Team S.E.E.M.

Scott, Edgar, Elvin, Malek

Presentation Overview

- Project Description
- Resources/Languages Used
- Requirements
- Features
- Front-End
- Back-End
- Implementation
- Testing
- Demo

Objectives

- Provide a simple diversion and entertainment
- Achieve an increased understanding in web programming languages and environment
- Achieve good work ethic while working as a team.

Stories

Our Project: Stories

- Fun interactive website for creating or altering already made stories.
  - It works similar to mad libs, but with more customizability.
- It is simple to use and instructions are provided to get the full experience.
- Anyone should be able to jump aboard and start creating.

Resources/Languages used

- HTML, CSS, Javascript
  - Templated.co for the initial website template
- PHP, PHPUnit, PHPStorm
- MySQL
- XAMPP
- GitHub for file sharing
- Trello
- Notepad++, ATOM

Requirements

- Customers
  - Each group member participated as a customer
    - Conflict of interest
    - Verification and Validation
- Non-Functional Requirements
  - Performance
  - Security
  - Recoverability
  - Data Integrity

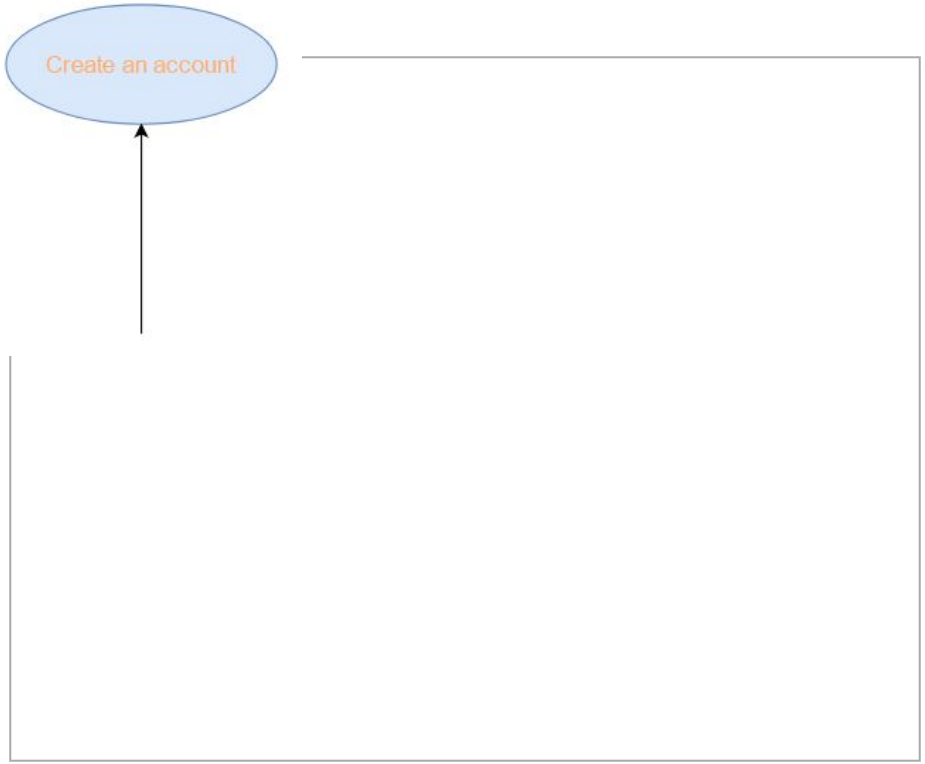
Requirements Cont

- Functional Requirements
  - Allowing users to register for an account
  - Accept User inputs for stories
    - Display input on a templated story
    - Edit your own stories/profile info
- Architectural Requirements
  - Designed on behavior of our website
  - User -> webpage -> server -> database -> display

Features

- Account system
- Allow users to fill in blanks with their own words, which are then shown in the complete story.
- Allow users to create their own stories and allow them to place blanks where they want to.
- Submitted stories are placed on the main page where anyone else can access them.
- User accounts to save any created stories.
- Personalize profile with choice picture and name. Both can be freely changed at user request.

Use Case Diagram



Development

Agile

- No specific checklist or procedure.
- Develop in small increments
- Customer kept changing its mind.
- XP- almost always worked in pairs

Switched to Plan-Driven

- Time was running short.
- Detailed plans, workflow, roles, responsibilities.
- software system architecture defined up-front

Front End Goals

- Incorporate a template as a starting point
  - Choose the the most visually appealing template
- Achieve a simplistic modern look that anyone could easily follow
- Have drop down menu button follows you as you scroll through the page
- Pictures that correlate to the story
- Page Content

Designing Front End

- Choose a visually appealing template
  - Modified HTML and CSS.
    - Difficult at times.
- Implement Pictures
  - Adds visual element to stories
    - Helps give a preview of what Stories are about
- Work together to create a sleek look overall that flows with interface and is pleasing for users
- Oversee page functionality and create page content
- Make design changes at the end

Back End

- Create a database using XAMPP
  - Information stored and pulled from Database
  - Includes the stories and pictures
- Contacting us
  - Create email
  - Sending email
- Javascript
  - Working with information pulled

Implementation

- Issues
  - Sessions- long delay
  - Breaking of previously working code
  - Getting correct display was tricky

- Solutions
  - Eventually delivered
  - Regressed to previously working code
  - Simple number changes to have correct display

Testing

- PHPUnit was used to test php code.

```php
// Testing to make sure chr limit is not smaller than 2 and not
// bigger than 20, to simulate what the database restricts on char limit
public function testcheckIfNameUpdated4()
{
    $str = "malek";
    $result = checkIfNameUpdated($str, 1);
    $result = $this->valiStrLen($str, 2, 20);
    $this->assertEquals(TRUE, $result);
}
```

```php
// Make sure that records is not updated when blank string
public function testcheckIfNameUpdated2()
{
    $result = checkIfNameUpdated("", 1);
    $this->assertEquals("nothing to submit", $result);
}
```

Demo Time