745L37.5 44.2969L75 6.67745L68.3437 0L37.5 30.942L6.65625 0L0 6.67745Z" fill="white"/>
65                                                  </svg>
66
67                                              </div>
68                                          </Grid>
69
70                                      </Grid>
71
72
73                                  </div>
74                              )
75                          }
76
77
78                  </Grid>
79              </Grid>
80          </div>
81
82          { !removecontent &&
83              <div id="target-section">
84                  <ScrollContent/>
85              </div>
86          }
87
88
89      </div>
90
91
92  );
93  }
94
95  export default NewHome;
```

1.3 Test.jsx:

```jsx
import React, { useState } from 'react';
import Logo from '../../assets/Logo.jpg';
import firstcolor from '../../assets/firstcolor.jpg';
import secondcolor from '../../assets/secondcolor.PNG';
import thirdcolor from '../../assets/thirdcolor.PNG';
import forthcolor from '../../assets/forthcolor.PNG';
import lastcolor from '../../assets/lastcolor.jpg';

import { Button, Grid } from '@mui/material';
import classes from "./TestPage.module.css";
import { useNavigate } from "react-router-dom";
import axios from 'axios';
import { API_URL } from "../../api.js"


const TestPage =()=> {

    const navigate = useNavigate();
    const nextpage = () => {
        navigate("/results");
    }

    const [selectedImages, setSelectedImages] = useState([]);
    const [selectedImagePaths, setSelectedImagePaths] = useState([]);
    const [text, setText] = useState("Choose your 1st Option");
    const [isButton3Visible, setIsButton3Visible] = useState(true);
    const [isButton2Visible, setIsButton2Visible] = useState(true);
    const [isButton1Visible, setIsButton1Visible] = useState(true);
    const [isButton4Visible, setIsButton4Visible] = useState(true);
    const [isButton0Visible, setIsButton0Visible] = useState(true);


    console.log("selectedButtons:", selectedImages);

    const handleSubmit = async () => {
        const post = {
            code: selectedImages
        };

        try {
            const res = await axios.post(`${API_URL}/api/test`, post);
            console.log(res.data);
        } catch (e) {
            alert(e);
        }
    };

    const changeText = () => {
        if (text === "Choose your 2nd Option") {
            setText("Choose your 3rd Option");
        }
        else if (text === "Choose your 3rd Option"){
            setText("Choose your 4th Option");
        }
        else if (text === "Choose your 4th Option"){
            setText("Choose your last Option");
        }
        else{
            setText("Choose your 2nd Option");
        }
    }


```

```
64      const handleButton3Click = (imageSrc) => {
65          setIsButton3Visible(false);
66          changeText();
67          const updatedSelectedImages = [...selectedImages, imageSrc];
68          setSelectedImages(updatedSelectedImages);
69
70          const updatedSelectedImagePaths = [...selectedImagePaths, secondcolor];
71          setSelectedImagePaths(updatedSelectedImagePaths);
72
73          localStorage.setItem('res_num', JSON.stringify(updatedSelectedImages));
74          localStorage.setItem('image_paths', JSON.stringify(updatedSelectedImagePaths));
75        }
76
77
78
79      const handleButton2Click = (imageSrc) => {
80          setIsButton2Visible(false);
81          changeText();
82          const updatedSelectedImages = [...selectedImages, imageSrc];
83          setSelectedImages((prevSelectedImages) => [...prevSelectedImages, imageSrc]);
84
85          const updatedSelectedImagePaths = [...selectedImagePaths, lastcolor];
86          setSelectedImagePaths(updatedSelectedImagePaths);
87
88          localStorage.setItem('res_num', JSON.stringify(updatedSelectedImages));
89          localStorage.setItem('image_paths', JSON.stringify(updatedSelectedImagePaths));
90        }
91
92      const handleButton1Click = (imageSrc) => {
93          setIsButton1Visible(false);
94          changeText();
95          const updatedSelectedImages = [...selectedImages, imageSrc];
96          setSelectedImages((prevSelectedImages) => [...prevSelectedImages, imageSrc]);
97
98          const updatedSelectedImagePaths = [...selectedImagePaths, forthcolor];
99          setSelectedImagePaths(updatedSelectedImagePaths);
100
101         localStorage.setItem('res_num', JSON.stringify(updatedSelectedImages));
102         localStorage.setItem('image_paths', JSON.stringify(updatedSelectedImagePaths));
103       }
104     const handleButton4Click = (imageSrc) => {
105         setIsButton4Visible(false);
106         changeText();
107         const updatedSelectedImages = [...selectedImages, imageSrc];
108         setSelectedImages((prevSelectedImages) => [...prevSelectedImages, imageSrc]);
109
110         const updatedSelectedImagePaths = [...selectedImagePaths, firstcolor];
111         setSelectedImagePaths(updatedSelectedImagePaths);
112
113         localStorage.setItem('res_num', JSON.stringify(updatedSelectedImages));
114         localStorage.setItem('image_paths', JSON.stringify(updatedSelectedImagePaths));
115       }
116
117
118     const handleButton0Click = (imageSrc) => {
119         setIsButton0Visible(false);
120         changeText();
121         const updatedSelectedImages = [...selectedImages, imageSrc];
122         setSelectedImages((prevSelectedImages) => [...prevSelectedImages, imageSrc]);
123
124         const updatedSelectedImagePaths = [...selectedImagePaths, thirdcolor];
125         setSelectedImagePaths(updatedSelectedImagePaths);
126
127         localStorage.setItem('res_num', JSON.stringify(updatedSelectedImages));
128         localStorage.setItem('image_paths', JSON.stringify(updatedSelectedImagePaths));
129       }
130
```

```
131     return (
132         <Grid container>
133 
134             <Grid container justifyContent="center">
135                 <Grid item xs={12} xl={12} md={10} justifyContent="flex-start">
136                     <img src={Logo} alt='logo' className={classes.logodisplay} />
137                 </Grid>
138             </Grid>
139 
140             <div className={classes.content}>
141                 <Grid container justifyContent="center" >
142                     <Grid xs={4} xl={4} md={3.5}>
143 
144                         <form onSubmit={handleSubmit}>
145 
146                             {/* top left button 3  . onClick={handleButton3Click}*/}
147                             {isButton3Visible && (
148                                 <div>
149                                     <button className={classes.squarebutton0} onClick={() => handleButton3Click(3)}>
150                                         <img src={secondcolor} alt="second color" />
151                                     </button>
152                                     <Grid xs={1} xl={12} md={11}>
153                                         <div className={classes.textcontainer}>{text}</div>
154                                     </Grid>
155                                 </div>
156 
157                             )}
158                             {!isButton3Visible && (
159                                 <button className={classes.squarebutton0}  style={{ visibility: 'hidden' }}>
160 
161                                 </button>
162                             )}
163                         </form>
164                     </Grid>
165 
166                     <Grid xs={3.5} xl={1} md={1.5} >
167                         <form onSubmit={handleSubmit}>
168 
169                             {/* top right  button 4  onClick={handleButton4Click} */}
170                             {isButton4Visible && (
171 
172                                 <div>
173                                     <button className={classes.squarebutton1}  onClick={() => handleButton4Click(4)}>
174                                         <img src={firstcolor} alt="first color" />
175                                     </button>
176                                     <Grid xs={12} xl={12} md={12} justifyContent="center">
177                                         <div className={classes.textcontainer} >{text}</div>
178                                     </Grid>
179                                 </div>
180 
181                             )}
182                             {!isButton4Visible && (
183                                 <button className={classes.squarebutton1} style={{ visibility: 'hidden' }}>
184 
185                                 </button>
186 
187                             )}
```

```
<Grid container justifyContent="center" >
    <Grid item xs={4} xl={1} md={1.45} justifyContent="center">
        <form onSubmit={handleSubmit}>

            {/* center  button 0  onClick={handleButton0Click}  */}
            {isButton0Visible && (
                <div>
                    <button className={classes.squarebutton2} onClick={() => handleButton0Click(0)}>
                        <img src={thirdcolor} alt="third color" />
                    </button>
                    <Grid xs={12} xl={12} md={12} justifyContent="center">
                        <div className={classes.textcontainer}>{text}</div>
                    </Grid>
                </div>


            )}

            {!isButton0Visible && (
                <button className={classes.squarebutton2} style={{ visibility: 'hidden' }}>

                </button>
            )}
        </form>

    </Grid>
</Grid>

<Grid container justifyContent="center" >
    <Grid item xs={4} xl={4} md={3.5} >
        <form onSubmit={handleSubmit}>

            {/* bottom left  button 2*/}
            {isButton2Visible && (
                <div>
                    <button className={classes.squarebutton3} onClick={() => handleButton2Click(2)}>
                        <img src={lastcolor} alt="last color" />
                    </button>
                    <Grid xs={1} xl={12} md={12} justifyContent="center">
                        <div className={classes.textcontainer}>{text}</div>
                    </Grid>
                </div>

            )}
            {!isButton2Visible && (
                <button className={classes.squarebutton3} style={{ visibility: 'hidden' }}>

                </button>
            )}
        </form>
    </Grid>
```

```
                    <Grid xs={3.5} xl={1} md={1.5} >
                        <form onSubmit={handleSubmit}>

                            {/* bottom right  button 1*/}
                            {isButton1Visible && (
                                <div>
                                    <button className={classes.squarebutton4} onClick={() => handleButton1Click(1)}>
                                        <img src={forthcolor} alt="forth color" />
                                    </button>
                                    <Grid xs={12} xl={12} md={12} justifyContent="center">
                                        <div className={classes.textcontainer}>{text}</div>
                                    </Grid>
                                </div>

                            )}
                            {!isButton1Visible && (
                                <button className={classes.squarebutton4}  style={{ visibility: 'hidden' }}>

                                </button>
                            )}
                        </form>
                    </Grid>
                </Grid>
            </div>

            {!isButton2Visible && !isButton3Visible && !isButton4Visible && !isButton0Visible && !isButton1Visible && nextpage()}

        </Grid>
    );

}

export default TestPage;

```

1.4 Result.jsx:

