**Software Engineering Project – Hackelite**

**Project Title: Fuel Xpress (Online Fuel Delivery)**

**Group Name**: Hackelite

**Group Members:**

* Uday Vandanapu
* Sahithi Regalla
* Kavya Jangapalli
* Durga Devi Pampani
* Pranay Pendyala
* Hemanth Janapala
* Karthik Setti
* Mounika Eddala
* Vaishnavi Adapa

**Project Description**:

We are creating a website for online fuel delivery. The website features three user interfaces: admin, user, and petrol station. The following functionalities are available in the various interfaces:

Admin Interface: For managing all gas stations and monitoring orders, as well as setting gasoline prices.

Gas Station Interface: View and process pending orders, add vehicles, change inventory, cancel orders if they are out of gasoline, and recruit skilled drivers to deliver orders.

Driver Interface: Accepting the order, adding it to the queue, and traveling to the user's location using Google Maps. Drivers can view their working hours, ongoing and forthcoming deliveries.

User Interface: The user interface provides sign-up and login functionality. After signing up and logging in with the same credentials, a person can access his account. The user has access to the full website and can order fuel from any place. Users can specify the fuel quantity, type, time, and location for fuel delivery and have the option to cancel the order. They can also track the delivery person's location and projected arrival time using the integrated Google Maps. Users can see their upcoming, continuing, completed, and canceled orders.

The website includes a search function that saves the user time. The page includes information of several sorts of fuels. After selecting the location, gasoline quantity, and fuel type, the user can proceed with the order. He can either utilize a current GPS location or enter a new address here before being directed to the payment gateway, where he must make the payment. When the payment is completed, the user is returned to the site, where he can view the order that was placed in his account. Under certain conditions, the user may cancel the order. The quantity of the product is changed depending on the quantity of the order on the website whenever the order is placed or cancelled.

The delivery process begins once the order is received by the local gasoline supplier. The truck moves to the spot where fueling is required, with handy tracking supplied by built-in maps. When the driver gets at the location, he or she fills the tank while following all safety standards and procedures to ensure safe fuel delivery. Once delivery is complete, the order status can be changed to finished. If the user cancels the order, the sum will be reimbursed to the user.

* **Language and technology:** We aim to build the website using Django as the backend, angular as the frontend, and MySQL as the database. VS Code, and MySQL Workbench are our preferred IDEs for Django, angular, and MySQL. Our code repository will be on GitHub.

**Project Timeline:**

* **Milestones:**

**Milestone1:**

**Planning phase:** Gathering the requirements essential for developing the website front end, backend end and database software.

**Milestone 2:**

**Designing phase:** Designing the front-end and interfaces for the 4 entities Admin, user, gas station, driver with all required features and options.

**Milestone 3:**

**Implementation phase:** Developing background logic by connecting multiple interfaces and databases.

**Milestone 4:**

**Testing Phase:** Testing the functionality of the entire system to ensure the entire application is working as expected. If any bugs find will develop and again test the accordingly missing features should be updated.

**Milestone 5:**

**Deployment: O**nce testing is done and everything working fine will prepare proper documentation.

* **Timeline:**

A screenshot of a project schedule

Description automatically generated

* **Chart:**

**Risk Management:**

Risk is an unwanted event that has negative consequences

**Risk exposure = (risk probability) \*(risk impact)**

Following are the risks we encountered:

1. **Password Encryption:** The payload tab in the browser will display the data sent from the user registration form. The password will also be displayed when it is sent to the backend for verification. The attackers now have an opportunity to access the user account and take the data. To prevent this, the password is encrypted on the client side before being sent to the server in a request, which is then encrypted once more before being saved in the database.
2. **Authentication:** Vulnerabilities in authentication can have a devastating effect. An attacker has access to all the information and features of the compromised account once they have either bypassed authentication or have brute-forced their way into another user's account. A token is supplied to the user when his or her credentials have been verified. For any ensuing queries to confirm the authenticity of the user, this token is passed from the user side. Requests to the server are refused with an error code if the token is invalid. This token has a secret code that is checked at the servers' end to see if it is invalid.
3. **Cross Site Scripting:** By supplying untrusted user data, the attacker tries to run malicious code in a web application. Assuming that the request is valid, the server fulfills it and transfers the data to the hackers' server. The hackers can obtain vital User data because of this vulnerability. In our app, this is avoided by verifying each request made to the server. If the data is not what was expected, the server will respond with incorrect request code

**Contingency Plan to Mitigate Risks:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.NO | Risks | Risk probability | Impact | Solution |
| 1 | Password Encryption | High | High | To prevent this, the password is encrypted on the client side before being sent to the server in a request, which is then encrypted once more before being saved in the database |
| 2 | Authentication | Medium | High | Web token has a secret code that is checked at the servers' end to see if it is invalid. |
| 3 | Cross Site Scripting | Low | Medium | This is avoided by verifying each request made to the server |

**Team Roles:**

|  |  |
| --- | --- |
| Backend | Karthik Setti, Pranay Pendyala |
| Frontend | Kavya Jangapalli, Vaishanavi Adapa |
| MySQL | Durga Devi Pampani |
| Testing | Sahithi Regalla, Mounika Eddala |
| Design | Uday Vandanapu, Hemanth Janapala |
| Documentation and Presentation | All the team members take part. |

**Member Contribution Table:**

|  |  |
| --- | --- |
| **Team Members** | **Responsibilities** |
| Karthik Setti | Risk Management, Video Presentation |
| Durga Devi Pampani | Project Description |
| Uday Vandanapu | Directory Structure |
| Vaishnavi Adapa | Project Timeline, Video Presentation |
| Sahithi Regalla | Meeting Minutes, Video Presentation |
| Kavya Jangapalli | Directory Structure |
| Mounika Eddala | PowerPoint, Video Presentation |
| Pranay Pendyala | Risk Management |
| Hemanth Janapala | Project Description |