

Team Pluvali

Alex Davis

Steve Kosovich

Tim Leikam

Greg Martini

Debra Parcheta

Katie Taliercio

Specifications:

The game will be implemented using MySQL, Django, HTML, and CSS. The MySQL database will contain user information, such as username, password, email, player points, and preferences. We will then implement a user interface through Django and HTML to interact with the database and provide a choose-your-own-adventure style game. A content creation tool will be made to handle the making new levels. This will just be done using the languages above. The game will save content progress through the use of tables in the database.

Database Considerations:

My Structured Query (MySQL):

pros: Portable across different DBMS. Designed with a focus on the web. Large amount of support and reliable databases, typically.

cons: Not as mature as other relational database management systems (Didn't start as a RDBMS but later changed to, but regardless shouldn't effect a project of our scale). Functionality can be dependent on add-ons (again shouldn't affect us).

SQLite:

pros: East start-up and great testing.

cons: no user management nor does it have very many performance optimization features.

Java Database Connectivity (JDBC):

pros: Portable across different DBMS. Solid performance.

cons: Should have a good understanding of Java

Open Database Connectivity (ODBC):

pros: Portable across different DBMS.

cons: Different drivers needed for different problems. When so many drivers are being used, an overhead of managing them is also increased (shouldn't effect a small project like this).

Database Interface Considerations:

Django:

pros: Easy to learn. Build custom framework for only what we need. More maintained

than PHP. Easier readability.
cons: Would have to show the future support of the program how to familiarize themselves with the custom framework.

PHP:

pros: Have a little previous experience already. Easy to learn.
cons: Maintainability can get complex.

Rails:

pros: Slight edge over Django for web development.
cons: Can be a slow learning curve.

Game Framework Considerations:

HTML:

pros: Easy to learn. Compatible everywhere.
cons: Can't support highly dynamic interfaces as well.

Java:

pros: More processing power.
Cons: iPhone doesn't support Java.

Unity:

pros: Strong environment with plenty of support.
cons: Wouldn't work for mobile systems. Too much for project scope.

User Progress Tracking Considerations:

Database Tables:

pros: Simple setup and future proof.
cons: If too many features added, unnecessary size.

Data Strings:

pros: Would have easy to set up and quick load times. Not device dependent.
cons: Creating a way to have user made scenarios auto implement this system without user making the data strings (future proof).

Cookies:

pros: The browser already makes use of cookies, this wouldn't be hard for the game to as well.
cons: If user deletes cookies without intending to delete save game. User account linked to specific device.

Linked with IP Address:

pros: Each IP is unique so easy to distinguish player.
cons: Multiple people couldn't play from same network mask.

Resources:

First deployment of the database will be on our local computers until a server is set up at the DDRC.