```jsx
import Header from "../Header/Header";

import { Grid, Paper } from '@material-ui/core';

import classes from "./Results.module.css";

import firstcolor from '../../assets/firstcolor.jpg';
import secondcolor from '../../assets/secondcolor.PNG';
import thirdcolor from '../../assets/thirdcolor.PNG';
import forthcolor from '../../assets/forthcolor.PNG';
import lastcolor from '../../assets/lastcolor.jpg';

import { useNavigate } from "react-router-dom";
import { useEffect, useState } from "react";
import axios from "axios";
import { API_URL } from "../../api.js"


const Results = (props) => {

    const navigate = useNavigate();

    const homeBtn = () => {
        navigate("/bookmeeting");
    };

    const answer = {
        numbers: [1, 0, 2, 4, 3]
    }
    const accessToken = localStorage.getItem('token');
    // const images = localStorage.getItem('image_paths');


    const [images, setImages] = useState([]);
    useEffect(() => {
        const storedImages = JSON.parse(localStorage.getItem('image_paths')) || [];
        setImages(storedImages);
    }, []);
    // const images = JSON.parse(localStorage.getItem('image_paths')) || [];
    // const secondcolor2 = images[0];
// return (
//   <img src={images[0]} alt="first color" />
// );

    const [data, setData] = useState([]);
```

```
45      const [data, setData] = useState([]);
46
47      useEffect(() => {
48          const resNumArray = JSON.parse(localStorage.getItem('res_num'));
49
50          // Convert the array of res_num to a comma-separated string
51          const resNumString = resNumArray.join(',');
52
53          axios.post(`${API_URL}/api/test/view?res_num=[${resNumString}]`, {
54              // your post request data here
55          },
56          {
57              headers: {
58                  'Authorization': `Bearer ${accessToken}`
59              }
60          })
61              .then(response => {
62                  console.log(response.data); // make sure the response is what you expect
63                  setData(response.data.data); // set the data state with the response data
64              })
65              .catch(error => {
66                  console.log(error);
67              });
68      }, []);
69
70      console.log(images[0])
71
72
73      return (
74          <div>
75              <Header />
76              <Grid container className={classes.results}>
77
78                  {/* Header */}
79                  <Grid xs={12} xl={12} md={12} lg={12}>
80                      <Grid container justify="center">
81                          <Grid xs={12} xl={12} md={12} lg={12}>
82                              <div className={classes.header}>
83                                  <h1>Your results</h1>
84                                  <h2>Thank you for taking the test we hope you find this results insightful</h2>
85                              </div>
86                          </Grid>

99                  {/* Result */}
100                 <Grid xs={12} xl={12} md={12} lg={12}>
101                     <Grid className={classes.result}>
102
103                         {/* first result */}
104                         <Grid container>
105                             <Grid xs={12} xl={12} md={12} lg={12}>
106                                 <Grid container>
107                                     <Grid className={classes.images} xs={12} xl={4} md={4} lg={3}>
108                                         <Grid container>
109                                             <Grid xs={6} xl={6} md={6} lg={6}>
110                                                 <img src={`${images[0]}`} alt="first color" />
111                                             </Grid>
112                                             <Grid xs={6} xl={6} md={6} lg={6}>
113                                                 <h4>First color insights</h4>
114                                             </Grid>
115                                         </Grid>
116                                         {/* <img src={secondcolor} className={classes.imageresult} alt="second color" /> */}
117                                     </Grid>
118
119                                     <Grid className={classes.imagesresult} xs={12} xl={6} md={6} lg={6}>
120                                         <p>
121                                             {data.map((item, index) => (
122                                                 <div key={index}>
123                                                     <p>{item[0]}</p>
124                                                 </div>
125                                             ))[0]}
126                                         </p>
127                                     </Grid>
128                                 </Grid>
129                             </Grid>
```

```
                    </Grid>

                    {/* second result */}
                    <Grid container>
                        <Grid xs={12} xl={12} md={12} lg={12}>
                            <Grid container>
                                <Grid className={classes.images} xs={12} xl={4} md={4} lg={3}>
                                    <Grid container>
                                        <Grid xs={6} xl={6} md={6} lg={6}>
                                            <img src={`${images[1]}`} alt="second color" />
                                        </Grid>
                                        <Grid xs={6} xl={6} md={6} lg={6}>
                                            <h4>Second color insights</h4>
                                        </Grid>
                                    </Grid>
                                    {/* <img src={secondcolor} className={classes.imageresult} alt="second color" /> */}
                                </Grid>

                                <Grid className={classes.imagesresult} xs={12} xl={6} md={6} lg={6}>
                                    <p>
                                        {data.map((item, index) => (
                                            <div key={index}>
                                                <p>{item[0]}</p>
                                            </div>
                                        ))[1]}
                                    </p>
                                </Grid>
                            </Grid>
                        </Grid>
                    </Grid>

                    {/* third result */}
                    <Grid container>
                        <Grid xs={12} xl={12} md={12} lg={12}>
                            <Grid container>
                                <Grid className={classes.images} xs={12} xl={4} md={4} lg={3}>
                                    <Grid container>
                                        <Grid xs={6} xl={6} md={6} lg={6}>
                                            <img src={`${images[2]}`} alt="third color" />
                                        </Grid>
                                        <Grid xs={6} xl={6} md={6} lg={6}>
                                            <h4>Third color insights</h4>
                                        </Grid>
                                    </Grid>
                                    {/* <img src={secondcolor} className={classes.imageresult} alt="second color" /> */}
                                </Grid>

                                <Grid className={classes.imagesresult} xs={12} xl={6} md={6} lg={6}>
                                    <p>
                                        {data.map((item, index) => (
                                            <div key={index}>
                                                <p>{item[0]}</p>
                                            </div>
                                        ))[2]}
                                    </p>
                                </Grid>
                            </Grid>
                        </Grid>
                    </Grid>
```

5.1.2 Website Back End:

2.1 Models:

2.1.1 Appointment:

```php
class Appointment extends Model
{
    use HasFactory;

    protected $fillable = [
        'patient_id',
        'config_id',
        'date',
        'note',

    ];
    protected $casts = [ 'date'=>'datetime'];
    /**
     * Get the user that owns the Configuration
     *
     * @return \Illuminate\Database\Eloquent\Relations\BelongsTo
     */
    public function config(): BelongsTo
    {
        return $this->belongsTo(Configuration::class, 'config_id');
    }
     /**
     * Get the user that owns the patient
     *
     * @return \Illuminate\Database\Eloquent\Relations\BelongsTo
     */
    public function patient(): BelongsTo
    {
        return $this->belongsTo(Patient::class, 'patient_id');
    }
}

```

2.1.2 Article:

```php
class Artical extends Model
{
    protected $table = 'articles';
    protected $fillable = [
        'title',
        'body',
        'img',
        'doctor_id',
        'description',
    ];
    /**
     * Get the doctor that owns the Artical
     *
     * @return \Illuminate\Database\Eloquent\Relations\BelongsTo
     */
    public function doctor(): BelongsTo
    {
        return $this->belongsTo(Doctor::class, 'doctor_id');
    }
}
```

2.1.3 Configuration:

```php
class Configuration extends Model
{
    use HasFactory;

    protected $table = 'configuration_doctors';

    protected $fillable = [
        'doctor_id',
        'day',
        'from',
        'to',
        'fees',
        'location',
        'type',
        'secretary_id',
        'slug',
    ];
    protected $dates = ['from', 'to'];
    /**
     * Get the user that owns the Configuration
     *
     * @return \Illuminate\Database\Eloquent\Relations\BelongsTo
     */
    public function doctor(): BelongsTo
    {
        return $this->belongsTo(Doctor::class, 'doctor_id');
    }
     /**
     * Get the user that owns the Configuration
     *
     * @return \Illuminate\Database\Eloquent\Relations\BelongsTo
     */
    public function secretary(): BelongsTo
    {
        return $this->belongsTo(Secretary::class, 'secretary_id');
    }
}
```

2.1.4 Doctor:

```php
class Doctor extends Model
{
    //use  HasFactory, Notifiable, HasApiTokens;
    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'user_id',
        'info',
        'bio',
    ];
    /**
     * Get the user that owns the Doctor
     *
     * @return \Illuminate\Database\Eloquent\Relations\BelongsTo
     */
    public function user(): BelongsTo
    {
        return $this->belongsTo(User::class, 'user_id');
    }

}
```

2.1.5 Patient:

```php
class Patient extends Model
{
    //use  HasFactory, Notifiable, HasApiTokens;
    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'user_id',
        'age',
        'current_city',
        'original_city',
        'gender',
        'status',
        'medication',
        'resources',
        'language',
        'problem',
    ];

 /**
     * Get the user that owns the Doctor
     *
     * @return \Illuminate\Database\Eloquent\Relations\BelongsTo
     */
    public function user(): BelongsTo
    {
        return $this->belongsTo(User::class, 'user_id');
    }

}
```

2.1.6 Secretary:

```php
class Secretary extends Model
{
    //use  HasFactory, Notifiable, HasApiTokens;
    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'user_id',
    ];

 /**
     * Get the user that owns the Doctor
     *
     * @return \Illuminate\Database\Eloquent\Relations\BelongsTo
     */
    public function user(): BelongsTo
    {
        return $this->belongsTo(User::class, 'user_id');
    }

}
```

2.1.7 User:

```php
class User extends Authenticatable implements JWTSubject
{
    use HasApiTokens, HasFactory, Notifiable;

    /**
     * The attributes that are mass assignable.
     *
     * @var array<int, string>
     */
    protected $fillable = [
        'name',
        'email',
        'password',
        'phone_num',
        'type',
        'img',
    ];

    /**
     * The attributes that should be hidden for serialization.
     *
     * @var array<int, string>
     */
    protected $hidden = [
        'password',
        'remember_token',
    ];

    /**
     * The attributes that should be cast.
     *
     * @var array<string, string>
     */
    protected $casts = [
        'email_verified_at' => 'datetime',
    ];

```

```php
/**
     * Get the user associated with the User
     *
     * @return \Illuminate\Database\Eloquent\Relations\HasOne
     */
    public function doctor(): HasOne
    {
        return $this->hasOne(Doctor::class);
    }

    /**
     * Get the secretary associated with the User
     *
     * @return \Illuminate\Database\Eloquent\Relations\HasOne
     */
    public function secretary(): HasOne
    {
        return $this->hasOne(Secretary::class);
    }

    /**
     * Get the patient associated with the User
     *
     * @return \Illuminate\Database\Eloquent\Relations\HasOne
     */
    public function patient(): HasOne
    {
        return $this->hasOne(Patient::class);
    }
    /**
     * Get the identifier that will be stored in the subject claim of the JWT.
     *
     * @return mixed
     */
    public function getJWTIdentifier()
    {
        return $this->getKey();
    }
    /**
     * Return a key value array, containing any custom claims to be added to the JWT.
     *
     * @return array
     */
    public function getJWTCustomClaims()
    {
        return [];
    }
}
```

2.2 Controller:

2.2.1 API response trait:

```php
trait ApiResponseTrait
{
    public function apiResponse($data =null,$message=null,$status=null){
        $array =[
            'data'=>$data,
            'message'=>$message,
            'status'=>$status
        ];


        //return response()->json($array,$status);
        return response($array,$status);
    }
}

```

2.2.2 Controller API:

2.2.2.1 Appointment Controller:

2.2.2.1.1 insert Appointment:

```php
public function store(Request $request)
    {
        //validation
        $validator = Validator::make($request->all(), [
            'app_id' =>'required',
            'date' => 'required',
        ]);
        if ($validator->fails()) {
            return $this->apiResponse(null, $validator->errors(), 400);
        }
        // -------------
        //get patient id
        $patient = Patient::where('user_id', '=', Auth::id())->first();
        $config = Configuration::find($request->app_id);
        if ($patient) {
            $appointment = Appointment::create([
                'patient_id' => $patient->id,
                'config_id' => $request->app_id,
                'date' => $request->date,
            ]);
        }
        if ($appointment) {
            return $this->apiResponse($appointment, 'saved', 201);
        }
        return $this->apiResponse(null, 'appointment not save', 400);
    }
```

2.2.2.1.2 Insert Articles:

```php
public function store(Request $request){     //insert data
        //validation
        $validator = Validator::make($request->all(), [
            'doctor_id' => 'required',
            'title' => 'required|max:255',
            'body' => 'required',
            'img' => 'required|image|mimes:png,jpg,gif,jpeg,svg|max:2048',
            'description' =>'required',
        ]);
        if ($validator->fails()) {
            return $this->apiResponse(null,$validator->errors(),400);
        }
        // -------------
        if($request->hasFile('img'))
        {
            $file =$request->file('img');
            $extension = $file->getClientOriginalExtension();
            $fileName = time().'.'.$extension;
            $file->move('img/articals/',$fileName);
        }
        //--------
        $artical = Artical::create([
            'doctor_id' => $request->doctor_id,
            'title' => $request->title,
            'body' => $request->body,
            'description' => $request->description,
            'img' => 'img/articals/'.$fileName,
        ]);
        if($artical){
            return $this->apiResponse($artical,'saved',201);
        }
        return $this->apiResponse(null,'artical not save',400);
    }
```

2.2.2.1.3 Show all Configs:

```php
public function index()
{
    $config =  ConfigResource::collection(Configuration::get());
    $app =  Appointment::get();

    return $this->apiResponse($config, 'ok', 200); //PostResource:tore turn same of data

}
```

2.2.2.1.4 Show Doctor Appointment:

```php
public function doctorAppointments()
    {
        $doctor = Doctor::where('user_id', '=', Auth::id())->first();
        $doctor_id = $doctor->id;
        $today = date('Y-m-d');
        //$day_after = date('Y-m-d', strtotime('+6 days', strtotime($today))); // get 6 days after
        $appointments = Appointment::where('doctor_id', $doctor_id)
            ->where('date', '>=', $today)
            // ->where('date', '<=', $day_after)
            ->join('configuration_doctors', 'appointments.config_id', '=', 'configuration_doctors.id')
            ->join('patients', 'appointments.patient_id', '=', 'patients.id')
            ->join('users', 'patients.user_id', '=', 'users.id')
            ->orderBy('appointments.date')
            ->orderBy('configuration_doctors.from')
            ->select('appointments.id', 'users.name', 'users.img', 'configuration_doctors.day', 'appointments.date', 'from', 'to')
            ->get();
        return $this->apiResponse($appointments, 'done', 200);
    }
```

2.2.2.1.5 show doctor config in 1 week:

```php
public function show(Request $request)
    {
        $validator = Validator::make($request->all(), [
            'doctor_id' => 'required', // doctor id
            'date' => 'required', // today date
        ]);
        if ($validator->fails()) {
            return $this->apiResponse(null, $validator->errors(), 400);
        }
        //--------
        $config =  Configuration::where('doctor_id', '=', $request->doctor_id)->get();
        if ($config) {
            $date_before = date('Y-m-d', strtotime('+7 days', strtotime($request->date))); // get 6 days before
            $appointment = Appointment::where('date', '>', $date_before)->get();
            if ($appointment) {
                $new_config = [];
                // chick if the appointment is taken or not
                foreach ($config as $con) {
                    $chick = 0;
                    foreach ($appointment as $app) {
                        $dt = $app->date->format('l');
                        if ($con->day == $dt && $con->from == $app->from) {
                            $chick = 1; // to chick 1=> true 0=> false
                        }
                    }
                    if ($chick == 0) {
                        $new_config[] = $con;
                    }
                }
                return $this->apiResponse(ConfigResource::collection($new_config), 'ok', 200);
            } else {
                return $this->apiResponse(ConfigResource::collection($config), 'ok', 200);
            }
        } else {

            return $this->apiResponse(null, 'No time', 404);
        }
    }
```

2.2.2.2 Articles Controller:

2.2.2.2.1 Show all and 1 Articles:

```php
public function index(){
    $articals =ArticalResource::collection(Artical::get()); //data // return all data
    return $this->apiResponse($articals,'',200);
}
public function show($id){
    //$post = Post::find($id); // find to search
    $artical = Artical::find($id);
    if($artical){
        return $this->apiResponse(new ArticalResource($artical),'ok',200); //articalResource:tore turn same of data
    }
    return $this->apiResponse(null,'the artical not found',404);
}
```

2.2.2.2.2 Store Articles:

```php
public function store(Request $request){     //insert data
        //validation
        $validator = Validator::make($request->all(), [
            'doctor_id' => 'required',
            'title' => 'required|max:255',
            'body' => 'required',
            'img' => 'required|image|mimes:png,jpg,gif,jpeg,svg|max:2048',
            'description' =>'required',
        ]);
        if ($validator->fails()) {
            return $this->apiResponse(null,$validator->errors(),400);
        }
        // -------------
        if($request->hasFile('img'))
        {
            $file =$request->file('img');
            $extension = $file->getClientOriginalExtension();
            $fileName = time().'.'.$extension;
            $file->move('img/articals/',$fileName);
        }
        //--------
        $artical = Artical::create([
            'doctor_id' => $request->doctor_id,
            'title' => $request->title,
            'body' => $request->body,
            'description' => $request->description,
            'img' => 'img/articals/'.$fileName,
        ]);
        if($artical){
            return $this->apiResponse($artical,'saved',201);
        }
        return $this->apiResponse(null,'artical not save',400);
    }
```

2.2.2.3 auth Controller:

2.2.2.3.1 login:

```php
public function login(Request $request)
    {
        //validation
        $validator = Validator::make($request->all(), [
            'email' => 'required|email',
            'password' => 'required|string|min:6',
        ]);

        if ($validator->fails()) {
            return response()->json($validator->errors(), 400);
        }

        //------
        if (!$token = auth()->attempt($validator->validated())) {
            return response()->json(['error' => 'Unauthorized'], 401);
        }


        return $this->createNewToken($token);
    }
```

2.2.2.3.2 logout & refresh token:

```php
/**
 * Log the user out (Invalidate the token).
 *
 * @return \Illuminate\Http\JsonResponse
 */
public function logout()
{
    auth()->logout();
    return response()->json(['message' => 'User successfully signed out']);
}

/**
 * Refresh a token.
 *
 * @return \Illuminate\Http\JsonResponse
 */
public function refresh()
{
    return $this->createNewToken(auth::refresh(), null);
}
```

2.2.2.3.3 Register:

```php
public function register(Request $request)
    {
        //validation
        $validator = Validator::make($request->all(), [
            'name' => 'required|string|between:2,100',
            'email' => 'required|string|email|max:100|unique:users',
            'password' => 'required|string|min:6',
            'type' => 'required',
            'orignal_city' => 'required|string',
            'current_city' => 'required|string',
        ]);
        if ($validator->fails()) {
            return response()->json($validator->errors()->toJson(), 401);
        }
        //insert into user table first
        $user = User::create(array_merge(
            $validator->validated(),
            ['password' => bcrypt($request->password),
            'img'=> 'img/articals/1683667622.png']
        ));

        $id = $user->id;
        if ($request['type'] == 'patient') {
             Patient::create([
                'user_id' => $id,
                'original_city' => $request->orignal_city,
                'current_city' => $request->current_city,
            ]);

            return response()->json([
                'message' => 'User successfully registered',
                'user' => $user,
            ], 201);
        }
        return response()->json("You are not patient", 401);

    }
```

2.2.2.4 Test Controller:

2.2.2.4.1 Show test result:

```php
public function view(Request $request)
{
     //validation
     $validator = Validator::make($request->all(), [
        'res_num' => 'required',
    ]);
    if ($validator->fails()) {
        return $this->apiResponse(null,$validator->errors(),400);
    }
    // -------------
    $res_text=[];
    $cont=1;
    for ($i=1; $i <= 10 ; $i++) {
        if ($cont == 5) {
            $cont =1;
        }
        $data_res = TestData::where('id',$i)->pluck($request->res_num[$cont]);
        $res_text[]= $data_res;
        $cont= $cont+2;
    }
    if($res_text){
        return $this->apiResponse($res_text,'ok',200);
    }
    return $this->apiResponse(null,'the restData not found',404);
}

```

2.3 Route:

2.3.1 API:

```php
Route::group([
    'middleware' => 'api',
    'prefix' => 'auth',
], function ($router) {
    Route::post('/login', [AuthController::class, 'login']);
    Route::post('/register', [AuthController::class, 'register']);
    Route::post('/logout', [AuthController::class, 'logout']);
    Route::post('/refresh', [AuthController::class, 'refresh']);
    Route::get('/user-profile', [AuthController::class, 'userProfile']);
});
Route::middleware(['jwt.verify'])->group(function () {
    //appointment
    Route::get('/appointments', [AppointmentController::class, 'index']); // show all data
    Route::get('/doctors', [AppointmentController::class, 'doctorView']); // show all doctors
    Route::get('/appointment/show', [AppointmentController::class, 'show']); // store appointment
    Route::post('/appointment/store', [AppointmentController::class, 'store']); // store appointment
    Route::get('/doctor_appointments', [AppointmentController::class, 'doctorAppointments']); // to mobile app

    //test
    Route::post('/test/view', [TestController::class, 'view']);
});
//articals
Route::get('/articals', [ArticalController::class, 'index']); // show all data
Route::get('/artical/{id}', [ArticalController::class, 'show']); // show 1
Route::Post('/artical', [ArticalController::class, 'store']); // insert artical
```

2.3.1 Web:

