Phase 2: Building Dashboards

The Shrey Method Fitness Platform Guide

← Back to Phase 1 | Next: Phase 3 - Payments →

In this phase, we'll create the dashboards for both clients and coaches. These dashboards will serve as the main interfaces for your fitness platform, allowing:

- Clients to view their programs, track progress, and communicate with you
- You (as the coach) to manage clients, create programs, and monitor progress

We'll break this down into small, manageable steps.

Step 1: Extend Database Schema

First, we need to add more tables to our database to store program information.

1. In the terminal, type:

```
amplify update api
```

2. Select "GraphQL" when prompted
3. Select "Update the GraphQL schema" when asked what you'd like to do
4. It will open the schema file in VS Code
5. Add these new types to the file (after the existing types):

```
type Program @model
@auth(rules: [
  { allow: groups, groups: ["coaches"], operations: [create, read, update, delete]
},
  { allow: owner, operations: [read] }
]) {
  id: ID!
  name: String!
  description: String!
  duration: Int!
  type: String!
  createdAt: AWSDateTime!
}

type ClientProgram @model
@auth(rules: [
  { allow: groups, groups: ["coaches"], operations: [create, read, update, delete]
},
  { allow: owner, operations: [read] }
]) {
```

```
  id: ID!
  clientId: ID!
  programId: ID!
  startDate: AWSDateTime!
  currentWeek: Int!
  status: String!
  nextCheckIn: AWSDateTime
}

type ProgramContent @model
@auth(rules: [
  { allow: groups, groups: ["coaches"], operations: [create, read, update, delete]
},
  { allow: owner, operations: [read] }
]) {
  id: ID!
  programId: ID!
  clientId: ID
  title: String!
  description: String
  fileUrl: String!
  fileType: String!
  week: Int
  createdAt: AWSDateTime!
}
```

6. Save the file (Ctrl+S or File > Save)

What's happening here? You're adding more tables to your database to store information about workout programs, client assignments, and program content.

Step 2: Add Storage for Files

Now let's set up storage for files like workout PDFs, nutrition guides, etc.

1. In the terminal, type:

```
amplify add storage
```

2. Answer the prompts:
   - Select "Content (Images, audio, video, etc.)"
   - Resource name: "shreymethodstorage"
   - Bucket name: "shreymethodfitness-storage"
   - Who should have access: "Auth users only"
   - What kind of access: "read/write"
   - Do you want to add a Lambda Trigger for your S3 Bucket: "No"

What's happening here? You're setting up storage for files like workout PDFs, nutrition guides, etc. This will allow you to upload and share files with your clients.

Step 3: Deploy Your Updated Backend

Now let's update your AWS resources with the new database tables and storage.

1. In the terminal, type:

```
amplify push
```

2. Review the changes and confirm by typing "y"
3. Wait for the deployment to complete

What's happening here? AWS is updating your resources with the new database tables and storage that you configured.

Step 4: Create Dashboard CSS

Let's create the styling for your dashboard pages.

