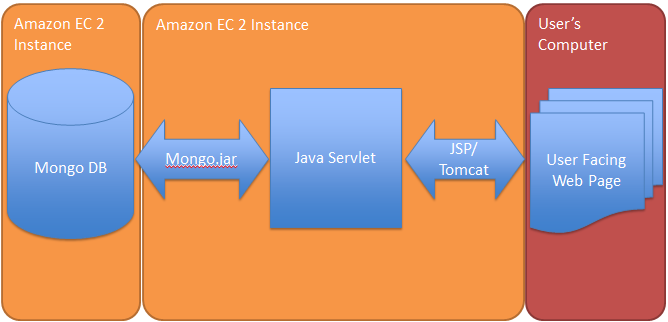
 The basics of our system will run on an Amazon EC2 instance. The system will be written as a Java Enterprise Edition (Java EE) application and run in a Tomcat server. The server will communicate with a MongoDB instance running on a second Amazon EC2 instances. The connection between the EC2 instances will be maintained by the java MongoDB driver. User facing web pages will be generated through the Tomcat server using JavaServer Pages (JSP) and dynamically generated HTML and JavaScript. The database will maintain two collections, the first containing user information and the second containing project information. The Users collection will contain relevant skills, personal information, interests, and current and former projects. The Projects collection will contain project information, IDs of current members, and project needs. The de-normalization of projects, skills, and members allows for efficient indexing within MongoDB by directly exposing relevant project and skill information directly with the user they are incorporated with. This process will increase the amount of time per write by needing to write to multiple places, but changes in skills and projects will be infrequent, thus not causing significant overhead created by the de-normalization.

Example Document for Projects Collection. In practice, members will be mongo objectIDs instead of integers.

Example Document for Users Collection. In practice passwords will be stored as MD5 hash + salts and project\_ids will be mongo objectIDs, instead of integers.



Overall System Schema. User facing web pages will be generated through dynamically generated HTML and javascript through JSP running in a Tomcat web server running in an Amazon EC2 instance. The Tomcat web server will run java servlets which interact with a MongoDB instance running in a separate Amazon EC2 instance.