FixIT

Documenting the existing application from MDS

1. Project Requirements

The list of User Stories provided:

- "I wish for an application specifically designed for the technical maintenance of a personal residence."
- "On the main page, I like that I can get a clearer idea of all the available service categories. There, I also have the option of specific search through the Search Bar feature."
- "I would like there to be a tips page that would help me solve minor problems individually."
- "When creating an account, I can select the desired user type: 'Service Recipient' or 'Service Provider.'"
- "I would like to be able to configure my profile."
- "I would like to be able to add friends and then see their list."
- "As a service provider, I like that I can post and edit detailed announcements about my services."
- "When searching for an announcement, I like that I can filter services based on numerous criteria."
- "If I access a post, I appreciate that I can see both photos and information about the offer, as well as user reviews for that specific service."
- "I would like to see the tips recommended by a craftsman on their profile."
- "As a craftsman, I would like to be able to post videos that I consider helpful for customers."

For the initial project, every single User Story was completed.

2. Team Description

The list of the team members that collaborated on FixIT:

- Bojneagu David
- Georgescu Elena
- Gheorghe Cosmina
- Neagu Mara
- Talpiga Vlad

The responsibilities for each member of the team:

- Bojneagu David
  - Database entities & functionalities, Profile page backend, Home page, post filtering on the Home page, CRUD posts, ReadME;

- Georgescu Elena
  - Database entities & functionalities, Profile page backend, Home page, post filtering on the Home page, managing the followers of an account;

- Gheorghe Cosmina
  - Database entities & functionalities, Profile page backend, Home page, post deletion, CRUD reviews, managing the different roles an account can take (Client & Master), managing the followers of an account, creating Entity Relationship, Use Case Diagrams;

- Neagu Mara
  - Database entities & functionalities, Profile page backend, Home page, post editing & view the detailed post, CRUD reviews, CRUD comments, creating WorkFlow Diagram;

- Talpiga Vlad
  - Database entities & functionalities, Home page, CRUD tips, Tips page.

3. **Software Architecture Report**

a) The technology we have used is **MERN Stack**:

Benefits of MERN:

- MongoDB (Database): Chosen for its flexibility and scalability, allowing easy adaptation to changing data structures.
- Express.js (Back-End Framework): Selected for its simplicity and robust features, facilitating the development of a scalable and maintainable back-end.
- React (Front-End Library): Enables the creation of dynamic user interfaces, enhancing the overall user experience.
- Node.js (Server Environment): Offers a non-blocking, event-driven architecture, ensuring optimal performance and scalability.

b) Architectural Patterns Implemented:

- **Model-View-Controller** (MVC):
  - Implementation: Adopted the MVC architectural pattern to separate concerns between the model (data), view (user interface), and controller (business logic).
  - Appropriateness: The MVC pattern has proven highly appropriate for managing the complexity of the application. It enhances maintainability, as

changes in one component do not necessitate modifications in others. This separation of concerns has facilitated efficient development and easier debugging.

- **Microservices**:
  - Implementation: Utilized a microservices architecture to break down the application into smaller, independent services.
  - Appropriateness: The microservices pattern is appropriate for a platform focused on communication between clients and service providers. It enables scalability, fault isolation, and independent deployment of services. This approach allows the application to evolve and scale more flexibly, especially in scenarios where specific functionalities might experience varying loads.

c) Coding Principles & Standards Established:

- **Testing Standards**:
  - Running automated tests as part of the development process and addressing any test failures promptly. Adhering to testing standards, such as writing meaningful test cases and maintaining test coverage.

d) There are **no** faults that were discovered during development that haven't been addressed by the time of delivery.

e) **No** part of the project requires refactoring.

This is the link to the MDS project repository: https://github.com/maraneagu/FixIT-MERN