1. Create a new file called `css/dashboard.css`
2. Copy and paste this code:

```
/* Dashboard Layout */
.dashboard-container {
    display: flex;
    min-height: 100vh;
}

.dashboard-sidebar {
    width: 250px;
    background-color: var(--secondary);
    color: white;
    padding: 20px 0;
    display: flex;
    flex-direction: column;
}

.dashboard-content {
    flex: 1;
    background-color: var(--light-gray);
    overflow-y: auto;
}

/* Sidebar Styles */
.logo {
    padding: 0 20px 20px;
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);
}

.logo h2 {
    margin: 0;
```

```
    font-size: 1.5rem;
}

.dashboard-nav {
    margin-top: 20px;
    flex: 1;
}

.dashboard-nav ul {
    list-style: none;
    padding: 0;
    margin: 0;
}

.dashboard-nav li {
    margin-bottom: 5px;
}

.dashboard-nav a {
    display: block;
    padding: 10px 20px;
    color: rgba(255, 255, 255, 0.8);
    text-decoration: none;
    transition: all 0.3s ease;
}

.dashboard-nav a:hover {
    background-color: rgba(255, 255, 255, 0.1);
    color: white;
}

.dashboard-nav li.active a {
    background-color: var(--primary);
    color: white;
}

.sidebar-footer {
    padding: 20px;
    border-top: 1px solid rgba(255, 255, 255, 0.1);
}

/* Content Area Styles */
.dashboard-header {
    background-color: white;
    padding: 20px;
    display: flex;
    justify-content: space-between;
    align-items: center;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.05);
}

.dashboard-header h1 {
    margin: 0;
    font-size: 1.8rem;
```

```
    color: var(--secondary);
}

.user-menu {
    display: flex;
    align-items: center;
}

.user-menu span {
    margin-right: 10px;
}

.avatar {
    width: 40px;
    height: 40px;
    border-radius: 50%;
    object-fit: cover;
}

.content-container {
    padding: 20px;
}

/* Dashboard Cards */
.dashboard-grid {
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
    gap: 20px;
}

.dashboard-card {
    background-color: white;
    border-radius: 10px;
    padding: 20px;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.05);
}

.dashboard-card h2 {
    margin-top: 0;
    margin-bottom: 15px;
    font-size: 1.3rem;
    color: var(--secondary);
}

/* Button Styles */
.btn-secondary {
    background-color: rgba(255, 255, 255, 0.2);
    color: white;
    border: none;
    padding: 8px 15px;
    border-radius: 5px;
    cursor: pointer;
    font-size: 14px;
    transition: background-color 0.3s ease;
```

```
    width: 100%;
}

.btn-secondary:hover {
    background-color: rgba(255, 255, 255, 0.3);
}

.quick-actions {
    display: flex;
    flex-direction: column;
    gap: 10px;
}

.quick-actions button {
    width: 100%;
}

/* Loading Indicator */
.loading {
    color: var(--dark-gray);
    font-style: italic;
    opacity: 0.7;
}

/* Stats Grid */
.stats-grid {
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(100px, 1fr));
    gap: 15px;
}

.stat-item {
    text-align: center;
    padding: 10px;
    background-color: var(--light-gray);
    border-radius: 5px;
}

.stat-item h3 {
    margin: 0 0 5px;
    font-size: 0.9rem;
    color: var(--dark-gray);
}

.stat-item p {
    margin: 0;
    font-size: 1.5rem;
    font-weight: 600;
    color: var(--primary);
}
```

3. Save the file (Ctrl+S or File > Save)

What's happening here? You're creating the CSS styling for your dashboard pages, which includes the layout, sidebar, cards, and other elements.

Step 5: Create Client Dashboard HTML

Now let's create the main dashboard page that clients will see after logging in.

1. Create a new file called `client-dashboard.html`
2. Copy and paste this code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Client Dashboard - The Shrey Method Fitness</title>
    <link rel="stylesheet" href="css/styles.css">
    <link rel="stylesheet" href="css/dashboard.css">
    <script src="https://cdn.jsdelivr.net/npm/aws-amplify@5.0.4/dist/aws-
amplify.min.js"></script>
</head>
<body>
    <div class="dashboard-container">
        <!-- Sidebar navigation -->
        <div class="dashboard-sidebar">
            <div class="logo">
                <h2>The Shrey Method</h2>
            </div>
            <nav class="dashboard-nav">
                <ul>
                    <li class="active"><a href="client-
dashboard.html">Overview</a></li>
                    <li><a href="client-program.html">My Program</a></li>
                    <li><a href="client-nutrition.html">Nutrition</a></li>
                    <li><a href="client-messages.html">Messages</a></li>
                    <li><a href="client-payments.html">Payments</a></li>
                    <li><a href="client-schedule.html">Schedule</a></li>
                </ul>
            </nav>
            <div class="sidebar-footer">
                <button id="logout-btn" class="btn-secondary">Log Out</button>
            </div>
        </div>

        <!-- Main content area -->
        <div class="dashboard-content">
            <header class="dashboard-header">
                <h1>Welcome Back, <span id="client-name">Client</span>!</h1>
                <div class="user-menu">
                    <span id="user-name"></span>
                    <img src="images/default-avatar.jpg" alt="Profile"
class="avatar" id="user-avatar">
```