```php
//Routes for configuration doctor controller
Route::get('/configs', [ConfigurationController::class, 'index'])->name('configs');            // show all
Route::get('/config/create', [ConfigurationController::class, 'create'])->name('config.create'); // create new
Route::post('/config/store', [ConfigurationController::class, 'store'])->name('config.store'); // store in database
Route::get('/config/show/{slug}', [ConfigurationController::class, 'show'])->name('config.show');  // show 1
Route::get('/config/{id}', [ConfigurationController::class, 'edit'])->name('config.edit');         // action edit
Route::post('/config/update/{id}', [ConfigurationController::class, 'update'])->name('config.update'); // update 1
Route::delete('/config/destroy/{id}', [ConfigurationController::class, 'destroy'])->name('config.destroy'); // delete



//Routes for appointments controller
Route::get('/appointments', [AppointmentController::class, 'index'])->name('appointments');            // show all
Route::get('/appointment/create', [AppointmentController::class, 'create'])->name('appointment.create'); // create new
Route::post('/appointment/store', [AppointmentController::class, 'store'])->name('appointment.store'); // store in database
Route::get('/appointment/show/{id}', [AppointmentController::class, 'show'])->name('appointment.show');  // show 1
Route::get('/appointment/{id}', [AppointmentController::class, 'edit'])->name('appointment.edit');         // action edit
Route::post('/config/appointment/{id}', [AppointmentController::class, 'update'])->name('appointment.update'); // update 1
Route::delete('/appointment/destroy/{id}', [AppointmentController::class, 'destroy'])->name('appointment.destroy'); // delete


Auth::routes();
Route::get('/', [App\Http\Controllers\HomeController::class, 'index'])->name('home');
//Route::get('/main', [App\Http\Controllers\HomeController::class, 'main'])->name('main');

Route::get('/layout', function () {
    return view('layouts/app');
});

```

5.2 Chatbot Code:

```python
# Import the necessary libraries
import streamlit as st
import random
import webbrowser
import time

st.set_page_config(menu_items=None,page_title="Inflow Chatbot", page_icon="path/to/your/favicon.ico", initial_sidebar_state="collapsed")


with open('style.css') as f:
    st.markdown(f'<style>{f.read()}</style>', unsafe_allow_html=True)


# Define a class to hold the session state variables
class SessionState:


    def __init__(self):

        self.current_state = "start"
        self.previous_state = "start" # Set the initial state to "start"


def greeting():
    if "state" not in st.session_state:   # Check if the session state has been initialized
        st.session_state.state = SessionState()   # Initialize the session state
    state = st.session_state.state        # Get the current state from the session state object

    greetings = ["Hello", "Hi", "Sup", "Hey there"]
    greeting_text = random.choice(greetings)

    greetingss = ["How can I assist you today?", "How can I help you today?", "What can i do for you today?", "Hope you are doing fine, How can i help u?"]
    greetings_text = random.choice(greetingss)

    st.title("Welcome to Inflow Chatbot!")
    st.write("I'm an ai model here to help you with whatever you need.")
    name = st.text_input("Before we get started, may I know your name?")
    if name:

        st.write(f"{greeting_text}, {name}!, {greetings_text}")
        state.current_state = "yesintstate"

```

```python
# Define the main chat function
def chat():
      # Set the width and height of the chat panel
    panel_width = 600
    panel_height = 400

    # Calculate the necessary CSS styles
    css_code = f"""
        <style>
        /* Center the chat panel */
        .element-container:nth-child(1) {{
            display: flex;
            justify-content: center;
            align-items: center;
            height: 00vh;
        }}

        /* Set the panel width and height */
        .element-container:nth-child(1) .stApp {{
            width: {panel_width}px;
            height: {panel_height}px;
            overflow-y: auto;
        }}
        </style>
    """

    # Render the CSS code using st.markdown
    st.markdown(css_code, unsafe_allow_html=True)

    # JavaScript code to scroll to the bottom of the chat panel
    scroll_js = """
        <script>
        const chatPanel = document.querySelector(".element-container:nth-child(1) .stApp");
        chatPanel.scrollTop = chatPanel.scrollHeight;
        </script>
    """

```

```
# st.title("Welcome to the Chatbot!")   # Set the title of the app
    if "state" not in st.session_state:   # Check if the session state has been initialized
        st.session_state.state = SessionState()   # Initialize the session state
    state = st.session_state.state        # Get the current state from the session state object
    if state.current_state == "start":    # If the current state is "start"
        if __name__ == "__main__":
            greeting()
        # st.write("Hey there, how can I help you? Are you interested to know more about our Website?? ")
            # if st.button("Yes am interested"):   # If the "Yes" button is clicked
            #     state.current_state = "yesintstate"   # Change the current state to "yesintstate"
    if state.current_state == "yesintstate":   # If the current state is "yesintstate"
        st.write("<h5><b>Select one:</b></h5>" , unsafe_allow_html=True)
        if st.button(" About us "):         # If the "About us" button is clicked
            state.current_state = "aboutusstate"
            st.session_state.answer = "yes"# Change the current state to "aboutusstate"
        if st.button(" What is luscher color test ? "):   # If the "What we service" button is clicked
            state.current_state = "servstate"
            st.session_state.answer = "yes"# Change the current state to "servstate"
        if st.button("How to Make a Appointment"):                # If the "Test" button is clicked
            state.current_state = "howtomakeappo"
            # st.session_state.answer = "yes"# Change the current state to "lol"
        if st.button("Contact us"):               # If the "Test" button is clicked
            state.current_state = "contactus"
```

5.3 Data Analysis code:

4.1 Data Validation and Cleansing:

In [12]:
```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import plotly.express as px
import seaborn as sns
import arabic_reshaper
from bidi.algorithm import get_display
```

In [2]:
```
data = pd.read_excel("patents_Data.xlsx")
```

In [3]:
```
data.head()
```

Out[3]:

	Gender	Age	Current_city	Status	No.Children	Problem	Code	Note	Type of Metting
0	Male	32	Morocco	Single	0	Psychological	4312	يشك بالأخرين	Offline
1	Female	30	KSA	Married	2	Organic	41302	يشك بالأخرين	Offline
2	Female	30	KSA	Married	2	Relationship	1432	دبلوماسي انتهازى	Offline
3	Female	36	UAE	Married	2	Organic\nRelationship	41032	يشك بالأخرين	Offline
4	Male	42	UAE	Married	2	Relationship	43012	يشك بالأخرين	Offline

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41 entries, 0 to 40
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   Gender           41 non-null     object
 1   Age              41 non-null     int64 
 2   Current_city     41 non-null     object
 3   Status           41 non-null     object
 4   No.Children      41 non-null     int64 
 5   Problem          41 non-null     object
 6   Code             41 non-null     int64 
 7   Note             41 non-null     object
 8   Type of Metting  41 non-null     object
dtypes: int64(3), object(6)
memory usage: 3.0+ KB
```

DATA Validation:

```
In [5]: data["Gender"].unique()
```

```
Out[5]: array(['Male', 'Female', 'Female '], dtype=object)
```

```
In [6]: d = {'Female ' : 'Female' }
        data.replace({"Gender": d}, inplace = True)
```

```
In [7]: data["Gender"].value_counts()
```

```
Out[7]: Female    31
        Male      10
        Name: Gender, dtype: int64
```

```
In [8]: data["Type of Metting"].value_counts()
```

```
Out[8]: Online     28
        Offline    13
        Name: Type of Metting, dtype: int64
```

```
In [9]: #I enhanced it using Excel:
        data["Note"].value_counts()
```

```
Out[9]: 11                    استحواذي
        7                 يشك بالأخرين
        5                  يشك في ذاته
        3           دبلوماسي انتهازي
        3                    استعراضي
        3                       متمرد
        3                  احتياج شديد
        2      طموح متوتر لازم احقق
        2                    عصبي شرس
        1                  شديد التعلق
        1                     ديكتاتور
        Name: Note, dtype: int64
```

In [10]:
```
data["Note"].unique()
```

Out[10]:
```
array(['استعراضي' , 'دبلوماسي انتهازي' , 'يشك بالأخرين',
       'طموح متوتر لازم احقق' , 'استحواذي' , 'متمرد' , 'يشك في ذاته',
       'شديد التعلق' , 'احتياج شديد' , 'عصبي شرس' , 'ديكتاتور'], dtype=object)
```

In [13]:
```
# reshape the arabic word to be shown at correct way.
x = [ ]
for item in data["Note"].values:
    x.append(get_display(arabic_reshaper.reshape(item)))
```

In [14]:
```
 data["Problem"].unique()
```

Out[14]:
```
array(['Psychological ', 'Organic', 'Relationship',
       'Organic\nRelationship', 'Organic ', 'Relationship ',
       'Relationship  ', 'Physical ', 'relationship ', 'physical ',
       'psychological ', 'Self, Work', 'self regulation ', 'psychology '],
      dtype=object)
```

In [15]:
```
d = {"Relationship" : "Psychological" ,
     "relationship " : "Psychological" ,
     "Organic": "Physical" , "Physical " : "Physical",
     "Relationship " : "Psychological", "physical " : "Physical",
     "psychological " : "Psychological" , "Psychological " : "Psychological" ,
     "Organic\nRelationship" : "Physical\Psychological" , "Organic ": "Physical" ,
     "Relationship  " : "Psychological" , "Self, Work" : "Psychological" ,
     "self regulation " : "Psychological" , "psychology " : "Psychological"}
data.replace({"Problem": d}, inplace = True)
```

In [16]:
```
data["Problem"].value_counts()
```

Out[16]:
```
Psychological             28
Physical                  12
Physical\Psychological     1
Name: Problem, dtype: int64
```

In [17]:
```
data["Current_city"].unique()
```

Out[17]:
```
array(['Morocco', 'KSA', 'UAE', 'Palestine', 'Kuwait', 'Alx', 'Dubai',
       'Abu Dhabi', 'Cairo', 'UK', 'Qatar', 'Cairo ', 'Iraq', 'cairo ',
       'Canada', 'Egypt'], dtype=object)
```

In [18]:
```
d = {"Alx" : "Egypt" ,
     'Dubai' : 'UAE' ,
     'Abu Dhabi' : 'UAE' ,
     "Cairo" : "Egypt" ,
     "Cairo " : "Egypt",
     "cairo " : "Egypt" }
data.replace({"Current_city": d}, inplace = True)
```

