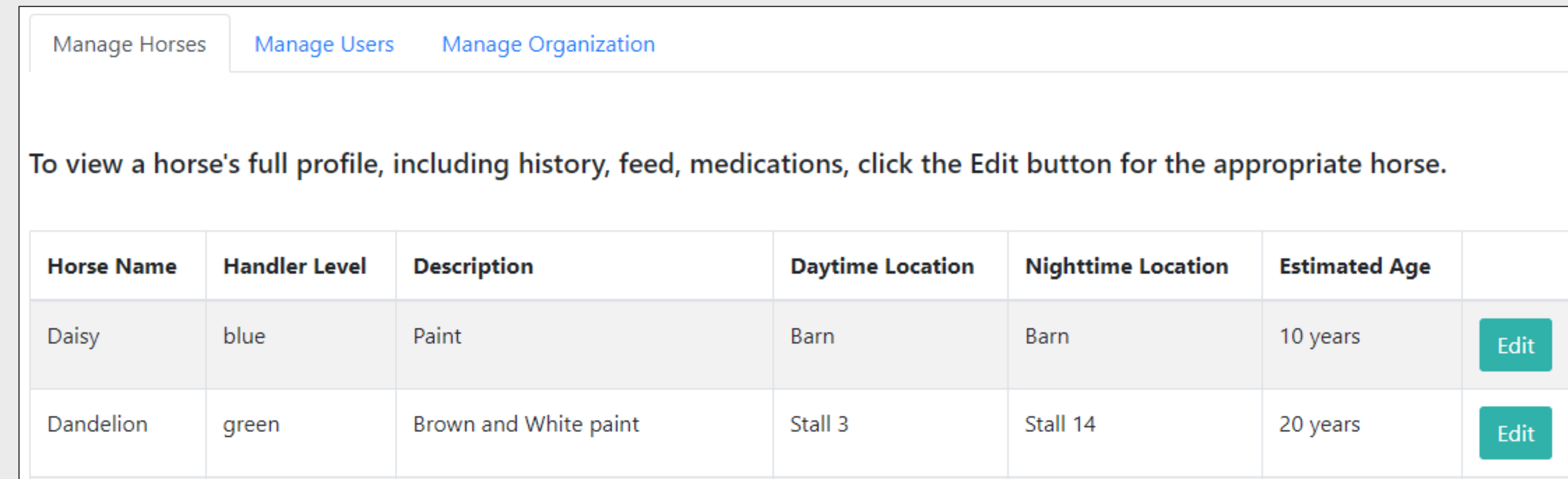


HORSE CAMP MANAGEMENT SYSTEM

- HTML
- CSS
- JavaScript
- SQL
- Bootstrap
- React
- MySQL Workbench
- Heroku

Horse Camp Management System is an application designed to support the care and management of horses. The system includes options to manage each horse's feed, medications, handling requirements, and other history information.

To view a horse's full profile, including history, feed, medications, click the Edit button for the appropriate horse.



Horse Medications

Horse Name	Daytime Location	Nighttime Location	Medication
Dandelion	Stall 3	Stall 14	<ul style="list-style-type: none"> • AM: 1 scoop Probiotics
Easy	Field	Field	

Horse Medications

			Horse Name
Horse Name	Daytime Location	Nighttime Location	Dandelion
Dandelion	Stall 3	Stall 14	Easy
Easy	Field	Field	
Noble	Barn	Field	
			• 1 scoop Glucosamine

In action – Horse Medication Options

IN ACTION – MANAGE HORSES

Manage Horses is part of the Manage Preferences menu, which also includes Manage Users and Manage Organization.

In this menu, a user is able to add a new horse to the system, as well as edit or inactivate a horse that has already been added.

When clicking the Edit button, a modal (pop-up) appears with the fields populated with data from the database for that specific horse.

A user can then edit or add to the fields as needed and save the changes in order to update the database.