```
                </div>
            </header>

            <div class="content-container">
                <div class="dashboard-grid">
                    <!-- Program Overview Card -->
                    <div class="dashboard-card">
                        <h2>Current Program</h2>
                        <div id="program-info">
                            <div class="loading">Loading program information...
</div>
                        </div>
                    </div>

                    <!-- Progress Card -->
                    <div class="dashboard-card">
                        <h2>Program Progress</h2>
                        <div id="progress-container">
                            <div class="loading">Loading progress...</div>
                        </div>
                    </div>

                    <!-- Quick Access Card -->
                    <div class="dashboard-card">
                        <h2>Quick Access</h2>
                        <div class="quick-actions">
                            <button id="view-workout-btn" class="btn-primary">View
Today's Workout</button>
                            <button id="log-nutrition-btn" class="btn-primary">Log
Nutrition</button>
                            <button id="message-coach-btn" class="btn-
primary">Message Coach</button>
                        </div>
                    </div>

                    <!-- Recent Files Card -->
                    <div class="dashboard-card">
                        <h2>Recent Files</h2>
                        <div id="recent-files">
                            <div class="loading">Loading recent files...</div>
                        </div>
                    </div>
                </div>
            </div>
        </div>

    <script src="js/client-dashboard.js"></script>
</body>
</html>
```

3. Save the file (Ctrl+S or File > Save)

What's happening here? You're creating the HTML structure for the client dashboard, which includes a sidebar navigation, header, and various cards for displaying information.

Step 6: Create Client Dashboard JavaScript

Now let's create the JavaScript code that will handle the client dashboard functionality.

1. Create a new file called `js/client-dashboard.js`
2. Copy and paste this code:

```
// Configure Amplify
const awsConfig = {
    Auth: {
        region: 'us-west-2', // Your AWS region
        userPoolId: 'us-west-2_xxxxxxxx', // Your Cognito User Pool ID
        userPoolWebClientId: 'xxxxxxxxxxxxxxxxxxxxxxxxxx', // Your App Client ID
    },
    API: {
        endpoints: [{
            name: "shreymethodapi",
            endpoint: "https://xxxxxxxxxx.execute-api.us-west-2.amazonaws.com/dev"
        }]
    }
};

// Replace with your actual values (same as in auth.js)

Amplify.configure(awsConfig);

// Check authentication
async function checkAuth() {
    try {
        const user = await Amplify.Auth.currentAuthenticatedUser();
        return user;
    } catch (error) {
        window.location.href = 'client-login.html';
    }
}

// Initialize
document.addEventListener('DOMContentLoaded', async () => {
    try {
        const user = await checkAuth();

        // Set user name
        const clientName = document.getElementById('client-name');
        const userName = document.getElementById('user-name');

        clientName.textContent = user.attributes.name || user.username;
        userName.textContent = user.attributes.name || user.username;

        // Set up quick action buttons
```

```
        document.getElementById('view-workout-btn').addEventListener('click', ()
=> {
            alert('This feature will be implemented in a future update.');
        });

        document.getElementById('log-nutrition-btn').addEventListener('click', ()
=> {
            alert('This feature will be implemented in a future update.');
        });

        document.getElementById('message-coach-btn').addEventListener('click', ()
=> {
            alert('This feature will be implemented in a future update.');
        });

        // Set up logout
        document.getElementById('logout-btn').addEventListener('click', async ()
=> {
            try {
                await Amplify.Auth.signOut();
                window.location.href = 'client-login.html';
            } catch (error) {
                console.error('Error signing out: ', error);
            }
        });