In [19]:
```
data["Current_city"].value_counts()
```

Out[19]:
```
Egypt        13
KSA           8
UAE           7
Morocco       3
Palestine     3
Kuwait        3
UK            1
Qatar         1
Iraq          1
Canada        1
Name: Current_city, dtype: int64
```

In [20]:
```
data["Status"].unique()
```

Out[20]:
```
array(['Single', 'Married', 'Single '], dtype=object)
```

In [21]:
```
d = {"Single " : "Single" }
data.replace({"Status": d}, inplace = True)
```

In [18]:
```
data["Status"].unique()
```

Out[18]:
```
array(['Single', 'Married', 'Single '], dtype=object)
```

In [19]:
```
d = {"Single " : "Single" }
data.replace({"Status": d}, inplace = True)
```

In [20]:
```
data["Status"].value_counts()
```

Out[20]:
```
Married    22
Single     19
Name: Status, dtype: int64
```

In [21]:
```
data.drop("No.Children", axis=1, inplace=True)
```

In [22]:
```
def age_avg(x):
    if( 0<= x <= 12 ):
        return "Childhood"
    if( 13 <= x <= 20 ):
        return "Teenager"
    if( 21 <= x <= 30 ):
        return "Adulthood"
    if( 31 <= x <= 50 ):
        return "Middle-age"
    else:
        return "old-age"
```

In [23]:
```
data['age_avg'] = data['Age'].apply(lambda x: age_avg(x))
data.head()
```

Out[23]:

	Gender	Age	Current_city	Status	Problem	Code	Note	Type of Metting	age_avg
0	Male	32	Morocco	Single	Psychological	4312	يشك بالأخرين	Offline	Middle-age
1	Female	30	KSA	Married	Physical	41302	يشك بالأخرين	Offline	Adulthood
2	Female	30	KSA	Married	Psychological	1432	دبلوماسي انتهازي	Offline	Adulthood
3	Female	36	UAE	Married	Physical\Psychological	41032	يشك بالأخرين	Offline	Middle-age

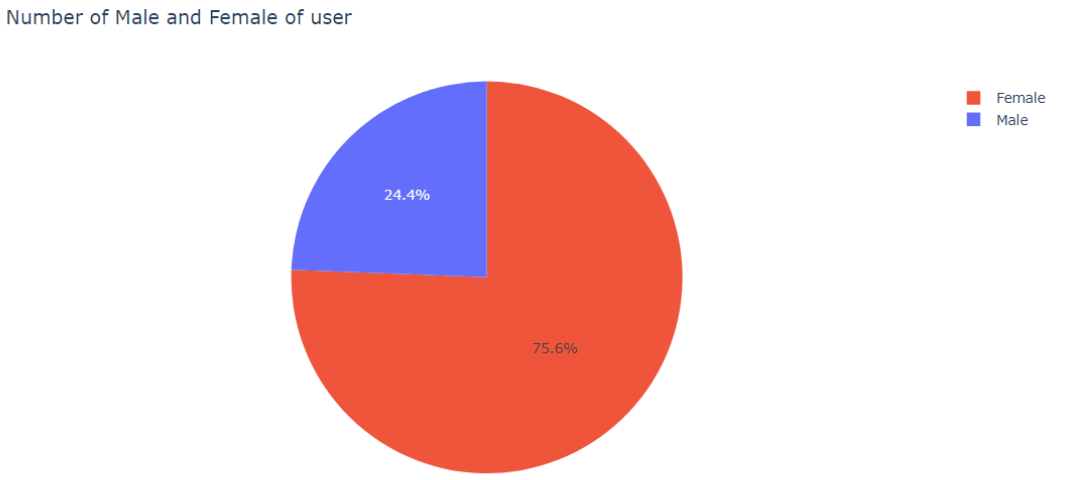
In [26]:
```
data['age_avg'].value_counts()
```

Out[26]:
```
Middle-age    17
Adulthood     17
old-age        3
Teenager       2
Childhood      2
Name: age_avg, dtype: int64
```

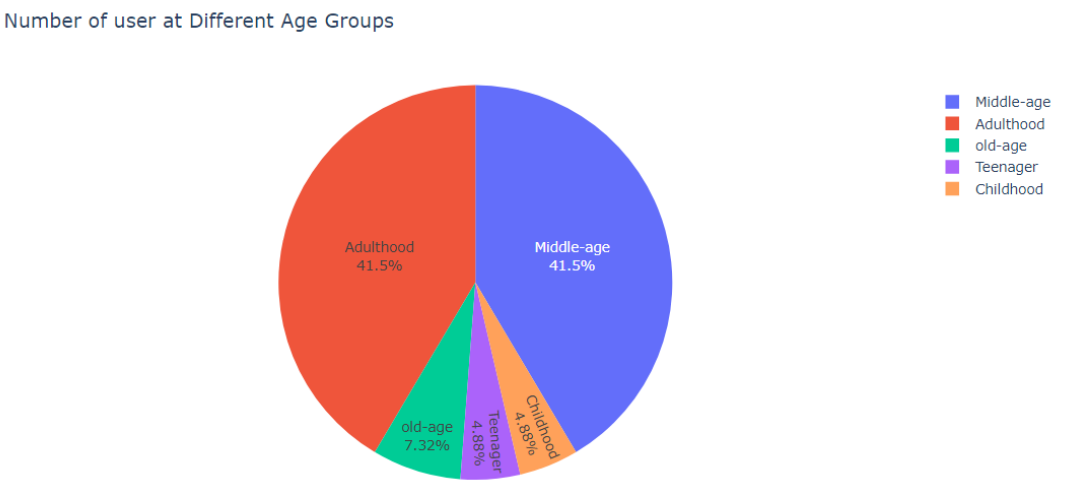
In [27]:
```
data.to_excel("C:\\Users\\Abdelrahman\\Documents\\data analysis\\INFLOW\\data.xlsx")
```

4.2 Data Visualization and Insight:

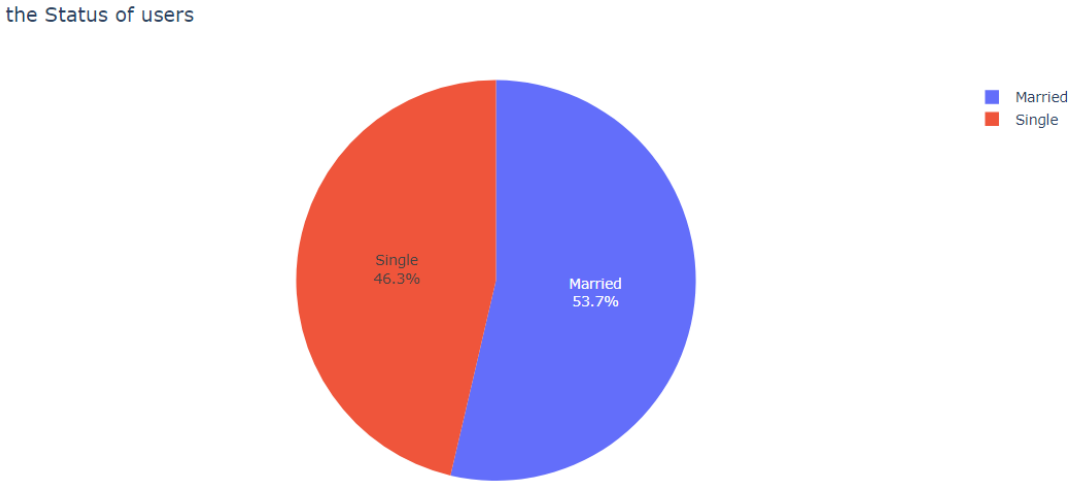
```
In [28]: fig = px.pie(data, names='Gender', color='Gender', title="Number of Male and Female of user" )
         fig.show()
```



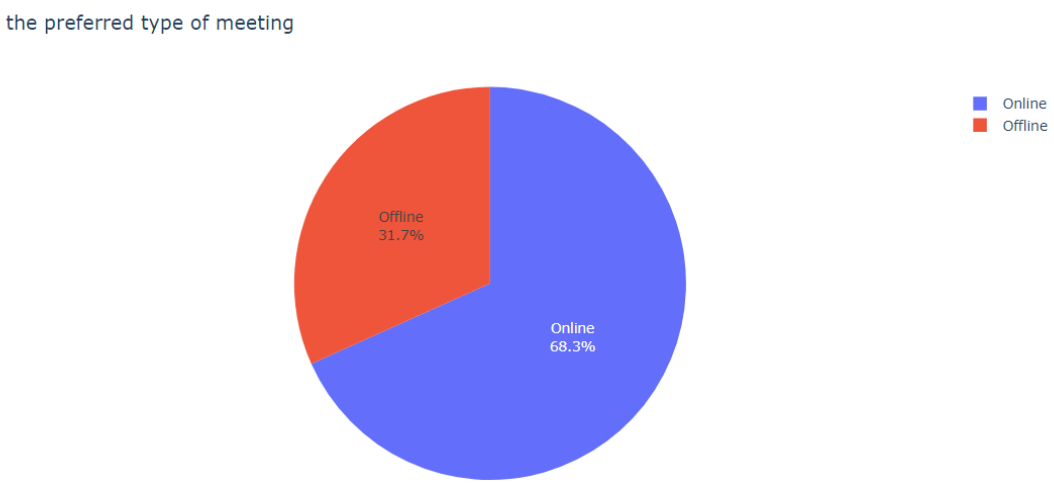

```
In [29]: fig = px.pie(data, names='age_avg', title="Number of user at Different Age Groups" )
         fig.update_traces(textinfo="label+percent")
         fig.show()
```