IN ACTION – HORSE MEDS

Horse Medications is a view that allows the user to see each horse, the horse's location in the day and night, and the medications they receive at those times, if applicable.

With this view, there are options to view all medications, or medications by morning or night. This allows the handler or the person preparing medications to efficiently gather the supplies needed for the appropriate time of day.

In addition to the Medication view are the Horse and Feed views, which allow handlers to view horses at the camp, as well as the feed each horse should receive.

HCMS: TECHNOLOGIES AND LANGUAGES

- HTML
- CSS
- JavaScript
- SQL
- Bootstrap
- React
- MySQL Workbench
- Heroku

```
render() {
  const {horses} = this.state;

  return (
    <div>
      <header />
      <div className="container">
        <h1 Horse Medications /h1>

        <form>
          <div className="switch-field">
            <input type="radio"
              value="AllMedi"
              id="AllMedi">
              <check/> {this.state.selectedOption === "AllMedi"}
              onChange={this.handleChange} />
              <label htmlFor="AllMedi">All Medi</label>
            <input type="radio"
              value="AM"
              id="AM">
              <check/> {this.state.selectedOption === "AM"}
              onChange={this.handleChange}/>
              <label htmlFor="AM">View AM Meds</label>
            <input type="radio"
              value="PM"
              id="PM">
              <check/> {this.state.selectedOption === "PM"}
              onChange={this.handleChange}>
              <label htmlFor="PM">View PM Meds</label>
          </div>
        </form>
      </div>
    </div>
  );
}
```

Horse Medication – All/AM/PM options

```

import class APIDescriptor extends React.Component {
  constructor(props) {
    super(props);

    this.state = {
      medid: []
    };
  }

  componentDidMount() {
    const horsedID = this.props.horsedID;
    fetch(`${config.api}/med-des/${horsedID}`, {
      method: "GET",
    })
      .then(response => response.json())
      .then(data => this.setState({medid: data}))
      .catch(err => console.log("Error reading data: ", err))
  }

  render() {
    const {medid} = this.state;

    return (
      <ul>
        {medid.map(currentMed =>
          <li key={currentMed.horsedID} {currentMed.amount} {currentMed.unit} {currentMed.medName} {currentMed.medNotes ? ("Notes: " + currentMed.medNotes + " "); } >{currentMed}
        )}
      </ul>
    );
  }
}