    } catch (error) {
        console.error('Error initializing dashboard:', error);
    }
});
```

3. Save the file (Ctrl+S or File > Save)
4. Update the configuration values with your actual values (same as in Phase 1, Step 7)

What's happening here? You're creating the JavaScript code that will handle the client dashboard functionality, including authentication, displaying user information, and setting up event listeners for the buttons.

Step 7: Create Coach Dashboard HTML

Now let's create the main dashboard page that you (as the coach) will see after logging in.

1. Create a new file called `coach-dashboard.html`
2. Copy and paste this code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Coach Dashboard - The Shrey Method Fitness</title>
```

```
    <link rel="stylesheet" href="css/styles.css">
    <link rel="stylesheet" href="css/dashboard.css">
    <script src="https://cdn.jsdelivr.net/npm/aws-amplify@5.0.4/dist/aws-
amplify.min.js"></script>
</head>
<body>
    <div class="dashboard-container">
        <!-- Sidebar navigation -->
        <div class="dashboard-sidebar">
            <div class="logo">
                <h2>The Shrey Method</h2>
            </div>
            <nav class="dashboard-nav">
                <ul>
                    <li class="active"><a href="coach-
dashboard.html">Dashboard</a></li>
                    <li><a href="coach-clients.html">Clients</a></li>
                    <li><a href="coach-messages.html">Messages</a></li>
                    <li><a href="coach-programs.html">Programs</a></li>
                    <li><a href="coach-settings.html">Settings</a></li>
                </ul>
            </nav>
            <div class="sidebar-footer">
                <button id="logout-btn" class="btn-secondary">Log Out</button>
            </div>
        </div>

        <!-- Main content area -->
        <div class="dashboard-content">
            <header class="dashboard-header">
                <h1>Coach Dashboard</h1>
                <div class="user-menu">
                    <span id="user-name">Coach</span>
                    <img src="images/default-avatar.jpg" alt="Profile"
class="avatar" id="user-avatar">
                </div>
            </header>

            <div class="content-container">
                <div class="dashboard-grid">
                    <!-- Stats Card -->
                    <div class="dashboard-card">
                        <h2>Client Statistics</h2>
                        <div class="stats-grid">
                            <div class="stat-item">
                                <h3>Active Clients</h3>
                                <p id="active-clients-count">0</p>
                            </div>
                            <div class="stat-item">
                                <h3>New This Month</h3>
                                <p id="new-clients-count">0</p>
                            </div>
                            <div class="stat-item">
                                <h3>Check-ins Today</h3>
```

```
                                <p id="checkins-today-count">0</p>
                            </div>
                        </div>
                    </div>

                    <!-- Quick Actions Card -->
                    <div class="dashboard-card">
                        <h2>Quick Actions</h2>
                        <div class="quick-actions">
                            <button id="add-client-btn" class="btn-primary">Add
New Client</button>
                            <button id="create-program-btn" class="btn-
primary">Create Program</button>
                            <button id="view-messages-btn" class="btn-
primary">View Messages</button>
                        </div>
                    </div>

                    <!-- Recent Activity Card -->
                    <div class="dashboard-card">
                        <h2>Recent Activity</h2>
                        <div id="recent-activity">
                            <p>No recent activity to display.</p>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>

    <script src="js/coach-dashboard.js"></script>
</body>
</html>
```

3. Save the file (Ctrl+S or File > Save)

What's happening here? You're creating the HTML structure for the coach dashboard, which includes a sidebar navigation, header, and various cards for displaying information and actions.

Step 8: Create Coach Dashboard JavaScript

Now let's create the JavaScript code that will handle the coach dashboard functionality.

