Shelter Site System Architecture and Planning

**Team name: Sheltered Ones**

**Team members and roles:**

Team Leader: Conor Babcock

Technical Writer and Full-stack Developer: Gary Hui

Front-end Web Developer: Roman Yacik

Database Developer: John Wang

**Team Github repo:**<https://github.com/ghui64/Sheltered-Ones>

**1.**    **System architecture (75 pts)**

**1.1 Overview**

The architecture of Shelter Website is based on a client-server 3-tier architecture (Fig. 1), which consists of client application, web server, and database.

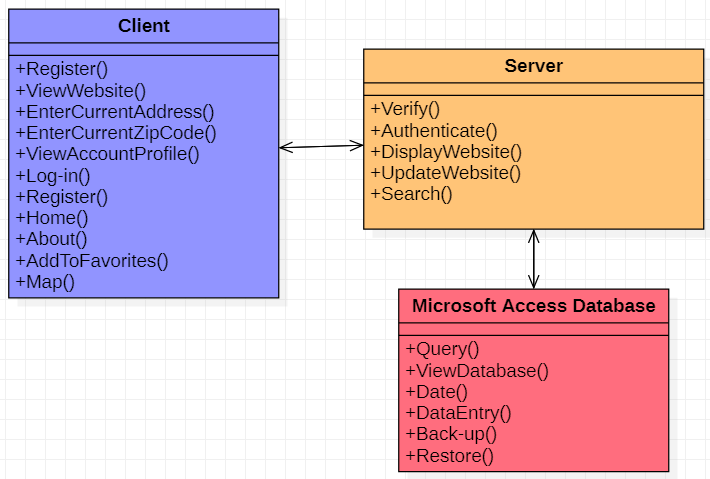


Fig. 1: Shelter Website System Architecture

The main reasons for considering three-tier architecture for the Shelter Website are as follows:

**Centralized:**

* Resources and data security are concentrated in one location and controlled through the server
* System configures accounts rather than configuring security and resource access for each computer on the network

**Flexibility:**

* Management of data is independent from the physical storage support.
* New technology may be easily integrated into the system
* Maintenance of the business logic is easier.
  + Database is currently designed to use SQL. But can easily transfer over to Azure and other cloud servers.
  + Website and database are separated for maintenance.

**Accessibility:**

* Server can be accessed through various platforms yet still process data
* Clients may simultaneously access resources on the file server
* Remote access to the server through a network connection is more efficient than a physical connection
* Client and server communicate with a series of get requests and replies with responses

**Reusability:**

* The software architecture of this project is based on the client-server template, which can be used in various applications that require a controlled distribution of information.
* This project is currently targeting Boston local shelters. But the client-server template and setup can be easily applied to different cities.

**Security:**

* Server data is secured, because normal members and visitors cannot edit the data (View only)
* Shelter staff are required to verify themselves before granting special privileges from IT staff to update changes (Only able to update changes on vacant beds and resources, and only able to change their shetler’s information).
* IT staff accounts are the only one that are able to view and edit information for both the website and database.

**1.2 Details**

**Client:**

Client provides the user interface to interact with the system. It communicates with web servers to process information based on user actions. Client module has the following components:

* Login: a login page that allows user to create a new account or sign in
* Catalog: a page that lists all homeless shelters within our database and allows them to search for nearby shelters
* Map: Google maps services appears when searching for nearby shelters
* Account Profile: a page that displays private account information
* Favorites: a page that displays a list of the account’s favorited shelters
* Home: a page that welcomes the users and
* About: a page that describes who the Sheltered Ones team are, our purpose,  email contact, and website description
* View the website: sends a request to view the current state of the website

**Server:**

The server processes interactions between the client and website. It also allows clients to handle data manipulation without using SQL commands to directly modify the database. The server module has the following components:

* Display website: sends the client a response to allow visibility and interactiveness of the website
* Update website: refreshes the displayed website information to match the database
* Authorization: checks if the client’s account type matches their privileges
* Verification: checks the input with the database to verify the authenticity of client actions
* Search: search the database for records

**Database:**

Stores the account information used to verify log-in credentials. Stores the information that clients would like to retrieve, such as shelter information. Data stored onto the database may be retrieved and manipulated by the server.

* Back-up: exports data to store the current state of the database into a location in memory
* Restore: imports data to recover a previous state of the database
* Date: get the date when the information was stored onto the database
* Data Entry: insert data via manual entry or import and both can lead to table creation
* Query: fetch data from existing tables
* View database: displays information stored on the database

**1.3 Tradeoffs**

**Database**:

Our project is targeting the Boston local shelters, so our plan for the database and maintenance is using Microsoft Access. It accommodates small organizations with many features, such as: table designs, data, queries, forms, reports, and modules. Using this will allow the servers and IT Staff to directly modify the stored information that they want to display on the website. The drawback to using Access is that it can only handle 200 simultaneous Access users onto the database. This limits the number of servers and IT staff we can provide service to. However, these Access features and data can be migrated to Microsoft Azure SQL for scalability, reliability, and long-term manageability. Initializing our project database with Access will allow our project to meet its deadline and demonstrate its functionality on a local scope before investing efforts into Azure for a larger scale.

**Website:**

For our project, we will be including a front-end website for users to access. The website will display information to all public users for shelter’s information, number of vacant beds and available resources. The website will have multiple components such as catalog with Shelters information, map detail, account profile etc. Users can request catalog and shelter information from the database, and the database will send the data and display on the website.