```

Horse Medication – AM option

```
<model show={this.state.isOpen} onHide={()>setModal}</hide> size="lg" backdrop="static">
<modal.Header closeButton>
  <modal.Title>
    {>this.props.modelTitle} horse </> ('' + horse.horseName) : ''</></> />
  </modal.Title>
</modal.Header>

<modal.Body>
  <form className="no-border">

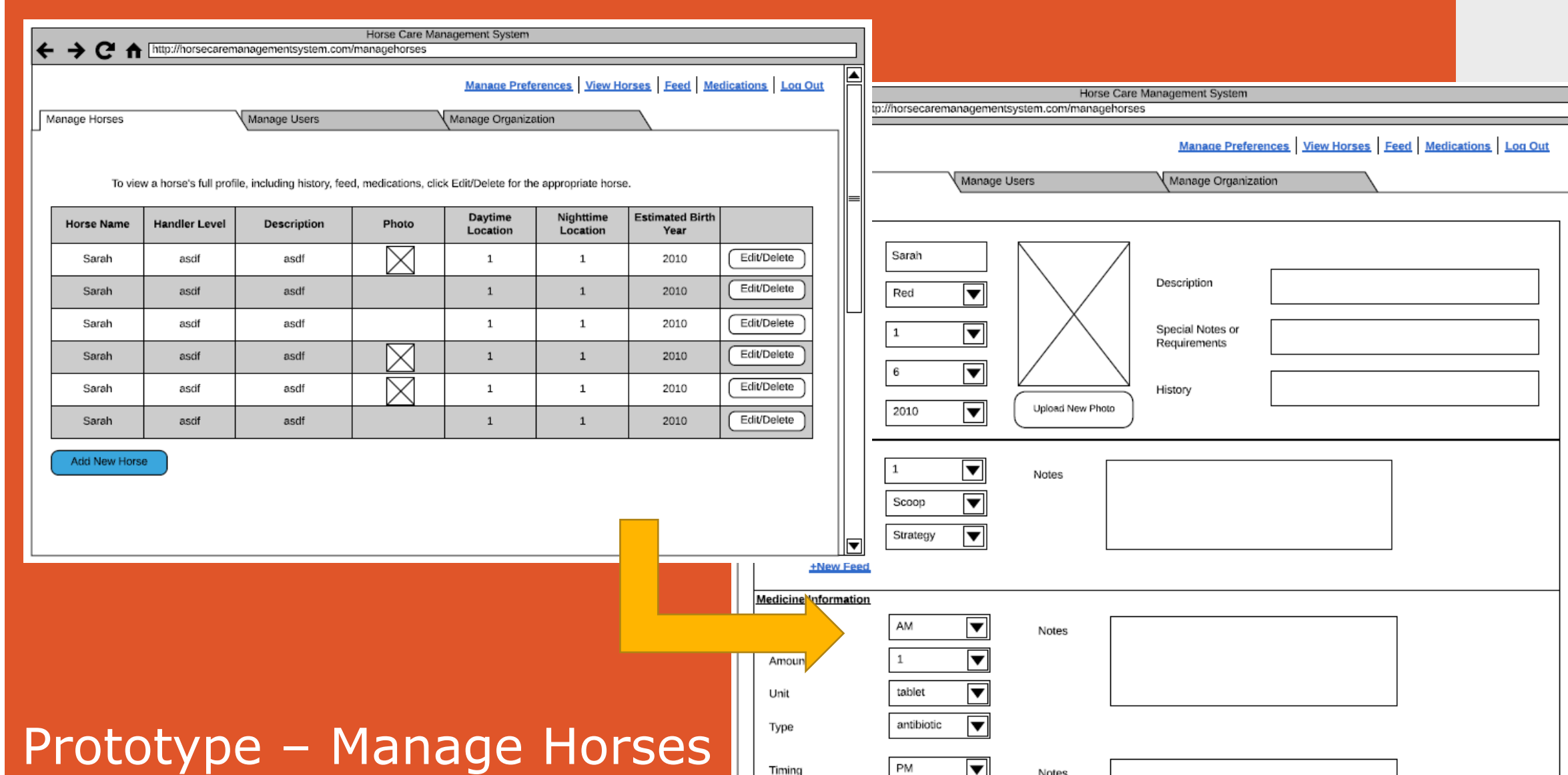
    {''General Horse Information''}

    <div className="section-wrapper">
      <h5 className="horse-header" id="horse-info-header">Horse Information</h5>
      <div className="horse-info">
        <div className="form-group">
          <label htmlFor="horse-name">Horse Name</label>
          <input type="text" className="form-control" name="horseName" id="horse-name" placeholder="Enter Name" default value={horse ? horse.horseName : undefined} onChange={this.handleChange}>
        </div>
        <handlerDropdown dropdownID="handler-level" currentHandlerLevel={horse ? horse.handlerLevelID : null} required="true" sendData={this.getHandlerData}></handler.Dropdown>
        <locationDropdown dropdownID="daytime-location" required="true" time="AM" location={horse ? horse.dayLocationID : null} sendData={this.getLocationData}></location.Dropdown>
      </div>
    </div>
  </form>
</modal.Body>
</modal>
```

Manage Horses – Edit Horse

<https://horse-care-mgmt-system.herokuapp.com/>
Github: <https://github.com/chantiggi/HCMS-App-CS467>

HCMS Database Schema



Prototype – Manage Horses