1. Create a new file called `js/coach-dashboard.js`
2. Copy and paste this code:

```
// Configure Amplify
const awsConfig = {
    Auth: {
        region: 'us-west-2', // Your AWS region
        userPoolId: 'us-west-2_xxxxxxxx', // Your Cognito User Pool ID
```

```
        userPoolWebClientId: 'xxxxxxxxxxxxxxxxxxxxxxxxxx', // Your App Client ID
    },
    API: {
        endpoints: [{
            name: "shreymethodapi",
            endpoint: "https://xxxxxxxxxx.execute-api.us-west-2.amazonaws.com/dev"
        }]
    }
};

// Replace with your actual values (same as in auth.js)

Amplify.configure(awsConfig);

// Check authentication
async function checkAuth() {
    try {
        const user = await Amplify.Auth.currentAuthenticatedUser();

        // Check if user is in coaches group
        const session = await Amplify.Auth.currentSession();
        const idToken = session.getIdToken().payload;

        if (!idToken['cognito:groups'] ||
!idToken['cognito:groups'].includes('coaches')) {
            // Redirect to client dashboard if not a coach
            window.location.href = 'client-dashboard.html';
            return;
        }

        return user;
    } catch (error) {
        window.location.href = 'client-login.html';
    }
}

// Initialize
document.addEventListener('DOMContentLoaded', async () => {
    try {
        const user = await checkAuth();

        // Set user name
        const userName = document.getElementById('user-name');
        userName.textContent = user.attributes.name || user.username;

        // Set up quick action buttons
        document.getElementById('add-client-btn').addEventListener('click', () =>
{
            alert('This feature will be implemented in a future update.');
        });

        document.getElementById('create-program-btn').addEventListener('click', ()
=> {
            alert('This feature will be implemented in a future update.');
```

```
        });

        document.getElementById('view-messages-btn').addEventListener('click', ()
=> {
            alert('This feature will be implemented in a future update.');
        });

        // Set up logout
        document.getElementById('logout-btn').addEventListener('click', async ()
=> {
            try {
                await Amplify.Auth.signOut();
                window.location.href = 'client-login.html';
            } catch (error) {
                console.error('Error signing out: ', error);
            }
        });

    } catch (error) {
        console.error('Error initializing dashboard:', error);
    }
});
```

3. Save the file (Ctrl+S or File > Save)
4. Update the configuration values with your actual values (same as in Phase 1, Step 7)

What's happening here? You're creating the JavaScript code that will handle the coach dashboard functionality, including authentication, checking if the user is a coach, displaying user information, and setting up event listeners for the buttons.

Step 9: Add a User to the Coaches Group

Now let's add your user account to the coaches group so you can access the coach dashboard.

1. In the terminal, type:

```
amplify console auth
```

2. This will open the AWS Cognito console in your browser
3. Click on "User Pools" in the left sidebar
4. Click on your user pool
5. Click on "Users and groups" in the left sidebar
6. Click on the "Groups" tab
7. Click on the "coaches" group
8. Click "Add users to group"
9. Select your user from the list
10. Click "Add"

What's happening here? You're adding your user account to the coaches group so you can access the coach dashboard.

Step 10: Create Images Folder

Let's add a default avatar image that will be displayed in the dashboard.

1. Create a folder called `images` in your project folder
2. Add a default avatar image to this folder (you can use any image or download one from the internet)
3. Rename it to `default-avatar.jpg`

What's happening here? You're adding a default avatar image that will be displayed in the dashboard.

Step 11: Test Both Dashboards

Now let's test your dashboards to make sure everything works correctly.

1. Open your website in a browser:
    - Right-click on `client-login.html` in VS Code
    - Select "Open with Live Server" (if you have the Live Server extension)
    - Or open the file directly in your browser
2. Log in with your account:
    - You should be redirected to the coach dashboard (since you're in the coaches group)
    - Check that your name is displayed correctly
    - Test the logout button
3. Create a new user account (if you haven't already):
    - Click "Sign Up" on the login page
    - Fill out the form with a different name and email
    - Verify the account
4. Log in with the new account:
    - You should be redirected to the client dashboard (since the new user is not in the coaches group)
    - Check that the name is displayed correctly
    - Test the logout button

What's happening here? You're testing both dashboards to make sure they work correctly and that users are redirected to the appropriate dashboard based on their group.

Next Steps

In this phase, we've created the basic structure for both the client and coach dashboards. In a real-world scenario, you would now implement the functionality for each section of the dashboard, such as:

- Displaying and managing client programs
- Uploading and sharing workout and nutrition files
- Implementing messaging between clients and coaches
- Creating and assigning workout programs

However, for the purposes of this guide, we'll move on to the next phase: adding payment processing with Stripe.

Congratulations!

You've completed Phase 2 of your fitness platform. You now have basic dashboards for both clients and coaches. In the next phase, we'll add payment processing capabilities to your platform.

← Back to Phase 1 | Next: Phase 3 - Payments →