# Assassin's Creed Wiki Website - Complete Implementation Plan

## 1. Project Summary

You'll build a static Assassin's Creed fan wiki using only HTML, CSS, and vanilla JavaScript. The site will display game information, character profiles, locations, and a timeline—all loaded from a local JSON file. No frameworks, no server required—just open `index.html` in your browser. This project reinforces DOM manipulation, event handling, fetch API, and responsive design fundamentals.

---

## 2. Feature List

### Required Features (Must Have)

- **Home page** with hero section and featured games grid
- **Games list page** showing all AC games with thumbnails and basic info
- **Game detail page** displaying full game information (year, platform, description)
- **Characters page** listing characters with filtering by game
- **Responsive navigation bar** with mobile hamburger menu
- **Search functionality** to find games or characters by name
- **Basic accessibility** (semantic HTML, keyboard navigation, alt text)

### Optional Features (Nice to Have)

- **Timeline page** showing chronological order of games/historical events
- **Bookmarks/favorites** using localStorage to save user preferences
- **Dark/light theme toggle** persisted in localStorage
- **Smooth scroll animations** and hover effects
- **Location pages** with map descriptions
- **"Back to top" button** for long pages

---

## 3. File & Folder Structure

```
ac-wiki/
├── index.html
├── games.html
├── game-detail.html
├── characters.html
├── timeline.html (optional)
├── css/
│   ├── style.css
│   ├── responsive.css
│   └── components.css
```

```
├── js/
│   ├── main.js
│   ├── data-loader.js
│   ├── render.js
│   ├── search.js
│   ├── nav.js
│   └── bookmarks.js (optional)
├── data/
│   └── ac_wiki.json
├── img/
│   ├── placeholder.jpg
│   └── (user-created images here)
└── README.md
```

---

# 4. Data Model (JSON Schema)

### File: `data/ac_wiki.json`

```
{
  "games": [
    {
      "id": "ac1",
      "title": "Assassin's Creed",
      "year": 2007,
      "platforms": ["Xbox 360", "PS3", "PC"],
      "setting": "Third Crusade, Holy Land",
      "protagonist": "Altaïr Ibn-La'Ahad",
      "description": "The first game in the series follows Altaïr, a
skilled assassin during the Third Crusade.",
      "image": "img/ac1.jpg"
    },
    {
      "id": "ac2",
      "title": "Assassin's Creed II",
      "year": 2009,
      "platforms": ["Xbox 360", "PS3", "PC"],
      "setting": "Renaissance Italy",
      "protagonist": "Ezio Auditore",
      "description": "Follow Ezio's journey from a young nobleman to a
Master Assassin in Renaissance Italy.",
      "image": "img/ac2.jpg"
    }
  ],
  "characters": [
    {
      "id": "char1",
      "name": "Altaïr Ibn-La'Ahad",
      "gameId": "ac1",
      "role": "Protagonist",
      "description": "A member of the Assassin Order during the Third
Crusade, known for his skills and arrogance.",
      "image": "img/altair.jpg"
    },
    {
      "id": "char2",
      "name": "Ezio Auditore da Firenze",
      "gameId": "ac2",
      "role": "Protagonist",
```

```
      "description": "A Florentine nobleman who becomes a legendary Master
Assassin during the Italian Renaissance.",
      "image": "img/ezio.jpg"
    }
  ],
  "locations": [
    {
      "id": "loc1",
      "name": "Jerusalem",
      "gameId": "ac1",
      "period": "1191 AD",
      "description": "A holy city during the Third Crusade, divided into
multiple districts."
    },
    {
      "id": "loc2",
      "name": "Florence",
      "gameId": "ac2",
      "period": "1476 AD",
      "description": "The birthplace of the Renaissance, home to artists,
merchants, and the Medici family."
    }
  ],
  "timeline": [
    {
      "year": 2007,
      "event": "Assassin's Creed released",
      "type": "game"
    },
    {
      "year": 2009,
      "event": "Assassin's Creed II released",
      "type": "game"
    }
  ]
}
```

---

# 5. UI Wireframes & Components

## Navigation Bar

- **Elements**: `<nav>`, `<ul>`, `<li>`, `<a>`, hamburger button (`<button>`)
- **ARIA**: `role="navigation"`, `aria-label="Main navigation"`, `aria-expanded` on mobile menu button