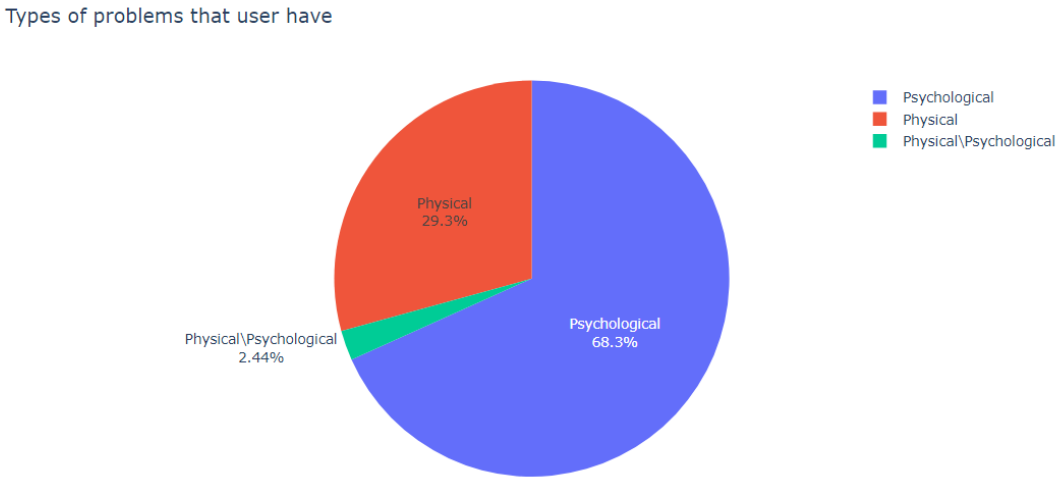
In [30]:
```
fig = px.pie(data, names='Status', title="the Status of users" )
fig.update_traces(textinfo="label+percent")
fig.show()
```



In [31]:
```
fig = px.pie(data, names='Type of Metting', title="the preferred type of meeting" )
fig.update_traces(textinfo="label+percent")
fig.show()
```



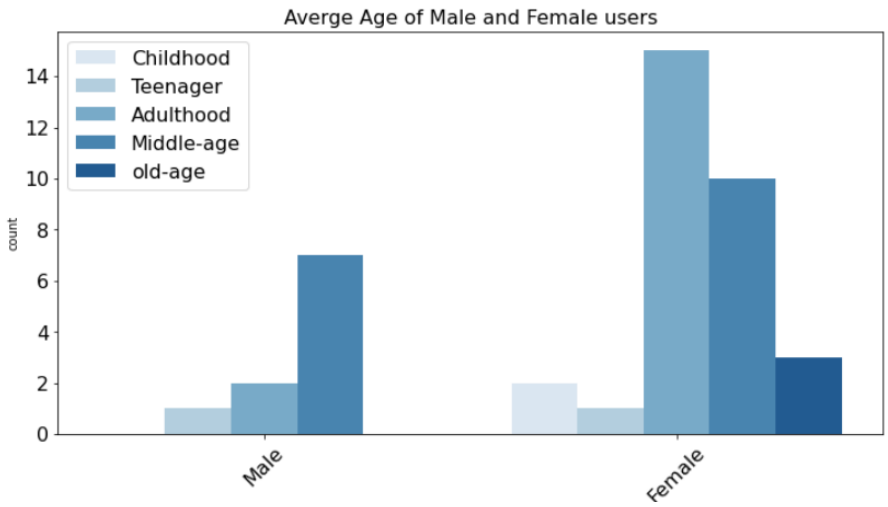
In [45]:
```
fig = px.pie(data, names='Problem' , title="Types of problems that user have")
fig.update_traces(textinfo="label+percent")
fig.show()
```



In [46]:
```
plt.figure(figsize=(12,6))
g = sns.countplot(data["Gender"], hue = data["age_avg"],
                  hue_order = ['Childhood', 'Teenager', 'Adulthood','Middle-age','old-age'], palette='Blues')
g = g.set_title("Averge Age of Male and Female users", fontsize=16)
plt.xticks(fontsize=16,rotation = 45)
plt.yticks(fontsize=16)
plt.legend(fontsize=16)
```

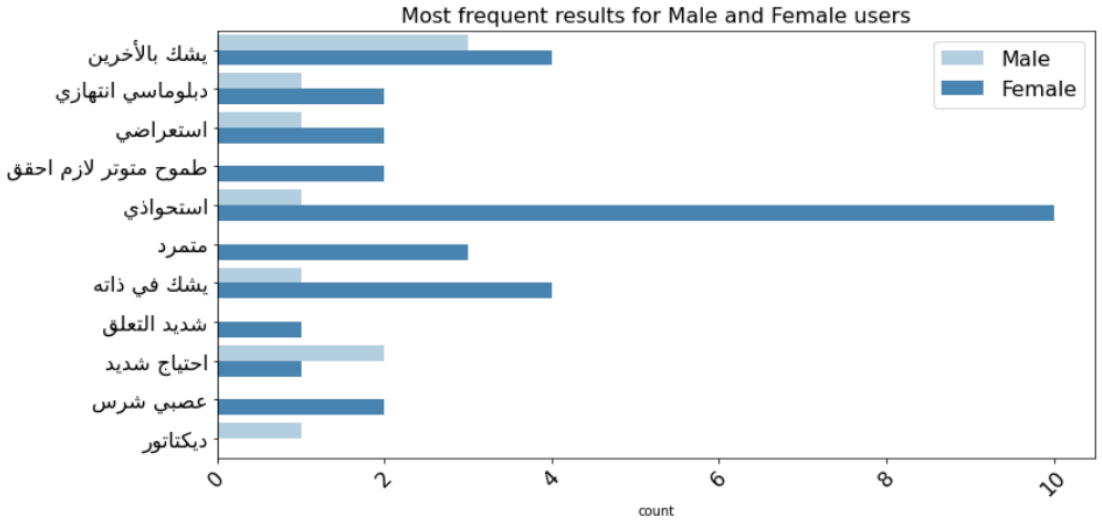
```
C:\Users\Abdelrahman\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:

Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`
ing other arguments without an explicit keyword will result in an error or misinterpretation.
```

Out[46]:
```
<matplotlib.legend.Legend at 0x1f46557acd0>
```



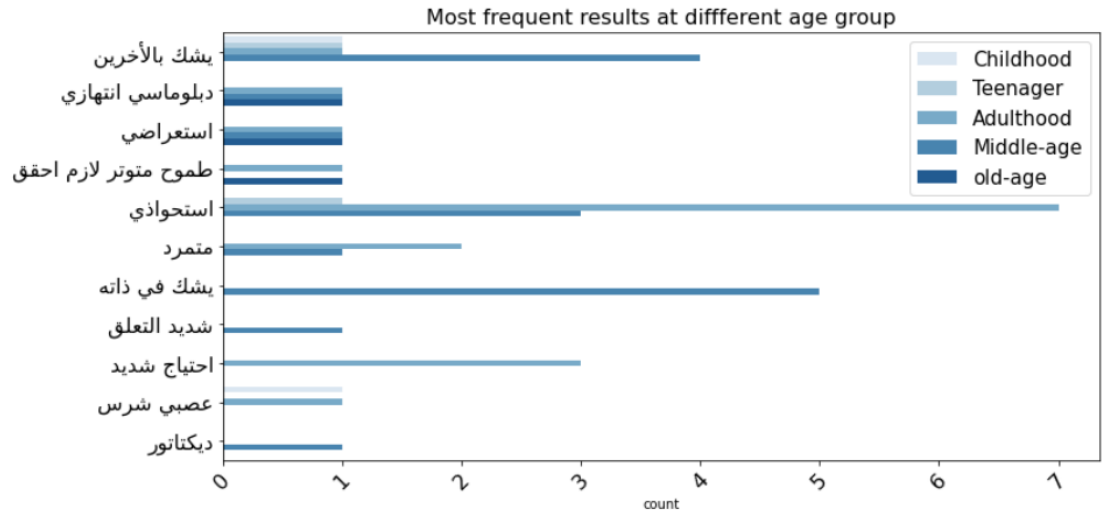
In [47]:
```
plt.figure(figsize=(12,6))
g = sns.countplot(y= x, hue = data["Gender"], palette='Blues')
g = g.set_title("Most frequent results for Male and Female users", fontsize=16)
plt.xticks(fontsize=16,rotation = 45)
plt.yticks(fontsize=16)
plt.legend(fontsize=16)
```

Out[47]:
```
<matplotlib.legend.Legend at 0x1f466033a90>
```



In [48]:
```
plt.figure(figsize=(12,6))
g = sns.countplot( y = x, hue = data["age_avg"],
                   hue_order = ['Childhood', 'Teenager', 'Adulthood','Middle-age','old-age'], palette='Blues')
g = g.set_title("Most frequent results at diffferent age group", fontsize=16)
plt.xticks(fontsize=16,rotation = 45)
plt.yticks(fontsize=16)
plt.legend(fontsize=15)
```

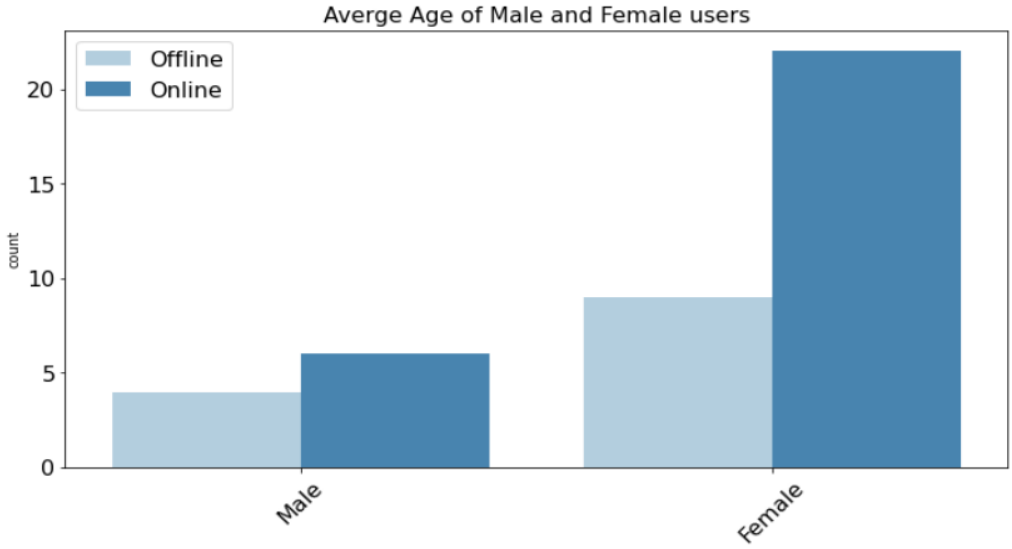
Out[48]:
```
<matplotlib.legend.Legend at 0x1f46612a1f0>
```



In [37]:
```
plt.figure(figsize=(12,6))
g = sns.countplot(data["Gender"], hue = data["Type of Metting"], palette='Blues')
g = g.set_title("the preferred type of meeting for Male and Female users", fontsize=16)
plt.xticks(fontsize=16,rotation = 45)
plt.yticks(fontsize=16)
plt.legend(fontsize=16)
```

```
C:\Users\Abdelrahman\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:

Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument
ing other arguments without an explicit keyword will result in an error or misinterpretation.
```

