**Process Plan for Meme Doppler**

**1. Introduction**

The Meme Doppler platform aims to combine the functionality of a weather app with the entertainment of meme culture given the pervasiveness of the meme through the millennial generation.

The primary scope is to provide local weather from reliable sources filtered through the DANK algorithm to produce a memed output as well as allowing users to affect the DANK algorithm to identify the most appropriate meme for a given set of temperature/weather conditions. The software will also alert the user of any emergency weather alerts in their area.

The software will not be a weather database aggregate, that is it does not care about analyzing the actual weather data from its sources, just interpreting it to be output as memes for the end user.

**1.1 Definitions and Acronyms**

EOF - End Of Finals

DANK - Definitely Appropriate Noteworthy Knowledge

STASH - SToring And Sharing Hoard

PEPE - Project Execution and Patching Errors

GPS - Global Positioning System

IP - Internet Protocol

**2. Process Description**

**2.1 Project Lifecycle**

During the design, implementation, and maintenance of our project, our team will have to sit down and reevaluate the direction, design, and development of our growing product. This will hopefully happen more during the beginning of our journey, but will continue all the way through to the end of our product’s lifestyle. With this in mind, we will use a lifestyle that will start most closely resembling Iterative, and will later transition to an Agile lifecycle as the project matures and we enter the maintenance and feature development stage..

**2.2 Process Activities**

1. **Locate User**

Acquire user’s location from GPS or IP address.

1. **Acquire Relevant Weather Data**

Query weather database for current data.

1. **Generate Weather Score**

From the current weather, an algorithm will determine the weather score.

1. **Get Meme’d On**

Select the meme corresponding to the weather score from the meme STASH, as well as the three other closest memes.

1. **Prepare Meme**

Generate text to be added to the meme. This will be related to current weather conditions.

1. **Render Meme**

Send weather data and the selected meme and text to user.

1. **Gather User Feedback**

Offer two buttons with “DANK” or “Not DANK” for users to choose. If DANK is chosen, then the meme is appropriate. If Not DANK is chosen, then users will be offered several options as to why the meme was not appropriate, such as inaccuracy, offensiveness, or outdatedness. After selecting their option, they will be offered the three other closest memes to select which would be more appropriate, as well as a “None of the above” option.

**3. Roles**

**3.1 Roles Table**

|  |  |
| --- | --- |
| **Role** | **Responsibility** |
| Project Manager | Coordinate communications between leads. Coordinate meetings and set goals for individual leads. |
| Requirements Lead | Set reasonable goals for development of entire project based upon current progress. |
| Quality Assurance Lead | Ensure that requirements are being met without errors. Test real code from development engineers. In charge of PEPE. |
| Design Lead | Brainstorm ideas to be implemented and organize layout of the interactive application. Analyze current product needs and plan further development. Responsible for design of new features. |
| Implementation Lead | Take input from requirements and design branches and feedback from QA to form pseudocode framework. |
| Full Stack Developer | Take pseudocode from implementation lead and translate into real code. |

#### 

**3.2 Role Assignment Table**

|  |  |
| --- | --- |
| **Team Member** | **Role** |
| Noah Brayer | Requirements Lead / Full Stack Developer |
| Dennis Donoghue | Implementation Lead / Full Stack Developer |
| Chad Klinefelter | Project Manager / QA Lead / Full Stack Developer |
| Kevin Moriarty | Design Lead / Full Stack Developer |

**4. Estimates**

*Estimated lines of code:* Approximately 8000; our software will use a bevy of existing resources for the actual gathering of weather data, doppler map, etc. Therefore our code will not be as robust as that of a program which independently provides these functions.

*Estimated defects:* We expect a total of a few (2-3) hundred defects with the majority being small errors that are easily rectified with several larger defects which take multiple tens of man-hours to rewrite or work around.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Task** | **Estimated Effort (person-days)** | **Start Date (DD/MMM/YY)** | **End Date (DD/MMM/YY)** | **Person** | **Actual Effort (man-hrs)** |
| System Design | 8 | 25SEP16 | 10OCT16 | D, N, K, C |  |
| Detailed Design | 14 | 10OCT16 | 10NOV16 | D, N, K, C |  |
| Code Input Module | 8 | 01NOV16 | 30NOV16 | D, K |  |
| Code Output Module | 8 | 01NOV16 | 30NOV16 | D, K |  |
| Code Map Module | 12 | 01NOV16 | 30NOV16 | N, K |  |
| Code Framework | 10 | 01NOV16 | 30NOV16 | D, N |  |
| Test Planning | 3 | 15NOV16 | 30NOV16 | N, K, C |  |
| Testing and Integration | 5 | 30NOV16 | 10DEC16 | N, C |  |
| Rework and Final | 3 | 10DEC16 | EOF | D, N, K, C |  |