## Home Page

- **Hero section**: `<header>` with `<h1>`, subtitle, background image
- **Featured games grid**: `<section>` with `<div class="card">` for each game
- **Elements**: `<img>`, `<h3>`, `<p>`, `<a>` (Read More button)

## Games List Page

- **Filter bar**: `<form>` with `<input type="search">`, filter dropdowns

- **Game grid**: Repeating card components
- **ARIA**: `role="search"` for search form, `aria-label` on inputs

### Game Detail Page

- **Layout**: Large hero image, game info table, description section
- **Elements**: `<main>`, `<article>`, `<dl>` (description list for specs), `<section>`
- **ARIA**: `role="main"` on main content area

### Characters Page

- **Filter by game**: `<select>` dropdown
- **Character cards**: Grid of cards with image, name, role
- **ARIA**: `role="list"` and `role="listitem"` if not using `<ul>`

### Timeline Page (Optional)

- **Vertical timeline**: `<ol>` with styled list items
- **Elements**: Year badges, event descriptions
- **ARIA**: `role="list"` with proper heading structure

### Search UI

- **Search bar**: Sticky or in navbar
- **Results dropdown**: `<div>` that appears below input with live results
- **ARIA**: `role="combobox"`, `aria-autocomplete="list"`, `aria-controls` linking to results

---

# 6. Routing Approach

**Recommended: Multi-page static HTML files**

**Why?** For a first-semester student:

- **Simpler mental model**: Each page is a separate `.html` file—easy to understand
- **No complex routing logic**: Browser handles navigation automatically
- **Easier debugging**: View source shows exactly what's on the page
- **SEO-friendly**: Each page has its own URL and metadata

**How it works:**

1. Create `index.html`, `games.html`, `game-detail.html`, `characters.html`
2. Use URL parameters for dynamic content: `game-detail.html?id=ac1`
3. JavaScript reads `window.location.search` to get the `id` parameter
4. Load JSON data and display the matching game/character

**Alternative (more advanced):** Hash-based SPA routing (`#/games`, `#/game/ac1`) using `window.location.hash` and event listeners, but this adds complexity without server-side benefits.

---

# 7. Core JavaScript Responsibilities

**`js/main.js`**

- Entry point that runs on page load
- Initializes other modules (calls `loadData()`, `initSearch()`, `initNav()`)
- Sets up event listeners for theme toggle, back-to-top button

**`js/data-loader.js`**

- **Function**: `loadData()` - fetches `data/ac_wiki.json` using `fetch()`
- Returns promise that resolves to parsed JSON object
- Handles errors (shows message if file missing)
- Caches data in memory to avoid repeated fetches

**`js/render.js`**

- **Functions**: `renderGameCard(game)`, `renderCharacterCard(character)`, `renderGameDetail(game)`
- Takes data objects and returns HTML strings or DOM elements
- Inserts rendered content into page using `innerHTML` or `appendChild()`
- Handles missing images with placeholder fallback

**`js/search.js`**

- **Function**: `initSearch()` - sets up search input listener
- **Function**: `performSearch(query)` - filters games/characters by name
- Displays live search results in dropdown
- Debounces input (waits 300ms after typing stops) to improve performance

**`js/nav.js`**

- **Function**: `initNav()` - sets up mobile hamburger menu toggle
- Adds active class to current page link
- Handles keyboard navigation (Tab, Enter, Escape)
- Closes mobile menu when clicking outside

**`js/bookmarks.js` (Optional)**

- **Function**: `addBookmark(id)`, `removeBookmark(id)`, `getBookmarks()`
- Uses `localStorage.setItem()` and `localStorage.getItem()`
- Stores array of favorite game/character IDs as JSON string
- Updates UI to show bookmarked items with a star icon