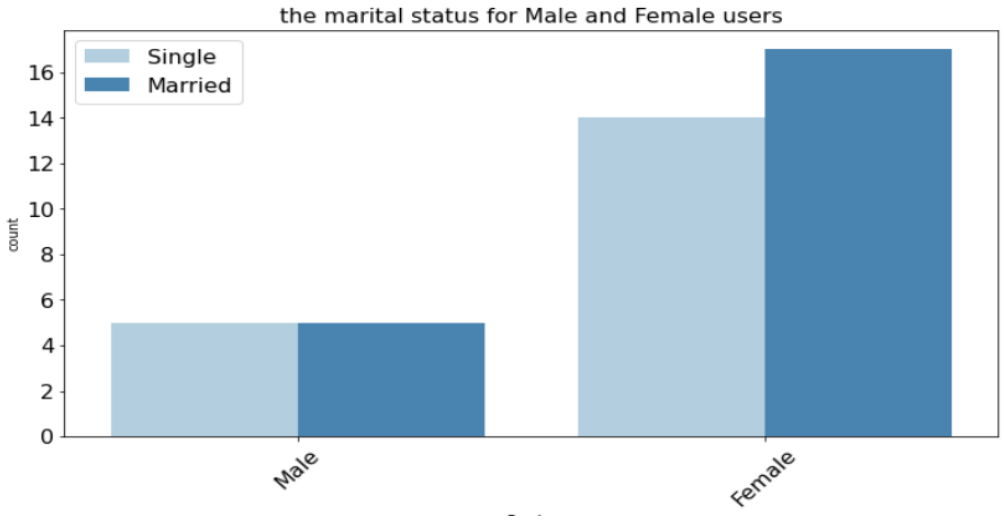
Out[37]:
```
<matplotlib.legend.Legend at 0x1f4639b7340>
```



In [49]:
```
plt.figure(figsize=(12,6))
g = sns.countplot(data["Gender"], hue = data["Status"], palette='Blues')
g = g.set_title("the marital status for Male and Female users", fontsize=16)
plt.xticks(fontsize=16,rotation = 45)
plt.yticks(fontsize=16)
plt.legend(fontsize=16)
```

```
C:\Users\Abdelrahman\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:

Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument
ing other arguments without an explicit keyword will result in an error or misinterpretation.
```

Out[49]:
```
<matplotlib.legend.Legend at 0x1f466167df0>
```



5.4 Mobile code:

5.1 Interceptor:

```
package com.mostafa.gradproject2.api

import ...



// this code it for pass the access token every request
class MyInterceptor(var accessToken: String) : Interceptor {
    override fun intercept(chain: Interceptor.Chain): Response {
        val request = chain.request() Request
            .newBuilder() Request.Builder
            .addHeader( name: "Authorization",  value: "Bearer $accessToken")
            .build()
        return chain.proceed(request)
    }
}
```

5.2 Home:

```
//get data from api
private fun getAppoinmentdata() {
    if (selectedDate.isNotEmpty()) {
        RetrofitInstance.apiInterface.getAppointments()
            .enqueue(object : Callback<AppointmentResponse?> {
                override fun onResponse(
                    call: Call<AppointmentResponse?>,
                    response: Response<AppointmentResponse?>
                ) {
                    if (response.isSuccessful && response.body() != null) {
                        val appointmentResponse = response.body()!!
                        appointmentResponseList = response.body()!!.data

                        // Convert time format
                        appointmentResponseList.forEach { appointment ->
                            appointment.from = convertTimeFormat(appointment.from)
                            appointment.to = convertTimeFormat(appointment.to)
                        }

                        // Filter the data based on selectedDate
                        val filteredList = appointmentResponseList.filter { appointment ->
                            val dateFormat = SimpleDateFormat( pattern: "yyyy-MM-dd", Locale.getDefault())
                            val appointmentDate =
                                dateFormat.format(dateFormat.parse(appointment.date))
                            val selectedDateFormat =
                                dateFormat.format(dateFormat.parse(selectedDate))
                            appointmentDate == selectedDateFormat  ^filter
                        }
```

```
    }

    // Update the number of appointments
    numberofAppointment = filteredList.size
    binding.numberOfAppointmentTextView.text =
        numberofAppointment.toString()

    // Set the filtered list in the adapter
    val appointmentAdapter = AppointmentAdapter(filteredList, clickListener: this@HomeFragment)
    AppointmentRecycleView.adapter = appointmentAdapter
    AppointmentRecycleView.layoutManager = LinearLayoutManager(context)

    // Apply layout animation
    val controller = loadLayoutAnimation(context, R.anim.layout_animation)
    AppointmentRecycleView.layoutAnimation = controller

    // Start layout animation
    AppointmentRecycleView.scheduleLayoutAnimation()


    // This is a success response

} else {
    val errorBody = response.errorBody()?.string()
    // Handle the error case when the response is not successful
    Toast.makeText(
        context,
        text: "Request failed: $errorBody",
        Toast.LENGTH_SHORT
```

```
//on appointment Click
//get PatientDetails form api
override fun onAppointmentClicked(appointment: Data) {
    val appointmentId = appointment.id

    RetrofitInstance.apiInterface.getPatientDetails(appointmentId.toString())
        .enqueue(object : Callback<PatientDetails> {
            override fun onResponse(
                call: Call<PatientDetails>,
                response: Response<PatientDetails>
            ) {
                if (response.isSuccessful && response.body() != null) {
                    val patientDetails = response.body()!!
                    val patientAge = patientDetails.data.age
                    val currentCity = patientDetails.data.current_city
                    val email = patientDetails.data.email
                    val gender = patientDetails.data.gender
                    val name = patientDetails.data.name
                    val originalCity = patientDetails.data.original_city
                    val patientImage = patientDetails.data.patient_img
                    val patientId = patientDetails.data.patient_id
                    val phoneNumber = patientDetails.data.phone_num
                    val status = patientDetails.data.status
                    val app_id = patientDetails.data.app_id.toString()


                    // Create a bundle with patient details
                    val bundle = bundleOf( ...pairs:
                        "patientAge" to patientAge,
```

5.3 Retrofit Requests:

```
package com.mostafa.gradproject2.api

import ...

interface ApiService {


    @POST("auth/login")
    fun login (@Body request: LoginRequest): Call<LoginResponse>

    @GET("doctor_appointments")
    fun getAppointments():Call<AppointmentResponse>

    @GET("appointment/{id}")
    fun getPatientDetails(@Path("id") appointmentId: String): Call<PatientDetails>

    @POST("appointment/note")
    fun addnote(@Body request: AddNoteRequest):Call<AddNoteResponse>



}
```

```
    if (currentDateTimeParsed != null && appointmentDateTime != null && currentDateTimeParsed.before(
            appointmentDateTime
        )
    ) {
        // Current date and time is after the appointment's date and time
        // Do something here, e.g., change the text color or background
        itemView.findViewById<TextView>(R.id.meeting_state_txt).text = "In process"
        // Change text color
        val textColor = ContextCompat.getColor(itemView.context, R.color.black)
        itemView.findViewById<TextView>(R.id.patientName_TextView).setTextColor(textColor)
        itemView.findViewById<TextView>(R.id.to_TextView).setTextColor(textColor)
        itemView.findViewById<TextView>(R.id.from_TextView).setTextColor(textColor)
        itemView.findViewById<TextView>(R.id.type_Textview).setTextColor(textColor)
        itemView.findViewById<TextView>(R.id.meeting_state_txt).setTextColor(textColor)
        itemView.findViewById<TextView>(R.id.dash_TextView).setTextColor(textColor)

        // Change background color
        val backgroundColor = ContextCompat.getColor(itemView.context, R.color.yellow)
        itemView.findViewById<ShapeableImageView>(R.id.redDot_imageView)
            .setBackgroundColor(backgroundColor)
        itemView.findViewById<ConstraintLayout>(R.id.card_layout).backgroundTintList =
            ContextCompat.getColorStateList(itemView.context, R.color.white)
    }

    // Load the image URL using Glide
    Glide.with(itemView.context) RequestManager
        .load( string: "https://backend.inflow2023.online/${appointment.img}") RequestBuilder<Drawable!>
        .into(patientImage)
```

Chapter Six:
Testing

6.1 Testing Plan:

The Test Plan outlines the scope, approach, resources, and schedule of all testing activities. It identifies the items and features to be tested, and types of testing. It contains a detailed and executable strategy for conducting. It defines the detailed testing objective specific to a particular system, the testing approach, test environment, test conditions, and the test plan.

The Test Plan has been created to communicate the test approach to team members. It includes the objectives, scope, schedule, risks, and test approach. This document will clearly identify what the test deliverables will be and what is deemed in and out of scope.

6.2 Test Objective:

The general test objectives are to test the correctness of the generation of the interface data file, the content of the interface data file, and any error conditions. The quality objectives of testing the website are to ensure complete validation of the business and software requirements:

- To evaluate the work products such as requirements, design, user stories, and code.
- To verify the fulfillment of all specified requirements.
- To validate if the test object is complete and works as per the expectation of the users and the stakeholders.
- To build confidence in the quality level of the test object.
- To prevent defects in the software product.
- To find defects in the software product.
- To provide sufficient information to stakeholders to allow them to make informed decisions, especially regarding the level of quality of the test object.
- To reduce the level of risk of insufficient software quality.
- To comply with contractual, legal, or regulatory requirements or standards, and to verify the test object’s compliance with such requirements or standards.

6.3 Testing Approach:

Criteria	Black Box Testing	White Box Testing
Definition	**Black Box Testing is a software testing method in which the internal structure/ design/implementation of the item being tested is NOT known to the tester**	**White Box Testing is a software testing method in which the internal structure/ design/implementation of the item being tested is known to the tester.**
Levels Applicable To	**Mainly applicable to higher levels of testing**	**Mainly applicable to lower levels of testing**
Responsibility	**Generally, independent Software Testers**	**Generally, Software Developers**
Programming Knowledge	**Not Required**	**Required**
Implementation Knowledge	**Not Required**	**Required**
Basis for Test Cases	**Requirement Specifications**	**Detail Design**

3.1 Requirement Analysis:

Requirements analysis is critical to the success or failure of a systems or software project, so we must verify, validate, and confirm each requirement. Requirements must be validated based on User Experience, User Interface, how to test the requirements, Requirements are up to date.
Make sure the major scenarios and requirements are mentioned in the document. If there is something missing, highlight the missing requirements and suggest improvements if there are any.