**Single Database**

Our database relies on Microsoft’s continued service. Even though the database is a single point of failure, Microsoft has a worldwide up-time of 99.98% in Q4 of 2019. Bottlenecking would be addressed by employing a cache system during the transition to Azure. Synchronization is not required for our current local scope, but may be implemented later if the scale of the project expands.

**1.4 Database management system choices (if applicable)**

Microsoft Access was chosen as our database for its functionality, user friendliness, and potential scalability. Our team is familiar with Access and have found that it is possible to communicate it with our website. Data implementation is easier than AWS and does not require billing fees up to our anticipated scale of the project.

If the project receives sufficient feedback and desire from the team to progress, our Access database will be moved to Microsoft Azure SQL. This allows us for an easy migration to a more robust and MORE scalable database. It also grants the system the inherited benefits of a cloud service. However, the query load will increase, so we would need to implement a caching resource. The transition will be performed if the projected number of combined servers and IT Staff accounts exceeds 200.

The individuals managing the database will be the IT Staff. They will be able to import and export the database to satisfy the data back-up and data recovery requirements.

**2.**    **Planning (25 pts)**

-          **Sprint 1:** Feb 18 – Mar 16.

-          **Sprint 1 review** (first demo, retrospective, and planning): Mar 17

-          **Sprint 2:** Mar 17 – April 8.

-          **Sprint 2 review** (final presentation, retrospective, and future work): April 9

TheShelter website product backlog and sample plan are as follows:

**Product Backlog**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Backlog Item** | **Max PW** | **Total PW** |
| **1** | UC-1: Register: to register an account on the website | 4 | 32 |
| **2** | UC-2: Database management: to be able to maintain stability for the database | 3 | 22 |
| **3** | UC-3: Adding new shelter with information: to store and display a shelter’s information that has not yet been added | 4 | 12 |
| **4** | UC-4: View the website: to view information on the website | 2 | 9 |
| **5** | UC-5: View the database: to view information on the database | 3 | 15 |
| **6** | UC-6: Update the shelter information: to modify information on the database | 3 | 11 |
| **7** | UC-7: Request shelter information: to view more information about a shelter | 2 | 12 |
| **8** | UC-8: Enter current address: to input their desired address for searching nearby shelters | 3 | 7 |
| **9** | UC-9: Enter current zip code: to input their desired zip code for searching nearby shelters | 4 | 4 |
| **10** | UC-10: View account profile: to display basic account information and favorite shelters | 4 | 8 |
| **11** | UC-11: Display nearest shelter: to show a map of shelters nearby a given location | 2 | 18 |
| **12** | UC-12: Log-in: to access an account on our website | 4 | 12 |

**Plan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Backlog Item** | **Task** | **Assign To** | **Time Estimate (1 hour unit)** |
| Sprint 1 | UC-2: Database management: to be able to maintain stability for the database | To enable file back-up capabilities of the database into a secure storage location | John W. | 2 hours |
| To enable file restoration capabilities of the database into a secure storage location | John W. | 2 hours |
| UC-4: View the website: to view information on the website | Design and implement website using Wordpress and use of plug-ins | Roman Y.  Conor B. | 10 hours |
| Design and implement various categories that display relevant information within them | Roman Y.  Conor B. | 1 hour |
| UC-5: View the database: to view information on the database | Register and configure Microsoft Access | John W. | 4 hours |
| Connect the database with the website | John W.  Roman Y.  Gary H. | 5 hours |
| UC-6: Update the shelter information: to modify information on the database | Configure database manipulation through input from the website | John W. | 3 hours |
| Push data from database onto the website with a timer | Gary H.  John W. | 3 hours |
| UC-7: Request shelter information: to view more information about a shelter | Design and implement pages about shelters | Roman Y. | 5 hours |
| Implement  redirection from button to webpage | Roman Y. | 1 hours |
| Sprint 2 | UC-1: Register: to register an account on the website | Create an entry for the database that stores account registration history | Gary H.  Roman Y. | 2 hours |
| Design and implement registration page with text fields that take user input | Gary H.  John W. | 4 hours |
| UC-12: Log-in: to access an account on our website | Implement credential verification | Gary H.  Roman Y. | 8 hours |
| Design and implement log-in element and page | Roman Y. | 4 hours |
| UC-3: Adding new shelter with information: to store and display a shelter’s information that has not yet been added | Allow querying from the IT Staff for the database | Gary H.  John W. | 2 hours |
| Establish communication system between the shelter organization and the IT Staff, so that an entry may be made into the database | Roman Y.  John W. | 6 hours |
| UC-10: View account profile: to display basic account information and favorite shelters | To design and implement account profile page | Gary Hui.  Roman Y. | 2 hours |
| Implement favoriting system | Conor B. | 4 hours |
| UC-8: Enter current address: to input their desired address for searching nearby shelters | Implement text field that takes in user input | Conor B. | 2 hours |
| UC-9: Enter current zip code: to input their desired zip code for searching nearby shelters | Implement text field that takes in user input | Conor B. | 2 hours |
| UC-11: Display nearest shelter: to show a map of shelters nearby a given location | Display and implement Google map services for our website | Roman  John W. | 5 hours |
| Implement text field that takes in user input | Conor B. | 2 hours |
| Final presentation | Slides | Team | 24 hours |
| Final Report | Documentation and demo video | Team | 14 hours |