---

# 8. UI/UX & Accessibility Rules

1. **Use semantic HTML**: `<header>`, `<nav>`, `<main>`, `<article>`, `<section>`, `<footer>` for structure
2. **All images must have `alt` text**: Describe what's shown, or use `alt=""` for decorative images
3. **Keyboard navigation**: All interactive elements must be reachable with Tab key and activated with Enter/Space
4. **Color contrast**: Text must have 4.5:1 contrast ratio with background (use a contrast checker)
5. **Focus indicators**: Never remove outline with `outline: none` without providing visible alternative
6. **Skip to main content link**: Add invisible link at top that becomes visible on focus
7. **Descriptive link text**: Avoid "click here"—use "View Ezio's profile" instead
8. **Form labels**: Every `<input>` must have an associated `<label>` with matching `for` attribute

---

# 9. Styling Guidance

## Layout Strategy

- **Container**: Max-width 1200px, centered with `margin: 0 auto`
- **Navigation**: Use flexbox with `justify-content: space-between`
- **Card grids**: CSS Grid with `grid-template-columns: repeat(auto-fill, minmax(280px, 1fr))`
- **Game detail**: Two-column layout with `display: grid; grid-template-columns: 2fr 1fr`

## Responsive Breakpoints

```
/* Mobile first approach */
/* Base: 320px+ (mobile) */
@media (min-width: 768px) { /* Tablet */ }
@media (min-width: 1024px) { /* Desktop */ }
```

## Typography Scale

```
:root {
  --font-base: 16px;
  --font-small: 0.875rem;   /* 14px */
  --font-medium: 1rem;       /* 16px */
  --font-large: 1.25rem;     /* 20px */
  --font-xl: 1.75rem;        /* 28px */
  --font-2xl: 2.5rem;        /* 40px */
}
```

## Color Palette (Assassin's Creed Theme)

```
:root {
  --color-primary: #C9A961;    /* Gold */
  --color-secondary: #1A1A1A;  /* Almost Black */
  --color-accent: #8B0000;     /* Dark Red */
  --color-bg: #F5F5F5;         /* Light Gray */
  --color-text: #333333;       /* Dark Gray */
  --color-border: #DDDDDD;     /* Light Border */
}
```

### Animation Suggestions

- **Card hover**: `transform: translateY(-5px); box-shadow: 0 8px 16px rgba(0,0,0,0.2);` with `transition: all 0.3s ease`
- **Button hover**: Scale slightly with `transform: scale(1.05)`
- **Focus states**: Add animated outline with `box-shadow: 0 0 0 3px rgba(201,169,97,0.4)`
- **Page transitions**: Fade in main content with CSS animation on page load

---

# 10. Performance & Size Constraints

- **Target page weight**: Keep total page under 500KB (HTML + CSS + JS + initial images)
- **Image optimization**: Compress JPGs to 80% quality, use WebP if possible, max width 1200px
- **Lazy loading**: Add `loading="lazy"` to all `<img>` tags not in viewport
- **JSON efficiency**: Keep descriptions under 200 characters, limit to 10-15 games/characters for demo
- **CSS/JS minification**: Not required for student project, but remove comments in final version
- **Defer non-critical JS**: Add `defer` to script tags in `<head>`

---

# 11. SEO & Metadata

### Rules

- Each page must have unique `<title>` tag (50-60 characters)
- Include `<meta name="description">` (150-160 characters)
- Add Open Graph tags for social media sharing
- Use descriptive heading hierarchy (`<h1>` once per page, then `<h2>`, `<h3>`)

### Example (Game Detail Page)

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Assassin's Creed II - AC Wiki</title>
```