3.2 Design Testing:

Test all the designs and verify all the designs must be correct as per the requirements. And make sure the designs for all the specified languages and dark themes.

3.3 Functionality Testing:

Test the functional requirements and determine every function of the software is acting in accordance with the pre-determined requirements and tasks. At website, performed testing of all the links in web pages, checking the database connections, forms used in the web pages for submitting or getting information from user & Cookie testing. Functional testing is extended to the types given below.

3.3.1 Testing all the Links:

1. Test the outgoing links from all the pages from a specific domain under test.
2. Test all internal links.
3. Test links jumping on the same pages.
4. Test links used to send email to admin or other users from web pages.
5. Test to check if there are any orphan pages.
6. Lastly in link checking, check for broken links in all above-mentioned links.

3.3.2 Testing of the forms on the web pages:

Forms are the essential and integral parts of a website.
Forms are used to get information.
from users and to keep interaction with them.
The following should be checked on the forms:

- Check all the validations in each field.
- Check for the default values of fields.
- Wrong inputs to the fields in the forms.
- Options to create forms, if any, form delete, view, or modify the forms.
- Check that no empty forms are created.
- There are different field validations like email-id's, user financial information, date, etc.

All the above validations should be checked in a manual or an automated way.

3.4 Validation (HTML/CSS):

- HTML/CSS validation is very important for optimizing the website for
- search engines.
- The site has full and correct Doctype.
- The site uses a character set.
- The site uses valid XHTML.
- The site uses valid CSS.
- The site has no unnecessary ids or classes.
- The site uses well-structured code.
- The site has no broken links.
- The site has clearly defined visited links.

3.5 Database Testing:

- Check for data integrity and errors while you edit, delete, modify the forms, or do.
- any database related functionality.
- Check if all the database queries are executing correctly and data is retrieved.
- correctly and updated correctly.
- Also, we need to validate the Database by executing the queries.

3.6 API Testing:

Set of procedures to verify the expected functionality, reliability, and security and ensure.
the correct interaction between backend and frontend. To validate the logic of the build
architecture within a short amount of time. Each API test consists of some test actions.
mentioned below. Further details of API Testing will be covered in API Test Plan

- Verify the URL accordingly environment.
- Verify required request headers and their correct values.
- Verify response payload.
- Verify correct HTTP status code and response headers.
- Verify expected result and correct application state.
- Verify correct performance sanity.

3.7 Usability Testing:

The goals of usability testing include establishing a baseline of user performance,
establishing and validating user performance measures and identifying potential design.
concerns to be addressed to improve efficiency, productivity, and end-user.
satisfaction.
-The usability test objectives are:

- To determine design inconsistencies and usability problem areas within the user interface and content areas.
- Potential sources of error may include:
- Navigation errors – failure to locate functions, excessive keystrokes to complete a function, failure to follow recommended screen flow.
- Presentation errors – failure to locate and properly act upon desired.
  information in screens, selection errors due to labeling ambiguities.
- Control usage problems – improper toolbar or entry field usage.
- Exercise the application or web site under controlled test conditions with
- representative users. Data will be used to access usability goals regarding.
- an effective, efficient, and well-received user interface has been achieved.
- Establish baseline user performance and user-satisfaction levels of the use interface for future usability evaluations.

-Basic Usability:

- The site should have a clear hierarchy.
- Headings clearly indicate the structure of the document.
- Navigation should be easy to understand.
- Navigation is consistent throughout the site.
- The site uses underlined links.
- The site uses consistent and appropriate language.
- The site has an easy to find sitemap and contact page.
- The site has a search tool.
- The site has a link to the home page on every page.

- The site has clearly defined visited links.

3.8 Compatibility Testing:

3.8.1 Browser compatibility:

- Some requirements are very dependent on browsers. Different browsers have different.
- configurations and settings that the web page should be compatible with. The web site
- coding should be cross browser platform compatible. Test the UI of website, functionality,
- security checks or validations then stresses browser compatibility testing of the web.
- application. Test web application on different browsers like Internet explorer, Chrome,
- Firefox, Netscape navigator, AOL, Safari, Opera browsers with different versions

3.8.2 OS compatibility:

- Some functionality in the web application may not be compatible with all operating.
- systems. All new technologies used in web development like graphics designs, interface.
- calls like different API's may not be available in all Operating Systems. Testing the web
- application on different operating systems like Windows, Unix, MAC, Linux, Solaris with
- different OS flavors.

3.8.3 Mobile browsing:

- Mobile browsing is also an integral part of browsing. Testing the web pages on mobile
- browsers are highly important. Compatibility issues may be there on mobile. Currently, the system is not designed for mobile browsing, although this is an area we can add in future.

3.9 Performance testing:

- Website should sustain heavy load. website performance testing should include Load.
- Testing & Web Stress Testing. Test the website on different Internet connection speeds.

3.9.1 Load Testing:

- Test website if many users are accessing or requesting the same page. Can the system
- sustain peak load times? Site should handle many simultaneous user requests, large input
- data from users, Simultaneous connection to DB, heavy load on specific pages etc. For load
- testing we will use blaze meter and robot scripts.

3.9.2 Stress Testing:

- Generally, stress means stretching the system beyond its specification limits. Web stress
- testing is performed to break the site by giving stress and check how the system reacts to
- stress and how the system recovers from crashes. Stress is generally given on input fields,
- login and sign-up areas. In website performance testing website functionality on different
- operating systems, different hardware platforms are checked for software, hardware.
- memory leakage errors. For stress testing we will use JMeter and blaze meter.

3.10 Smoke Testing:

Smoke Testing is a software testing process that determines whether the deployed software build is stable or not. Inside the smoke testing QA Engineer will make sure all the critical functionalities are working fine. We will create a checklist for smoke testing, Smoke testing will be performed at two stages. Once new features are added, the other is before finalizing the build for Production/live. Create a checklist for smoke testing.

3.11 Beta Testing:

Beta testing is basically a release for specific users to use a product in a production environment to uncover any bugs or issues before a general release. Beta testing is the final round of testing before releasing a product to a wide audience. The objective is to uncover as many bugs or usability issues as possible in this controlled setting. QA will also perform the beta Testing.

3.12 Test Strategy:

The overall strategy of this testing initiative is manual, black box testing. We are testing the data, interface part and implemented system in detail. The testing at the SAP end of the interface will be covered by the SAP functional testing. Follow the testing phases and techniques mentioned inside “Detailed Test Approach”. All type of testing are covered in this document. Some of the test specifications use test data which needs to be set-up in the test environment prior to executing the test cases. For each level of testing,

a separate test plan is prepared with the following set of deliverables:

- Test Cases/Test Scenarios
- Features to be tested.
- Items to be tested.
- Pass / Fail criteria.
- Bugs cycle.
- Automation.
- Expected Results.
- Actual Results.

3.13 Test Schedule:

The test schedule is the timeline of acceptance testing activities and deliverable dates, Testing activities are mentioned below.

- Requirement Analysis.
- Design Testing
- Develop test scenarios.
- Develop test cases.
- Review scenarios/test cases for accuracy, completeness, and sequence (confirm test data is correct).
- Integration testing.
- API Testing.
- Regression Testing.
- Functionality Testing.
- Database Testing.
- Integration Test Specification.
- Usability Testing.
- Compatibility Testing.
- Performance Testing.
- UAT Testing.
- Automation Testing.
- Smoke Testing.
- Beta Testing.

3.14 Problem/Bug Severity Classification:

The identified severity for each problem implies a general reward for resolving it, and a general risk for not addressing it, in the current release.

- Severity 1 - Crash or High impact problems that often prevent a user/host from correctly completing an experience/booking.
- Severity 2 - Moderate to high frequency problems with the functionality/UI or UX impact
- Severity 3 - Either moderate problems with low frequency or low problems with moderate frequency; these are minor annoyance problems faced by a number of participants.
- Severity 4 - Low impact problems faced by few participants; there is low risk of not resolving these problems. Reward for resolution is typically exhibited in increased user satisfaction.

3.15 Environment:

Start testing on a staging server once a certain level is achieved, then move to Production and give the final approval at Production. All the experiments should be performed at staging. Testing data must be private at Production.

3.16 Tools and defect Tracking:

Jira will be used for defect reporting and issue bugs/defects management and traceability.

3.17 Exit Criteria:

All the test cases and test scenarios must be covered in the testing process.

Chapter Seven:
Conclusion and Future Work

7.1 Summary of achieved result:

We made a website to make it easy for people to ask for help to professional life coaches which can really help user in their life choices and problems, Our website is also useful for couples to manage their problems and understand each other's perspective,, Our website can be used by companies to test employees in interviews, Our website offers articles for user to increase their awareness about his mental health., We also created a mobile application for coach to manage the whole process automatically.

7.2 further development

- We're planning to add a payment feature to make the whole process automated.
- We're planning to use machine learning to recommend books to read for the user based on his results from the test (if possible).
- We're planning to add more tests that can be used in increasing user insights.
- We're planning to make a mobile application for the whole system.

الفكرة العامة

النظام النفسي للتحليل والتشخيص هو موقع يوفر تواصل سريع بين العميل واخصائي العلاقات النفسية، يوفر هذا التواصل عن طريق شبكة الانترنت او بالتواصل وجها لوجه، التواصل عن بعد يكون عن طريق حجز موعد، واجراء اختبار ليعرفوا اكثر عن انفسهم نظام تحليل نفسي حسب طلب العميل هو نظام يعطي العميل القدرة على فحص الصحة العاطفية اعتمادا على نظام تيست لوشر للألوان، و النظام الاخر هو نظام على الاجهزة المحمولة. يتيح للأخصائي ان يدير النظام وبيانات العميل ويتتبع مواعيده ويمكنه من اخذ الملاحظات خلال الجلسة. والحمد لله رب العالمين.

References

- https://www.youtube.com/playlist?list=PL1FWK-sgJ9elLBrEyAXK74pM37lJtDzut
- https://www.youtube.com/@programmingwithmosh
- https://www.youtube.com/@ElzeroWebSchool
- https://www.guru99.com/software-testing-introduction-importance.html
- https://www.youtube.com/watch?v=jy8T-QQ7MiQ
- https://www.youtube.com/playlist?list=PL1FWK-sgJ9elLBrEyAXK74pM37lJtDzut
- https://www.youtube.com/watch?v=F9UC9DY-vIU
- https://www.w3schools.com/html/default.asp
- https://developer.android.com/studio
- https://www.youtube.com/watch?v=K7RfBgoeg48