```
  <meta name="description" content="Learn about Assassin's Creed II,
featuring Ezio Auditore in Renaissance Italy. Released 2009 for Xbox 360,
PS3, and PC.">
  <meta property="og:title" content="Assassin's Creed II - AC Wiki">
  <meta property="og:description" content="Explore the story of Ezio
Auditore in Renaissance Italy.">
  <meta property="og:image" content="img/ac2.jpg">
</head>
```

---

# 12. Testing Checklist

### Functionality Tests

- [ ] All navigation links lead to correct pages
- [ ] URL parameters work (e.g., `game-detail.html?id=ac2` loads correct game)
- [ ] Search returns accurate results for games and characters
- [ ] Filter dropdown on characters page works correctly
- [ ] Bookmarks save and load from localStorage (if implemented)

### Accessibility Tests

- [ ] Tab through entire page—all interactive elements are reachable
- [ ] Press Enter/Space on links and buttons—they activate correctly
- [ ] View with images disabled—alt text appears
- [ ] Check color contrast with WebAIM contrast checker tool
- [ ] Test with screen reader (NVDA on Windows or VoiceOver on Mac)

### Responsive Tests

- [ ] View on mobile (375px width)—hamburger menu works, content stacks vertically
- [ ] View on tablet (768px)—layout adjusts appropriately
- [ ] View on desktop (1920px)—content doesn't stretch too wide
- [ ] Rotate device—content reflows correctly

### Browser Tests

- [ ] Works in Chrome/Edge
- [ ] Works in Firefox
- [ ] Works in Safari
- [ ] Console shows no JavaScript errors

---

# 13. Grading Rubric (100 Points Total)

## Functionality (60 points)

1. **Core pages work** (20pts): All HTML pages load, navigation works, content displays
2. **Data loading** (20pts): JSON file loads successfully, data renders on all pages

3. **Interactive features** (20pts): Search works, filters work, URL parameters load correct content

### Code Quality (20 points)

1. **Clean HTML** (7pts): Proper indentation, semantic tags, no duplicate IDs
2. **Organized CSS** (7pts): Logical structure, uses variables, no !important overuse
3. **Readable JavaScript** (6pts): Functions are named clearly, includes comments, no global variables

### Accessibility (10 points)

1. **Semantic structure** (3pts): Uses header, nav, main, footer correctly
2. **Keyboard navigation** (4pts): Can tab through site, focus indicators visible
3. **Image alt text** (3pts): All images have appropriate alt attributes

### Responsiveness (10 points)

1. **Mobile layout** (4pts): Works on 375px width, menu is usable
2. **Tablet layout** (3pts): Content adapts at 768px breakpoint
3. **Desktop layout** (3pts): Looks good at 1920px, uses available space well

---

## 14. Starter Tasks / Step-by-Step Plan

**Estimated time: 8-12 hours**

### Milestone 1: Setup & Structure (1 hour)

1. Create folder structure and all HTML files
2. Add basic HTML5 boilerplate to each page
3. Link CSS and JS files in `<head>`
4. Create placeholder JSON file with 2 games, 2 characters

### Milestone 2: Navigation & Layout (2 hours)

5. Build navbar HTML with logo and links
6. Style navbar with flexbox, make it sticky
7. Create hamburger menu for mobile with JavaScript toggle
8. Add footer to all pages

### Milestone 3: Data Loading (1.5 hours)

9. Write `loadData()` function in `data-loader.js`
10. Test JSON loading in console
11. Handle errors (file not found message)

## Milestone 4: Home Page (1.5 hours)

12. Create hero section with background image
13. Render featured games grid from JSON data
14. Add hover effects to game cards

## Milestone 5: Games List & Detail (2 hours)

15. Build games list page with all games rendering
16. Create game detail page template
17. Add URL parameter reading (`?id=ac1`)
18. Render correct game data on detail page

## Milestone 6: Characters Page (1.5 hours)

19. Render all characters as cards
20. Add filter dropdown by game
21. Implement filter JavaScript logic

## Milestone 7: Search Feature (1.5 hours)

22. Add search bar to navbar
23. Implement search function that filters games/characters
24. Display results in dropdown below search input

## Milestone 8: Styling & Polish (2 hours)

25. Apply color palette throughout site
26. Add responsive breakpoints for mobile/tablet
27. Implement hover animations and transitions
28. Test all pages for visual consistency

## Milestone 9: Testing & Accessibility (1 hour)

29. Run through testing checklist
30. Fix any broken links or JavaScript errors
31. Add missing alt text and ARIA labels
32. Test keyboard navigation

---

**What you CAN use:**

- **Your own screenshots**: If you own the game, capture your own images
- **Public domain images**: Historical paintings/photos of locations (Florence, Jerusalem)
- **Placeholder images**: Use `https://via.placeholder.com/300x200?text=AC+Game` for development
- **Your own descriptions**: Write summaries in your own words based on public knowledge

- **Creative Commons images**: Search Wikimedia Commons for historical imagery
- **Icon fonts**: Use free icon libraries like Font Awesome for UI elements

**Best practice**: Use placeholder images during development, then replace with properly licensed assets or your own photos before publishing. Include a `credits.txt` file listing image sources.

---